PART II

UNCONSTRAINED OPTIMIZATION

CHAPTER 6

BASICS OF SET-CONSTRAINED AND UNCONSTRAINED OPTIMIZATION

6.1 Introduction

In this chapter we consider the optimization problem

$$\begin{aligned} &\text{minimize} && f(\boldsymbol{x}) \\ &\text{subject to} && \boldsymbol{x} \in \Omega. \end{aligned}$$

The function $f : \mathbb{R}^n \to \mathbb{R}$ that we wish to minimize is a real-valued function called the *objective function* or *cost function*. The vector $\boldsymbol{x}$ is an n-vector of independent variables: $\boldsymbol{x} = [x_1, x_2, \ldots, x_n]^\top \in \mathbb{R}^n$. The variables $x_1, \ldots, x_n$ are often referred to as *decision variables*. The set Ω is a subset of $\mathbb{R}^n$ called the *constraint set* or *feasible set*.

The optimization problem above can be viewed as a decision problem that involves finding the "best" vector $\boldsymbol{x}$ of the decision variables over all possible vectors in Ω. By the "best" vector we mean the one that results in the smallest value of the objective function. This vector is called the *minimizer* of f over Ω. It is possible that there may be many minimizers. In this case, finding any of the minimizers will suffice.

An Introduction to Optimization, Fourth Edition.
By E. K. P. Chong and S. H. Żak.

There are also optimization problems that require maximization of the objective function, in which case we seek *maximizers*. Minimizers and maximizers are also called *extremizers*. Maximization problems, however, can be represented equivalently in the minimization form above because maximizing f is equivalent to minimizing $-f$. Therefore, we can confine our attention to minimization problems without loss of generality.

The problem above is a general form of a *constrained optimization problem*, because the decision variables are constrained to be in the constraint set Ω. If $\Omega = \mathbb{R}^n$, then we refer to the problem as an *unconstrained optimization problem*. In this chapter we discuss basic properties of the general optimization problem above, which includes the unconstrained case. In the remaining chapters of this part, we deal with iterative algorithms for solving unconstrained optimization problems.

The constraint "$\boldsymbol{x} \in \Omega$" is called a *set constraint*. Often, the constraint set Ω takes the form $\Omega = \{\boldsymbol{x} : \boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{0},\ \boldsymbol{g}(\boldsymbol{x}) \leq \boldsymbol{0}\}$, where $\boldsymbol{h}$ and $\boldsymbol{g}$ are given functions. We refer to such constraints as *functional constraints*. The remainder of this chapter deals with general set constraints, including the special case where $\Omega = \mathbb{R}^n$. The case where $\Omega = \mathbb{R}^n$ is called the *unconstrained* case. In Parts III and IV we consider constrained optimization problems with functional constraints.

In considering the general optimization problem above, we distinguish between two kinds of minimizers, as specified by the following definitions.

Definition 6.1 Suppose that $f : \mathbb{R}^n \to \mathbb{R}$ is a real-valued function defined on some set $\Omega \subset \mathbb{R}^n$. A point $\boldsymbol{x}^* \in \Omega$ is a *local minimizer* of f over Ω if there exists $\varepsilon > 0$ such that $f(\boldsymbol{x}) \geq f(\boldsymbol{x}^*)$ for all $\boldsymbol{x} \in \Omega \setminus \{\boldsymbol{x}^*\}$ and $\|\boldsymbol{x} - \boldsymbol{x}^*\| < \varepsilon$. A point $\boldsymbol{x}^* \in \Omega$ is a *global minimizer* of f over Ω if $f(\boldsymbol{x}) \geq f(\boldsymbol{x}^*)$ for all $\boldsymbol{x} \in \Omega \setminus \{\boldsymbol{x}^*\}$. ∎

If in the definitions above we replace "$\geq$" with ">," then we have a *strict local minimizer* and a *strict global minimizer*, respectively. In Figure 6.1, we illustrate the definitions for $n = 1$.

If $\boldsymbol{x}^*$ is a global minimizer of f over Ω, we write $f(\boldsymbol{x}^*) = \min_{\boldsymbol{x} \in \Omega} f(\boldsymbol{x})$ and $\boldsymbol{x}^* = \arg\min_{\boldsymbol{x} \in \Omega} f(\boldsymbol{x})$. If the minimization is unconstrained, we simply write $\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}} f(\boldsymbol{x})$ or $\boldsymbol{x}^* = \arg\min f(\boldsymbol{x})$. In other words, given a real-valued function f, the notation $\arg\min f(\boldsymbol{x})$ denotes the *argument* that minimizes the function f (a point in the domain of f), assuming that such a point is unique (if there is more than one such point, we pick one arbitrarily). For example, if $f : \mathbb{R} \to \mathbb{R}$ is given by $f(x) = (x+1)^2 + 3$, then $\arg\min f(x) = -1$. If we write $\arg\min_{x \in \Omega}$, then we treat "$x \in \Omega$" to be a constraint for the minimization. For example, for the function f above, $\arg\min_{x \geq 0} f(x) = 0$.

Strictly speaking, an optimization problem is solved only when a global minimizer is found. However, global minimizers are, in general, difficult to find. Therefore, in practice, we often have to be satisfied with finding local minimizers.

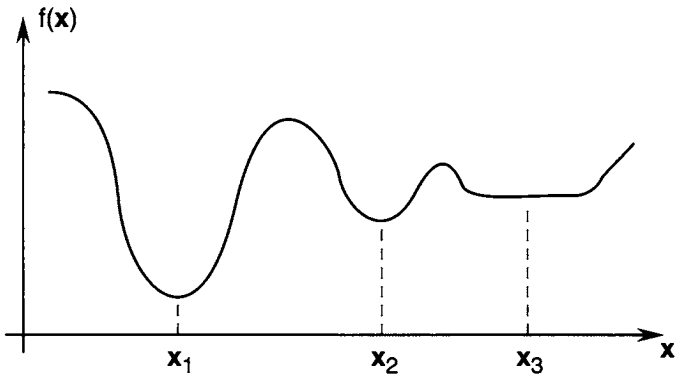

Figure 6.1 Examples of minimizers: $\boldsymbol{x}_1$: strict global minimizer; $\boldsymbol{x}_2$: strict local minimizer; $\boldsymbol{x}_3$: local (not strict) minimizer.

6.2 Conditions for Local Minimizers

In this section we derive conditions for a point $\boldsymbol{x}^*$ to be a local minimizer. We use derivatives of a function $f : \mathbb{R}^n \to \mathbb{R}$. Recall that the first-order derivative of f, denoted Df, is

$$Df \triangleq \left[\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \ldots, \frac{\partial f}{\partial x_n}\right].$$

Note that the gradient ∇f is just the transpose of Df; that is, $\nabla f = (Df)^\top$. The second derivative of $f : \mathbb{R}^n \to \mathbb{R}$ (also called the *Hessian* of f) is

$$\boldsymbol{F}(\boldsymbol{x}) \triangleq D^2 f(\boldsymbol{x}) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2}(\boldsymbol{x}) & \cdots & \frac{\partial^2 f}{\partial x_n \partial x_1}(\boldsymbol{x}) \\ \vdots & & \vdots \\ \frac{\partial^2 f}{\partial x_1 \partial x_n}(\boldsymbol{x}) & \cdots & \frac{\partial^2 f}{\partial x_n^2}(\boldsymbol{x}) \end{bmatrix}.$$

Example 6.1 Let $f(x_1, x_2) = 5x_1 + 8x_2 + x_1x_2 - x_1^2 - 2x_2^2$. Then,

$$Df(\boldsymbol{x}) = (\nabla f(\boldsymbol{x}))^\top = \left[\frac{\partial f}{\partial x_1}(\boldsymbol{x}), \frac{\partial f}{\partial x_2}(\boldsymbol{x})\right] = [5 + x_2 - 2x_1, 8 + x_1 - 4x_2]$$

and

$$\boldsymbol{F}(\boldsymbol{x}) = D^2 f(\boldsymbol{x}) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2}(\boldsymbol{x}) & \frac{\partial^2 f}{\partial x_2 \partial x_1}(\boldsymbol{x}) \\ \frac{\partial^2 f}{\partial x_1 \partial x_2}(\boldsymbol{x}) & \frac{\partial^2 f}{\partial x_2^2}(\boldsymbol{x}) \end{bmatrix} = \begin{bmatrix} -2 & 1 \\ 1 & -4 \end{bmatrix}.$$

∎

Given an optimization problem with constraint set Ω, a minimizer may lie either in the interior or on the boundary of Ω. To study the case where it lies on the boundary, we need the notion of *feasible directions.*

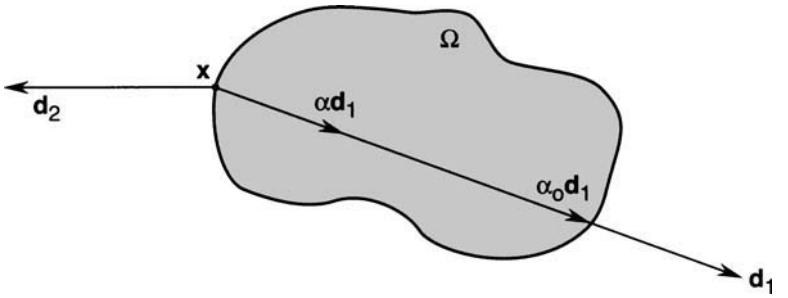

Figure 6.2 Two-dimensional illustration of feasible directions; $\boldsymbol{d}_1$ is a feasible direction, $\boldsymbol{d}_2$ is not a feasible direction.

Definition 6.2 A vector $\boldsymbol{d} \in \mathbb{R}^n$, $\boldsymbol{d} \neq \boldsymbol{0}$, is a *feasible direction* at $\boldsymbol{x} \in \Omega$ if there exists $\alpha_0 > 0$ such that $\boldsymbol{x} + \alpha \boldsymbol{d} \in \Omega$ for all $\alpha \in [0, \alpha_0]$. ∎

Figure 6.2 illustrates the notion of feasible directions.

Let $f : \mathbb{R}^n \to \mathbb{R}$ be a real-valued function and let $\boldsymbol{d}$ be a feasible direction at $\boldsymbol{x} \in \Omega$. The *directional derivative of f in the direction $\boldsymbol{d}$*, denoted $\partial f / \partial \boldsymbol{d}$, is the real-valued function defined by

$$\frac{\partial f}{\partial \boldsymbol{d}}(\boldsymbol{x}) = \lim_{\alpha \to 0} \frac{f(\boldsymbol{x} + \alpha \boldsymbol{d}) - f(\boldsymbol{x})}{\alpha}.$$

If $\|\boldsymbol{d}\| = 1$, then $\partial f / \partial \boldsymbol{d}$ is the rate of increase of f at $\boldsymbol{x}$ in the direction $\boldsymbol{d}$. To compute the directional derivative above, suppose that $\boldsymbol{x}$ and $\boldsymbol{d}$ are given. Then, $f(\boldsymbol{x} + \alpha \boldsymbol{d})$ is a function of α, and

$$\frac{\partial f}{\partial \boldsymbol{d}}(\boldsymbol{x}) = \frac{d}{d\alpha} f(\boldsymbol{x} + \alpha \boldsymbol{d}) \bigg|_{\alpha = 0}.$$

Applying the chain rule yields

$$\frac{\partial f}{\partial \boldsymbol{d}}(\boldsymbol{x}) = \frac{d}{d\alpha} f(\boldsymbol{x} + \alpha \boldsymbol{d}) \bigg|_{\alpha = 0} = \nabla f(\boldsymbol{x})^\top \boldsymbol{d} = \langle \nabla f(\boldsymbol{x}), \boldsymbol{d} \rangle = \boldsymbol{d}^\top \nabla f(\boldsymbol{x}).$$

In summary, if $\boldsymbol{d}$ is a unit vector ($\|\boldsymbol{d}\| = 1$), then $\langle \nabla f(\boldsymbol{x}), \boldsymbol{d} \rangle$ is the rate of increase of f at the point $\boldsymbol{x}$ in the direction $\boldsymbol{d}$.

Example 6.2 Define $f : \mathbb{R}^3 \to \mathbb{R}$ by $f(\boldsymbol{x}) = x_1 x_2 x_3$, and let

$$\boldsymbol{d} = \left[\frac{1}{2}, \frac{1}{2}, \frac{1}{\sqrt{2}} \right]^\top.$$

The directional derivative of f in the direction $\boldsymbol{d}$ is

$$\frac{\partial f}{\partial \boldsymbol{d}}(\boldsymbol{x}) = \nabla f(\boldsymbol{x})^\top \boldsymbol{d} = [x_2 x_3, x_1 x_3, x_1 x_2] \begin{bmatrix} 1/2 \\ 1/2 \\ 1/\sqrt{2} \end{bmatrix} = \frac{x_2 x_3 + x_1 x_3 + \sqrt{2} x_1 x_2}{2}.$$

Note that because $\|\boldsymbol{d}\| = 1$, the above is also the rate of increase of f at $\boldsymbol{x}$ in the direction $\boldsymbol{d}$. ∎

We are now ready to state and prove the following theorem.

Theorem 6.1 ***First-Order Necessary Condition (FONC).*** *Let Ω be a subset of $\mathbb{R}^n$ and $f \in C^1$ a real-valued function on Ω. If $\boldsymbol{x}^*$ is a local minimizer of f over Ω, then for any feasible direction $\boldsymbol{d}$ at $\boldsymbol{x}^*$, we have*

$$\boldsymbol{d}^\top \nabla f(\boldsymbol{x}^*) \geq 0.$$

□

Proof. Define

$$\boldsymbol{x}(\alpha) = \boldsymbol{x}^* + \alpha \boldsymbol{d} \in \Omega.$$

Note that $\boldsymbol{x}(0) = \boldsymbol{x}^*$. Define the composite function

$$\phi(\alpha) = f(\boldsymbol{x}(\alpha)).$$

Then, by Taylor's theorem,

$$f(\boldsymbol{x}^* + \alpha \boldsymbol{d}) - f(\boldsymbol{x}^*) = \phi(\alpha) - \phi(0) = \phi'(0)\alpha + o(\alpha) = \alpha \boldsymbol{d}^\top \nabla f(\boldsymbol{x}(0)) + o(\alpha),$$

where $\alpha \geq 0$ [recall the definition of $o(\alpha)$ ("little-oh of α") in Part I]. Thus, if $\phi(\alpha) \geq \phi(0)$, that is, $f(\boldsymbol{x}^* + \alpha \boldsymbol{d}) \geq f(\boldsymbol{x}^*)$ for sufficiently small values of $\alpha > 0$ ($\boldsymbol{x}^*$ is a local minimizer), then we have to have $\boldsymbol{d}^\top \nabla f(\boldsymbol{x}^*) \geq 0$ (see Exercise 5.8). ∎

Theorem 6.1 is illustrated in Figure 6.3.

An alternative way to express the FONC is

$$\frac{\partial f}{\partial \boldsymbol{d}}(\boldsymbol{x}^*) \geq 0$$

for all feasible directions $\boldsymbol{d}$. In other words, if $\boldsymbol{x}^*$ is a local minimizer, then the rate of increase of f at $\boldsymbol{x}^*$ in any feasible direction $\boldsymbol{d}$ in Ω is nonnegative. Using directional derivatives, an alternative proof of Theorem 6.1 is as follows. Suppose that $\boldsymbol{x}^*$ is a local minimizer. Then, for any feasible direction $\boldsymbol{d}$, there exists $\bar{\alpha} > 0$ such that for all $\alpha \in (0, \bar{\alpha})$,

$$f(\boldsymbol{x}^*) \leq f(\boldsymbol{x}^* + \alpha \boldsymbol{d}).$$

Hence, for all $\alpha \in (0, \bar{\alpha})$, we have

$$\frac{f(\boldsymbol{x}^* + \alpha \boldsymbol{d}) - f(\boldsymbol{x}^*)}{\alpha} \geq 0.$$

Taking the limit as $\alpha \to 0$, we conclude that

$$\frac{\partial f}{\partial \boldsymbol{d}}(\boldsymbol{x}^*) \geq 0.$$

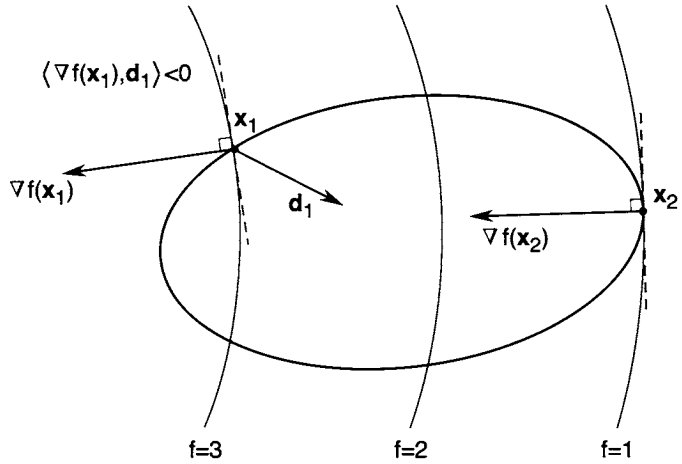

Figure 6.3 Illustration of the FONC for a constrained case; $\boldsymbol{x}_1$ does not satisfy the FONC, whereas $\boldsymbol{x}_2$ satisfies the FONC.

A special case of interest is when $\boldsymbol{x}^*$ is an interior point of Ω (see Section 4.4). In this case, any direction is feasible, and we have the following result.

Corollary 6.1 ***Interior Case.*** *Let Ω be a subset of $\mathbb{R}^n$ and $f \in \mathcal{C}^1$ a real-valued function on Ω. If $\boldsymbol{x}^*$ is a local minimizer of f over Ω and if $\boldsymbol{x}^*$ is an interior point of Ω, then*

$$\nabla f(\boldsymbol{x}^*) = \mathbf{0}.$$

□

Proof. Suppose that f has a local minimizer $\boldsymbol{x}^*$ that is an interior point of Ω. Because $\boldsymbol{x}^*$ is an interior point of Ω, the set of feasible directions at $\boldsymbol{x}^*$ is the whole of $\mathbb{R}^n$. Thus, for any $\boldsymbol{d} \in \mathbb{R}^n$, $\boldsymbol{d}^\top \nabla f(\boldsymbol{x}^*) \geq 0$ and $-\boldsymbol{d}^\top \nabla f(\boldsymbol{x}^*) \geq 0$. Hence, $\boldsymbol{d}^\top \nabla f(\boldsymbol{x}^*) = 0$ for all $\boldsymbol{d} \in \mathbb{R}^n$, which implies that $\nabla f(\boldsymbol{x}^*) = \mathbf{0}$. ∎

Example 6.3 Consider the problem

$$\begin{aligned} \text{minimize} \quad & x_1^2 + 0.5x_2^2 + 3x_2 + 4.5 \\ \text{subject to} \quad & x_1, x_2 \geq 0. \end{aligned}$$

a. Is the first-order necessary condition (FONC) for a local minimizer satisfied at $\boldsymbol{x} = [1, 3]^\top$?

b. Is the FONC for a local minimizer satisfied at $\boldsymbol{x} = [0, 3]^\top$?

c. Is the FONC for a local minimizer satisfied at $\boldsymbol{x} = [1, 0]^\top$?

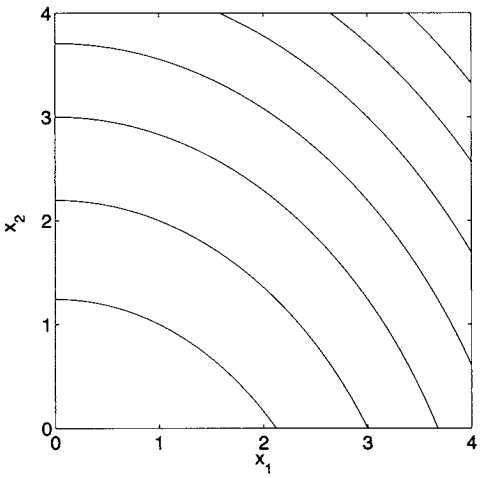

Figure 6.4 Level sets of the function in Example 6.3.

d. Is the FONC for a local minimizer satisfied at $\boldsymbol{x} = [0, 0]^\top$?

Solution: First, let $f : \mathbb{R}^2 \to \mathbb{R}$ be defined by $f(\boldsymbol{x}) = x_1^2 + 0.5x_2^2 + 3x_2 + 4.5$, where $\boldsymbol{x} = [x_1, x_2]^\top$. A plot of the level sets of f is shown in Figure 6.4.

a. At $\boldsymbol{x} = [1, 3]^\top$, we have $\nabla f(\boldsymbol{x}) = [2x_1, x_2 + 3]^\top = [2, 6]^\top$. The point $\boldsymbol{x} = [1, 3]^\top$ is an interior point of $\Omega = \{\boldsymbol{x} : x_1 \geq 0, x_2 \geq 0\}$. Hence, the FONC requires that $\nabla f(\boldsymbol{x}) = \boldsymbol{0}$. The point $\boldsymbol{x} = [1, 3]^\top$ does not satisfy the FONC for a local minimizer.

b. At $\boldsymbol{x} = [0, 3]^\top$, we have $\nabla f(\boldsymbol{x}) = [0, 6]^\top$, and hence $\boldsymbol{d}^\top \nabla f(\boldsymbol{x}) = 6d_2$, where $\boldsymbol{d} = [d_1, d_2]^\top$. For $\boldsymbol{d}$ to be feasible at $\boldsymbol{x}$, we need $d_1 \geq 0$, and d_2 can take an arbitrary value in $\mathbb{R}$. The point $\boldsymbol{x} = [0, 3]^\top$ does not satisfy the FONC for a minimizer because d_2 is allowed to be less than zero. For example, $\boldsymbol{d} = [1, -1]^\top$ is a feasible direction, but $\boldsymbol{d}^\top \nabla f(\boldsymbol{x}) = -6 < 0$.

c. At $\boldsymbol{x} = [1, 0]^\top$, we have $\nabla f(\boldsymbol{x}) = [2, 3]^\top$, and hence $\boldsymbol{d}^\top \nabla f(\boldsymbol{x}) = 2d_1 + 3d_2$. For $\boldsymbol{d}$ to be feasible, we need $d_2 \geq 0$, and d_1 can take an arbitrary value in $\mathbb{R}$. For example, $\boldsymbol{d} = [-5, 1]^\top$ is a feasible direction. But $\boldsymbol{d}^\top \nabla f(\boldsymbol{x}) = -7 < 0$. Thus, $\boldsymbol{x} = [1, 0]^\top$ does not satisfy the FONC for a local minimizer.

d. At $\boldsymbol{x} = [0, 0]^\top$, we have $\nabla f(\boldsymbol{x}) = [0, 3]^\top$, and hence $\boldsymbol{d}^\top \nabla f(\boldsymbol{x}) = 3d_2$. For $\boldsymbol{d}$ to be feasible, we need $d_2 \geq 0$ and $d_1 \geq 0$. Hence, $\boldsymbol{x} = [0, 0]^\top$ satisfies the FONC for a local minimizer. ∎

Example 6.4 Figure 6.5 shows a simplified model of a cellular wireless system (the distances shown have been scaled down to make the calculations

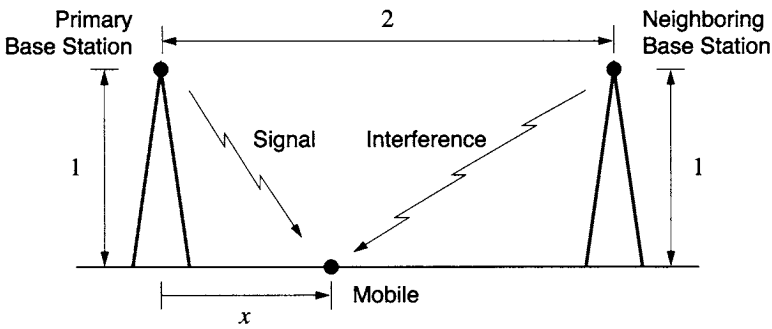

Figure 6.5 Simplified cellular wireless system in Example 6.4.

simpler). A mobile user (also called a *mobile*) is located at position x (see Figure 6.5).

There are two base station antennas, one for the primary base station and another for the neighboring base station. Both antennas are transmitting signals to the mobile user, at equal power. However, the power of the received signal as measured by the mobile is the reciprocal of the squared distance from the associated antenna (primary or neighboring base station). We are interested in finding the position of the mobile that maximizes the *signal-to-interference ratio*, which is the ratio of the signal power received from the primary base station to the signal power received from the neighboring base station.

We use the FONC to solve this problem. The squared distance from the mobile to the primary antenna is $1 + x^2$, while the squared distance from the mobile to the neighboring antenna is $1 + (2 - x)^2$. Therefore, the signal-to-interference ratio is

$$f(x) = \frac{1 + (2 - x)^2}{1 + x^2}.$$

We have

$$\begin{aligned} f'(x) &= \frac{-2(2 - x)(1 + x^2) - 2x(1 + (2 - x)^2)}{(1 + x^2)^2} \\ &= \frac{4(x^2 - 2x - 1)}{(1 + x^2)^2}. \end{aligned}$$

By the FONC, at the optimal position x^* we have $f'(x^*) = 0$. Hence, either $x^* = 1 - \sqrt{2}$ or $x^* = 1 + \sqrt{2}$. Evaluating the objective function at these two candidate points, it easy to see that $x^* = 1 - \sqrt{2}$ is the optimal position. ∎

The next example illustrates that in some problems the FONC is not helpful for eliminating candidate local minimizers. However, in such cases, there may be a recasting of the problem into an equivalent form that makes the FONC useful.

Example 6.5 Consider the set-constrained problem

$$\begin{aligned} &\text{minimize} && f(\boldsymbol{x}) \\ &\text{subject to} && \boldsymbol{x} \in \Omega, \end{aligned}$$

where $\Omega = \{[x_1, x_2]^\top : x_1^2 + x_2^2 = 1\}$.

a. Consider a point $\boldsymbol{x}^* \in \Omega$. Specify all feasible directions at $\boldsymbol{x}^*$.

b. Which points in Ω satisfy the FONC for this set-constrained problem?

c. Based on part b, is the FONC for this set-constrained problem useful for eliminating local-minimizer candidates?

d. Suppose that we use polar coordinates to parameterize points $\boldsymbol{x} \in \Omega$ in terms of a single parameter θ:

$$x_1 = \cos\theta \qquad x_2 = \sin\theta.$$

Now use the FONC for unconstrained problems (with respect to θ) to derive a necessary condition of this sort: If $\boldsymbol{x}^* \in \Omega$ is a local minimizer, then $\boldsymbol{d}^\top \nabla f(\boldsymbol{x}^*) = 0$ for all $\boldsymbol{d}$ satisfying a "certain condition." Specify what this certain condition is.

Solution:

a. There are no feasible directions at any $\boldsymbol{x}^*$.

b. Because of part a, *all* points in Ω satisfy the FONC for this set-constrained problem.

c. No, the FONC for this set-constrained problem is not useful for eliminating local-minimizer candidates.

d. Write $h(\theta) = f(g(\theta))$, where $g : \mathbb{R} \to \mathbb{R}^2$ is given by the equations relating θ to $\boldsymbol{x} = [x_1, x_2]^\top$. Note that $Dg(\theta) = [-\sin\theta, \cos\theta]^\top$. Hence, by the chain rule,

$$h'(\theta) = Df(g(\theta))Dg(\theta) = Dg(\theta)^\top \nabla f(g(\theta)).$$

Notice that $Dg(\theta)$ is tangent to Ω at $\boldsymbol{x} = g(\theta)$. Alternatively, we could say that $Dg(\theta)$ is orthogonal to $\boldsymbol{x} = g(\theta)$.

Suppose that $\boldsymbol{x}^* \in \Omega$ is a local minimizer. Write $\boldsymbol{x}^* = g(\theta^*)$. Then θ^* is an unconstrained minimizer of h. By the FONC for unconstrained problems, $h'(\theta^*) = 0$, which implies that $\boldsymbol{d}^\top \nabla f(\boldsymbol{x}^*) = 0$ for all $\boldsymbol{d}$ tangent to Ω at $\boldsymbol{x}^*$ (or, alternatively, for all $\boldsymbol{d}$ orthogonal to $\boldsymbol{x}^*$). ∎

We now derive a second-order necessary condition that is satisfied by a local minimizer.

Theorem 6.2 ***Second-Order Necessary Condition (SONC).*** *Let $\Omega \subset \mathbb{R}^n$, $f \in C^2$ a function on Ω, $\boldsymbol{x}^*$ a local minimizer of f over Ω, and $\boldsymbol{d}$ a feasible direction at $\boldsymbol{x}^*$. If $\boldsymbol{d}^\top \nabla f(\boldsymbol{x}^*) = 0$, then*

$$\boldsymbol{d}^\top \boldsymbol{F}(\boldsymbol{x}^*)\boldsymbol{d} \geq 0,$$

where $\boldsymbol{F}$ is the Hessian of f. □

Proof. We prove the result by contradiction. Suppose that there is a feasible direction $\boldsymbol{d}$ at $\boldsymbol{x}^*$ such that $\boldsymbol{d}^\top \nabla f(\boldsymbol{x}^*) = 0$ and $\boldsymbol{d}^\top \boldsymbol{F}(\boldsymbol{x}^*)\boldsymbol{d} < 0$. Let $\boldsymbol{x}(\alpha) = \boldsymbol{x}^* + \alpha \boldsymbol{d}$ and define the composite function $\phi(\alpha) = f(\boldsymbol{x}^* + \alpha \boldsymbol{d}) = f(\boldsymbol{x}(\alpha))$. Then, by Taylor's theorem,

$$\phi(\alpha) = \phi(0) + \phi''(0)\frac{\alpha^2}{2} + o(\alpha^2),$$

where by assumption, $\phi'(0) = \boldsymbol{d}^\top \nabla f(\boldsymbol{x}^*) = 0$ and $\phi''(0) = \boldsymbol{d}^\top \boldsymbol{F}(\boldsymbol{x}^*)\boldsymbol{d} < 0$. For sufficiently small α,

$$\phi(\alpha) - \phi(0) = \phi''(0)\frac{\alpha^2}{2} + o(\alpha^2) < 0,$$

that is,

$$f(\boldsymbol{x}^* + \alpha \boldsymbol{d}) < f(\boldsymbol{x}^*),$$

which contradicts the assumption that $\boldsymbol{x}^*$ is a local minimizer. Thus,

$$\phi''(0) = \boldsymbol{d}^\top \boldsymbol{F}(\boldsymbol{x}^*)\boldsymbol{d} \geq 0.$$

∎

Corollary 6.2 ***Interior Case.*** *Let $\boldsymbol{x}^*$ be an interior point of $\Omega \subset \mathbb{R}^n$. If $\boldsymbol{x}^*$ is a local minimizer of $f : \Omega \to \mathbb{R}$, $f \in C^2$, then*

$$\nabla f(\boldsymbol{x}^*) = \boldsymbol{0},$$

and $\boldsymbol{F}(\boldsymbol{x}^*)$ is positive semidefinite ($\boldsymbol{F}(\boldsymbol{x}^*) \geq 0$); that is, for all $\boldsymbol{d} \in \mathbb{R}^n$,

$$\boldsymbol{d}^\top \boldsymbol{F}(\boldsymbol{x}^*)\boldsymbol{d} \geq 0.$$

□

Proof. If $\boldsymbol{x}^*$ is an interior point, then all directions are feasible. The result then follows from Corollary 6.1 and Theorem 6.2. ∎

In the examples below, we show that the necessary conditions are *not* sufficient.

Example 6.6 Consider a function of one variable $f(x) = x^3$, $f : \mathbb{R} \to \mathbb{R}$. Because $f'(0) = 0$, and $f''(0) = 0$, the point $x = 0$ satisfies both the FONC and SONC. However, $x = 0$ is not a minimizer (see Figure 6.6). ∎

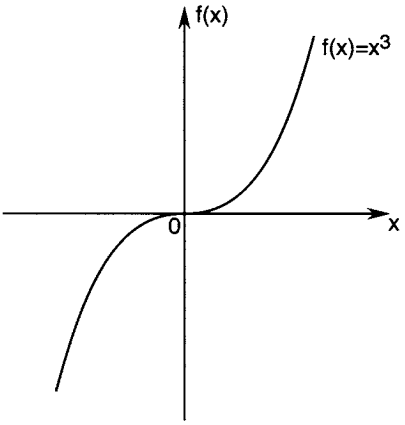

Figure 6.6 The point 0 satisfies the FONC and SONC but is not a minimizer.

Example 6.7 Consider a function $f : \mathbb{R}^2 \to \mathbb{R}$, where $f(\boldsymbol{x}) = x_1^2 - x_2^2$. The FONC requires that $\nabla f(\boldsymbol{x}) = [2x_1, -2x_2]^\top = \mathbf{0}$. Thus, $\boldsymbol{x} = [0,0]^\top$ satisfies the FONC. The Hessian matrix of f is

$$\boldsymbol{F}(\boldsymbol{x}) = \begin{bmatrix} 2 & 0 \\ 0 & -2 \end{bmatrix}.$$

The Hessian matrix is indefinite; that is, for some $\boldsymbol{d}_1 \in \mathbb{R}^2$ we have $\boldsymbol{d}_1^\top \boldsymbol{F}\boldsymbol{d}_1 > 0$ (e.g., $\boldsymbol{d}_1 = [1,0]^\top$) and for some $\boldsymbol{d}_2$ we have $\boldsymbol{d}_2^\top \boldsymbol{F}\boldsymbol{d}_2 < 0$ (e.g., $\boldsymbol{d}_2 = [0,1]^\top$). Thus, $\boldsymbol{x} = [0,0]^\top$ does not satisfy the SONC, and hence it is not a minimizer. The graph of $f(\boldsymbol{x}) = x_1^2 - x_2^2$ is shown in Figure 6.7. ∎

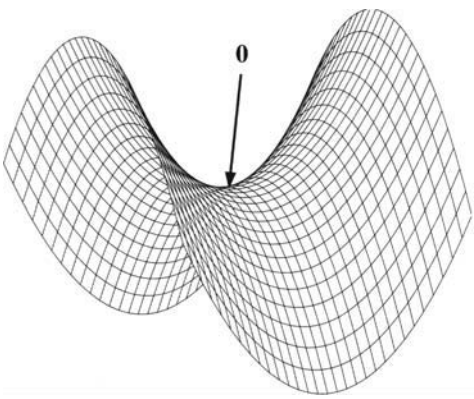

Figure 6.7 Graph of $f(\boldsymbol{x}) = x_1^2 - x_2^2$. The point $\mathbf{0}$ satisfies the FONC but not SONC; this point is not a minimizer.

We now derive sufficient conditions that imply that $\boldsymbol{x}^*$ is a local minimizer.

Theorem 6.3 ***Second-Order Sufficient Condition (SOSC), Interior Case.*** *Let $f \in \mathcal{C}^2$ be defined on a region in which $\boldsymbol{x}^*$ is an interior point. Suppose that*

1. $\nabla f(\boldsymbol{x}^*) = \boldsymbol{0}$.
2. $\boldsymbol{F}(\boldsymbol{x}^*) > 0$.

Then, $\boldsymbol{x}^*$ is a strict local minimizer of f. □

Proof. Because $f \in \mathcal{C}^2$, we have $\boldsymbol{F}(\boldsymbol{x}^*) = \boldsymbol{F}^\top(\boldsymbol{x}^*)$. Using assumption 2 and Rayleigh's inequality it follows that if $\boldsymbol{d} \neq \boldsymbol{0}$, then $0 < \lambda_{\min}(\boldsymbol{F}(\boldsymbol{x}^*))\|\boldsymbol{d}\|^2 \leq \boldsymbol{d}^\top \boldsymbol{F}(\boldsymbol{x}^*)\boldsymbol{d}$. By Taylor's theorem and assumption 1,

$$f(\boldsymbol{x}^* + \boldsymbol{d}) - f(\boldsymbol{x}^*) = \frac{1}{2}\boldsymbol{d}^\top \boldsymbol{F}(\boldsymbol{x}^*)\boldsymbol{d} + o(\|\boldsymbol{d}\|^2) \geq \frac{\lambda_{\min}(\boldsymbol{F}(\boldsymbol{x}^*))}{2}\|\boldsymbol{d}\|^2 + o(\|\boldsymbol{d}\|^2).$$

Hence, for all $\boldsymbol{d}$ such that $\|\boldsymbol{d}\|$ is sufficiently small,

$$f(\boldsymbol{x}^* + \boldsymbol{d}) > f(\boldsymbol{x}^*),$$

which completes the proof. ∎

Example 6.8 Let $f(\boldsymbol{x}) = x_1^2 + x_2^2$. We have $\nabla f(\boldsymbol{x}) = [2x_1, 2x_2]^\top = \boldsymbol{0}$ if and only if $\boldsymbol{x} = [0, 0]^\top$. For all $\boldsymbol{x} \in \mathbb{R}^2$, we have

$$\boldsymbol{F}(\boldsymbol{x}) = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} > 0.$$

The point $\boldsymbol{x} = [0, 0]^\top$ satisfies the FONC, SONC, and SOSC. It is a strict local minimizer. Actually, $\boldsymbol{x} = [0, 0]^\top$ is a strict global minimizer. Figure 6.8 shows the graph of $f(\boldsymbol{x}) = x_1^2 + x_2^2$. ∎

In this chapter we presented a theoretical basis for the solution of nonlinear unconstrained problems. In the following chapters we are concerned with iterative methods of solving such problems. Such methods are of great importance in practice. Indeed, suppose that one is confronted with a highly nonlinear function of 20 variables. Then, the FONC requires the solution of 20 nonlinear simultaneous equations for 20 variables. These equations, being nonlinear, will normally have multiple solutions. In addition, we would have to compute 210 second derivatives (provided that $f \in \mathcal{C}^2$) to use the SONC or SOSC. We begin our discussion of iterative methods in the next chapter with search methods for functions of one variable.

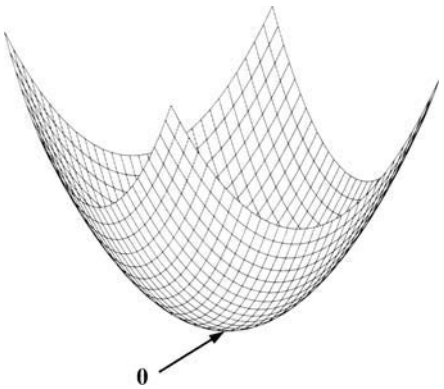

Figure 6.8 Graph of $f(\boldsymbol{x}) = x_1^2 + x_2^2$.

EXERCISES

6.1 Consider the problem

$$\begin{aligned} &\text{minimize} && f(\boldsymbol{x}) \\ &\text{subject to} && \boldsymbol{x} \in \Omega, \end{aligned}$$

where $f \in \mathcal{C}^2$. For each of the following specifications for Ω, $\boldsymbol{x}^*$, and f, determine if the given point $\boldsymbol{x}^*$ is: (i) definitely a local minimizer; (ii) definitely not a local minimizer; or (iii) possibly a local minimizer.

a. $f : \mathbb{R}^2 \to \mathbb{R}$, $\Omega = \{\boldsymbol{x} = [x_1, x_2]^\top : x_1 \geq 1\}$, $\boldsymbol{x}^* = [1, 2]^\top$, and gradient $\nabla f(\boldsymbol{x}^*) = [1, 1]^\top$.

b. $f : \mathbb{R}^2 \to \mathbb{R}$, $\Omega = \{\boldsymbol{x} = [x_1, x_2]^\top : x_1 \geq 1, x_2 \geq 2\}$, $\boldsymbol{x}^* = [1, 2]^\top$, and gradient $\nabla f(\boldsymbol{x}^*) = [1, 0]^\top$.

c. $f : \mathbb{R}^2 \to \mathbb{R}$, $\Omega = \{\boldsymbol{x} = [x_1, x_2]^\top : x_1 \geq 0, x_2 \geq 0\}$, $\boldsymbol{x}^* = [1, 2]^\top$, gradient $\nabla f(\boldsymbol{x}^*) = [0, 0]^\top$, and Hessian $\boldsymbol{F}(\boldsymbol{x}^*) = \boldsymbol{I}$ (identity matrix).

d. $f : \mathbb{R}^2 \to \mathbb{R}$, $\Omega = \{\boldsymbol{x} = [x_1, x_2]^\top : x_1 \geq 1, x_2 \geq 2\}$, $\boldsymbol{x}^* = [1, 2]^\top$, gradient $\nabla f(\boldsymbol{x}^*) = [1, 0]^\top$, and Hessian

$$\boldsymbol{F}(\boldsymbol{x}^*) = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}.$$

6.2 Find minimizers and maximizers of the function

$$f(x_1, x_2) = \frac{1}{3}x_1^3 - 4x_1 + \frac{1}{3}x_2^3 - 16x_2.$$

6.3 Show that if $\boldsymbol{x}^*$ is a global minimizer of f over Ω, and $\boldsymbol{x}^* \in \Omega' \subset \Omega$, then $\boldsymbol{x}^*$ is a global minimizer of f over Ω'.

6.4 Suppose that $\boldsymbol{x}^*$ is a local minimizer of f over Ω, and $\Omega \subset \Omega'$. Show that if $\boldsymbol{x}^*$ is an interior point of Ω, then $\boldsymbol{x}^*$ is a local minimizer of f over Ω'. Show that the same conclusion cannot be made if $\boldsymbol{x}^*$ is not an interior point of Ω.

6.5 Consider the problem of minimizing $f : \mathbb{R} \to \mathbb{R}$, $f \in C^3$, over the constraint set Ω. Suppose that 0 is an *interior point* of Ω.

a. Suppose that 0 is a local minimizer. By the FONC we know that $f'(0) = 0$ (where f' is the first derivative of f). By the SONC we know that $f''(0) \geq 0$ (where f'' is the second derivative of f). State and prove a *third-order necessary condition* (*TONC*) involving the third derivative at 0, $f'''(0)$.

b. Give an example of f such that the FONC, SONC, and TONC (in part a) hold at the interior point 0, but 0 is not a local minimizer of f over Ω. (Show that your example is correct.)

c. Suppose that f is a third-order polynomial. If 0 satisfies the FONC, SONC, and TONC (in part a), then is this *sufficient* for 0 to be a local minimizer?

6.6 Consider the problem of minimizing $f : \mathbb{R} \to \mathbb{R}$, $f \in C^3$, over the constraint set $\Omega = [0, 1]$. Suppose that $x^* = 0$ is a local minimizer.

a. By the FONC we know that $f'(0) \geq 0$ (where f' is the first derivative of f). By the SONC we know that if $f'(0) = 0$, then $f''(0) \geq 0$ (where f'' is the second derivative of f). State and prove a *third-order necessary condition* involving the third derivative at 0, $f'''(0)$.

b. Give an example of f such that the FONC, SONC, and TONC (in part a) hold at the point 0, but 0 is not a local minimizer of f over $\Omega = [0, 1]$.

6.7 Let $f : \mathbb{R}^n \to \mathbb{R}$, $\boldsymbol{x}_0 \in \mathbb{R}^n$, and $\Omega \subset \mathbb{R}^n$. Show that

$$\boldsymbol{x}_0 + \arg\min_{\boldsymbol{x} \in \Omega} f(\boldsymbol{x}) = \arg\min_{\boldsymbol{y} \in \Omega'} f(\boldsymbol{y}),$$

where $\Omega' = \{\boldsymbol{y} : \boldsymbol{y} - \boldsymbol{x}_0 \in \Omega\}$.

6.8 Consider the following function $f : \mathbb{R}^2 \to \mathbb{R}$:

$$f(\boldsymbol{x}) = \boldsymbol{x}^\top \begin{bmatrix} 1 & 2 \\ 4 & 7 \end{bmatrix} \boldsymbol{x} + \boldsymbol{x}^\top \begin{bmatrix} 3 \\ 5 \end{bmatrix} + 6.$$

a. Find the gradient and Hessian of f at the point $[1,1]^\top$.

b. Find the directional derivative of f at $[1,1]^\top$ with respect to a unit vector in the direction of maximal rate of increase.

c. Find a point that satisfies the FONC (interior case) for f. Does this point satisfy the SONC (for a minimizer)?

6.9 Consider the following function:

$$f(x_1, x_2) = x_1^2 x_2 + x_2^3 x_1.$$

a. In what direction does the function f *decrease* most rapidly at the point $\boldsymbol{x}^{(0)} = [2,1]^\top$?

b. What is the rate of increase of f at the point $\boldsymbol{x}^{(0)}$ in the direction of maximum decrease of f?

c. Find the rate of increase of f at the point $\boldsymbol{x}^{(0)}$ in the direction $\boldsymbol{d} = [3,4]^\top$.

6.10 Consider the following function $f : \mathbb{R}^2 \to \mathbb{R}$:

$$f(\boldsymbol{x}) = \boldsymbol{x}^\top \begin{bmatrix} 2 & 5 \\ -1 & 1 \end{bmatrix} \boldsymbol{x} + \boldsymbol{x}^\top \begin{bmatrix} 3 \\ 4 \end{bmatrix} + 7.$$

a. Find the directional derivative of f at $[0,1]^\top$ in the direction $[1,0]^\top$.

b. Find all points that satisfy the first-order necessary condition for f.

Does f have a minimizer? If it does, then find all minimizer(s); otherwise, explain why it does not.

6.11 Consider the problem

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

where $x_1, x_2 \in \mathbb{R}$.

a. Does the point $[x_1, x_2]^\top = \mathbf{0}$ satisfy the first-order necessary condition for a minimizer? That is, if f is the objective function, is it true that $\boldsymbol{d}^\top \nabla f(\mathbf{0}) \geq 0$ for all feasible directions $\boldsymbol{d}$ at $\mathbf{0}$?

b. Is the point $[x_1, x_2]^\top = \mathbf{0}$ a local minimizer, a strict local minimizer, a local maximizer, a strict local maximizer, or none of the above?

6.12 Consider the problem

$$\begin{aligned} \text{minimize} \quad & f(\boldsymbol{x}) \\ \text{subject to} \quad & \boldsymbol{x} \in \Omega, \end{aligned}$$

where $f : \mathbb{R}^2 \to \mathbb{R}$ is given by $f(\boldsymbol{x}) = 5x_2$ with $\boldsymbol{x} = [x_1, x_2]^\top$, and $\Omega = \{\boldsymbol{x} = [x_1, x_2]^\top : x_1^2 + x_2 \geq 1\}$.

a. Does the point $\boldsymbol{x}^* = [0, 1]^\top$ satisfy the first-order necessary condition?

b. Does the point $\boldsymbol{x}^* = [0, 1]^\top$ satisfy the second-order necessary condition?

c. Is the point $\boldsymbol{x}^* = [0, 1]^\top$ a local minimizer?

6.13 Consider the problem

$$\begin{aligned} \text{minimize} \quad & f(\boldsymbol{x}) \\ \text{subject to} \quad & \boldsymbol{x} \in \Omega, \end{aligned}$$

where $f : \mathbb{R}^2 \to \mathbb{R}$ is given by $f(\boldsymbol{x}) = -3x_1$ with $\boldsymbol{x} = [x_1, x_2]^\top$, and $\Omega = \{\boldsymbol{x} = [x_1, x_2]^\top : x_1 + x_2^2 \leq 2\}$. Answer each of the following questions, showing complete justification.

a. Does the point $\boldsymbol{x}^* = [2, 0]^\top$ satisfy the first-order necessary condition?

b. Does the point $\boldsymbol{x}^* = [2, 0]^\top$ satisfy the second-order necessary condition?

c. Is the point $\boldsymbol{x}^* = [2, 0]^\top$ a local minimizer?

6.14 Consider the problem

$$\begin{aligned} \text{minimize} \quad & f(\boldsymbol{x}) \\ \text{subject to} \quad & \boldsymbol{x} \in \Omega, \end{aligned}$$

where $\Omega = \{\boldsymbol{x} \in \mathbb{R}^2 : x_1^2 + x_2^2 \geq 1\}$ and $f(\boldsymbol{x}) = x_2$.

a. Find all point(s) satisfying the FONC.

b. Which of the point(s) in part a satisfy the SONC?

c. Which of the point(s) in part a are local minimizers?

6.15 Consider the problem

$$\begin{aligned} \text{minimize} \quad & f(\boldsymbol{x}) \\ \text{subject to} \quad & \boldsymbol{x} \in \Omega \end{aligned}$$

where $f : \mathbb{R}^2 \to \mathbb{R}$ is given by $f(\boldsymbol{x}) = 3x_1$ with $\boldsymbol{x} = [x_1, x_2]^\top$, and $\Omega = \{\boldsymbol{x} = [x_1, x_2]^\top : x_1 + x_2^2 \geq 2\}$. Answer each of the following questions, showing complete justification.

a. Does the point $\boldsymbol{x}^* = [2, 0]^\top$ satisfy the first-order necessary condition?

b. Does the point $\boldsymbol{x}^* = [2, 0]^\top$ satisfy the second-order necessary condition?

c. Is the point $\boldsymbol{x}^* = [2, 0]^\top$ a local minimizer?
Hint: Draw a picture with the constraint set and level sets of f.

6.16 Consider the problem

$$\begin{aligned} &\text{minimize} && f(\boldsymbol{x}) \\ &\text{subject to} && \boldsymbol{x} \in \Omega, \end{aligned}$$

where $\boldsymbol{x} = [x_1, x_2]^\top$, $f : \mathbb{R}^2 \to \mathbb{R}$ is given by $f(\boldsymbol{x}) = 4x_1^2 - x_2^2$, and $\Omega = \{\boldsymbol{x} : x_1^2 + 2x_1 - x_2 \geq 0, x_1 \geq 0, x_2 \geq 0\}$.

a. Does the point $\boldsymbol{x}^* = \mathbf{0} = [0, 0]^\top$ satisfy the first-order necessary condition?

b. Does the point $\boldsymbol{x}^* = \mathbf{0}$ satisfy the second-order necessary condition?

c. Is the point $\boldsymbol{x}^* = \mathbf{0}$ a local minimizer of the given problem?

6.17 Consider the problem

$$\begin{aligned} &\text{maximize} && f(\boldsymbol{x}) \\ &\text{subject to} && \boldsymbol{x} \in \Omega, \end{aligned}$$

where $\Omega \subset \{\boldsymbol{x} \in \mathbb{R}^2 : x_1 > 0, x_2 > 0\}$ and $f : \Omega \to \mathbb{R}$ is given by $f(\boldsymbol{x}) = \log(x_1) + \log(x_2)$ with $\boldsymbol{x} = [x_1, x_2]^\top$, where "log" represents natural logarithm. Suppose that $\boldsymbol{x}^*$ is an optimal solution. Answer each of the following questions, showing complete justification.

a. Is it possible that $\boldsymbol{x}^*$ is an interior point of Ω?

b. At what point(s) (if any) is the second-order necessary condition satisfied?

6.18 Suppose that we are given n real numbers, $x_1, \ldots, x_n$. Find the number $\bar{x} \in \mathbb{R}$ such that the sum of the squared difference between $\bar{x}$ and the numbers above is minimized (assuming that the solution $\bar{x}$ exists).

6.19 An art collector stands at a distance of x feet from the wall, where a piece of art (picture) of height a feet is hung, b feet above his eyes, as shown in

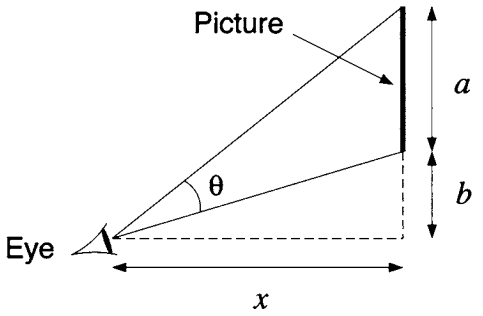

Figure 6.9 Art collector's eye position in Exercise 6.19.

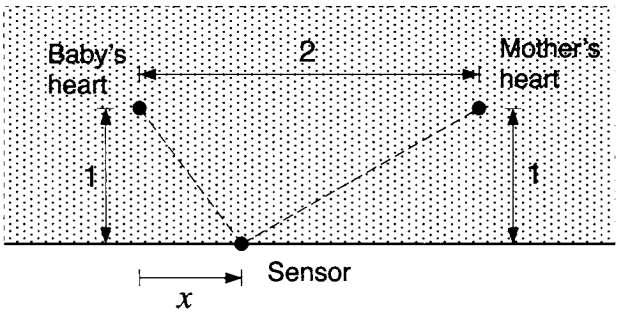

Figure 6.10 Simplified fetal heart monitoring system for Exercise 6.20.

Figure 6.9. Find the distance from the wall for which the angle θ subtended by the eye to the picture is maximized.
Hint: (1) Maximizing θ is equivalent to maximizing $\tan(\theta)$.
(2) If $\theta = \theta_2 - \theta_1$, then $\tan(\theta) = (\tan(\theta_2) - \tan(\theta_1))/(1 + \tan(\theta_2)\tan(\theta_1))$.

6.20 Figure 6.10 shows a simplified model of a fetal heart monitoring system (the distances shown have been scaled down to make the calculations simpler). A heartbeat sensor is located at position x (see Figure 6.10).

The energy of the heartbeat signal measured by the sensor is the reciprocal of the squared distance from the source (baby's heart or mother's heart). Find the position of the sensor that maximizes the *signal-to-interference ratio*, which is the ratio of the signal energy from the baby's heart to the signal energy from the mother's heart.

6.21 An amphibian vehicle needs to travel from point A (on land) to point B (in water), as illustrated in Figure 6.11. The speeds at which the vehicle travels on land and water are v_1 and v_2, respectively.

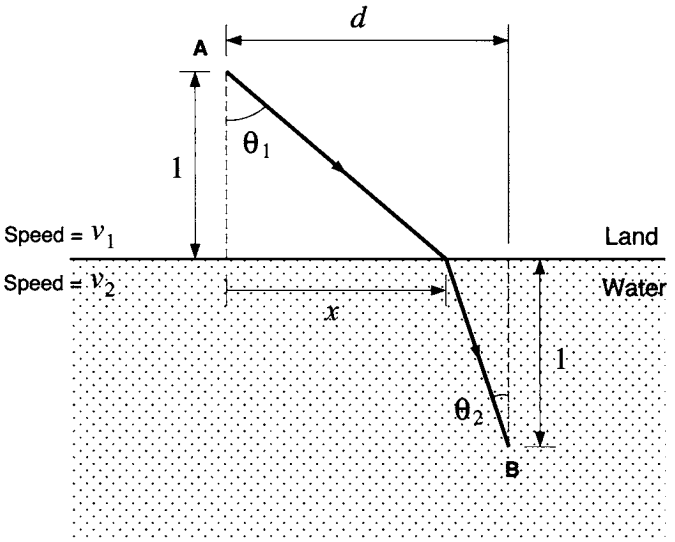

Figure 6.11 Path of amphibian vehicle in Exercise 6.21.

a. Suppose that the vehicle traverses a path that minimizes the total time taken to travel from A to B. Use the first-order necessary condition to show that for the optimal path above, the angles θ_1 and θ_2 in Figure 6.11 satisfy Snell's law:

$$\frac{\sin\theta_1}{\sin\theta_2} = \frac{v_1}{v_2}.$$

b. Does the minimizer for the problem in part a satisfy the second-order sufficient condition?

6.22 Suppose that you have a piece of land to sell and you have two buyers. If the first buyer receives a fraction x_1 of the piece of land, the buyer will pay you $U_1(x_1)$ dollars. Similarly, the second buyer will pay you $U_2(x_2)$ dollars for a fraction of x_2 of the land. Your goal is to sell parts of your land to the two buyers so that you maximize the total dollars you receive. (Other than the constraint that you can only sell whatever land you own, there are no restrictions on how much land you can sell to each buyer.)

a. Formulate the problem as an optimization problem of the kind

$$\begin{aligned} &\text{maximize} && f(\boldsymbol{x}) \\ &\text{subject to} && \boldsymbol{x} \in \Omega \end{aligned}$$

by specifying f and Ω. Draw a picture of the constraint set.

b. Suppose that $U_i(x_i) = a_i x_i$, $i = 1, 2$, where a_1 and a_2 are given positive constants such that $a_1 > a_2$. Find all feasible points that satisfy the first-order necessary condition, giving full justification.

c. Among those points in the answer of part b, find all that also satisfy the second-order necessary condition.

6.23 Let $f : \mathbb{R}^2 \to \mathbb{R}$ be defined by

$$f(\boldsymbol{x}) = (x_1 - x_2)^4 + x_1^2 - x_2^2 - 2x_1 + 2x_2 + 1,$$

where $\boldsymbol{x} = [x_1, x_2]^\top$. Suppose that we wish to minimize f over $\mathbb{R}^2$. Find all points satisfying the FONC. Do these points satisfy the SONC?

6.24 Show that if $\boldsymbol{d}$ is a feasible direction at a point $\boldsymbol{x} \in \Omega$, then for all $\beta > 0$, the vector $\beta\boldsymbol{d}$ is also a feasible direction at $\boldsymbol{x}$.

6.25 Let $\Omega = \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}\}$. Show that $\boldsymbol{d} \in \mathbb{R}^n$ is a feasible direction at $\boldsymbol{x} \in \Omega$ if and only if $\boldsymbol{A}\boldsymbol{d} = \boldsymbol{0}$.

6.26 Let $f : \mathbb{R}^2 \to \mathbb{R}$. Consider the problem

$$\begin{aligned} \text{minimize} \quad & f(\boldsymbol{x}) \\ \text{subject to} \quad & x_1, x_2 \geq 0, \end{aligned}$$

where $\boldsymbol{x} = [x_1, x_2]^\top$. Suppose that $\nabla f(\boldsymbol{0}) \neq \boldsymbol{0}$, and

$$\frac{\partial f}{\partial x_1}(\boldsymbol{0}) \leq 0, \qquad \frac{\partial f}{\partial x_2}(\boldsymbol{0}) \leq 0.$$

Show that $\boldsymbol{0}$ cannot be a minimizer for this problem.

6.27 Let $\boldsymbol{c} \in \mathbb{R}^n$, $\boldsymbol{c} \neq \boldsymbol{0}$, and consider the problem of minimizing the function $f(\boldsymbol{x}) = \boldsymbol{c}^\top \boldsymbol{x}$ over a constraint set $\Omega \subset \mathbb{R}^n$. Show that we cannot have a solution lying in the interior of Ω.

6.28 Consider the problem

$$\begin{aligned} \text{maximize} \quad & c_1 x_1 + c_2 x_2 \\ \text{subject to} \quad & x_1 + x_2 \leq 1 \\ & x_1, x_2 \geq 0, \end{aligned}$$

where c_1 and c_2 are constants such that $c_1 > c_2 \geq 0$. This is a *linear programming* problem (see Part III). Assuming that the problem has an optimal feasible solution, use the first-order necessary condition to show that the *unique* optimal feasible solution $\boldsymbol{x}^*$ is $[1, 0]^\top$.

Hint: First show that $\boldsymbol{x}^*$ cannot lie in the interior of the constraint set. Then, show that $\boldsymbol{x}^*$ cannot lie on the line segments $L_1 = \{\boldsymbol{x} : x_1 = 0, 0 \le x_2 < 1\}$, $L_2 = \{\boldsymbol{x} : 0 \le x_1 < 1, x_2 = 0\}$, $L_3 = \{\boldsymbol{x} : 0 \le x_1 < 1, x_2 = 1 - x_1\}$.

6.29 *Line Fitting.* Let $[x_1, y_1]^\top, \ldots, [x_n, y_n]^\top$, $n \ge 2$, be points on the $\mathbb{R}^2$ plane (each $x_i, y_i \in \mathbb{R}$). We wish to find the straight line of "best fit" through these points ("best" in the sense that the average squared error is minimized); that is, we wish to find $a, b \in \mathbb{R}$ to minimize

$$f(a,b) = \frac{1}{n}\sum_{i=1}^{n}(ax_i + b - y_i)^2 .$$

a. Let

$$\overline{X} = \frac{1}{n}\sum_{i=1}^{n} x_i,$$

$$\overline{Y} = \frac{1}{n}\sum_{i=1}^{n} y_i,$$

$$\overline{X^2} = \frac{1}{n}\sum_{i=1}^{n} x_i^2,$$

$$\overline{Y^2} = \frac{1}{n}\sum_{i=1}^{n} y_i^2,$$

$$\overline{XY} = \frac{1}{n}\sum_{i=1}^{n} x_i y_i.$$

Show that $f(a,b)$ can be written in the form $\boldsymbol{z}^\top \boldsymbol{Q}\boldsymbol{z} - 2\boldsymbol{c}^\top \boldsymbol{z} + d$, where $\boldsymbol{z} = [a,b]^\top$, $\boldsymbol{Q} = \boldsymbol{Q}^\top \in \mathbb{R}^{2\times 2}$, $\boldsymbol{c} \in \mathbb{R}^2$ and $d \in \mathbb{R}$, and find expressions for $\boldsymbol{Q}$, $\boldsymbol{c}$, and d in terms of $\overline{X}$, $\overline{Y}$, $\overline{X^2}$, $\overline{Y^2}$, and $\overline{XY}$.

b. Assume that the x_i, $i = 1, \ldots, n$, are not all equal. Find the parameters a^* and b^* for the line of best fit in terms of $\overline{X}$, $\overline{Y}$, $\overline{X^2}$, $\overline{Y^2}$, and $\overline{XY}$. Show that the point $[a^*, b^*]^\top$ is the only local minimizer of f.
Hint: $\overline{X^2} - (\overline{X})^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{X})^2$.

c. Show that if a^* and b^* are the parameters of the line of best fit, then $\overline{Y} = a^*\overline{X} + b^*$ (and hence once we have computed a^*, we can compute b^* using the formula $b^* = \overline{Y} - a^*\overline{X}$).

6.30 Suppose that we are given a set of vectors $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(p)}\}$, $\boldsymbol{x}^{(i)} \in \mathbb{R}^n$, $i = 1, \ldots, p$. Find the vector $\bar{\boldsymbol{x}} \in \mathbb{R}^n$ such that the average squared distance (norm) between $\bar{\boldsymbol{x}}$ and $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(p)}$,

$$\frac{1}{p}\sum_{i=1}^{p} \|\bar{\boldsymbol{x}} - \boldsymbol{x}^{(i)}\|^2,$$

is minimized. Use the SOSC to prove that the vector $\bar{\boldsymbol{x}}$ found above is a strict local minimizer. How is $\bar{\boldsymbol{x}}$ related to the centroid (or center of gravity) of the given set of points $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(p)}\}$?

6.31 Consider a function $f : \Omega \to \mathbb{R}$, where $\Omega \subset \mathbb{R}^n$ is a convex set and $f \in \mathcal{C}^1$. Given $\boldsymbol{x}^* \in \Omega$, suppose that there exists $c > 0$ such that $\boldsymbol{d}^\top \nabla f(\boldsymbol{x}^*) \geq c\|\boldsymbol{d}\|$ for all feasible directions $\boldsymbol{d}$ at $\boldsymbol{x}^*$. Show that $\boldsymbol{x}^*$ is a strict local minimizer of f over Ω.

6.32 Prove the following generalization of the second-order sufficient condition:

Theorem: Let Ω be a convex subset of $\mathbb{R}^n$, $f \in \mathcal{C}^2$ a real-valued function on Ω, and $\boldsymbol{x}^*$ a point in Ω. Suppose that there exists $c \in \mathbb{R}$, $c > 0$, such that for all feasible directions $\boldsymbol{d}$ at $\boldsymbol{x}^*$ ($\boldsymbol{d} \neq \boldsymbol{0}$), the following hold:

1. $\boldsymbol{d}^\top \nabla f(\boldsymbol{x}^*) \geq 0$.
2. $\boldsymbol{d}^\top \boldsymbol{F}(\boldsymbol{x}^*)\boldsymbol{d} \geq c\|\boldsymbol{d}\|^2$.

Then, $\boldsymbol{x}^*$ is a strict local minimizer of f.

6.33 Consider the quadratic function $f : \mathbb{R}^n \to \mathbb{R}$ given by

$$f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{Q}\boldsymbol{x} - \boldsymbol{x}^\top \boldsymbol{b},$$

where $\boldsymbol{Q} = \boldsymbol{Q}^\top > 0$. Show that $\boldsymbol{x}^*$ minimizes f if and only if $\boldsymbol{x}^*$ satisfies the FONC.

6.34 Consider the linear system $x_{k+1} = ax_k + bu_{k+1}$, $k \geq 0$, where $x_i \in \mathbb{R}$, $u_i \in \mathbb{R}$, and the initial condition is $x_0 = 0$. Find the values of the control inputs $u_1, \ldots, u_n$ to minimize

$$-qx_n + r\sum_{i=1}^{n} u_i^2,$$

where $q, r > 0$ are given constants. This can be interpreted as desiring to make x_n as large as possible but at the same time desiring to make the total input energy $\sum_{i=1}^{n} u_i^2$ as small as possible. The constants q and r reflect the relative weights of these two objectives.

CHAPTER 7

ONE-DIMENSIONAL SEARCH METHODS

7.1 Introduction

In this chapter, we are interested in the problem of minimizing an objective function $f : \mathbb{R} \to \mathbb{R}$ (i.e., a one-dimensional problem). The approach is to use an iterative search algorithm, also called a line-search method. One-dimensional search methods are of interest for the following reasons. First, they are special cases of search methods used in multivariable problems. Second, they are used as part of general multivariable algorithms (as described later in Section 7.8).

In an iterative algorithm, we start with an initial candidate solution $x^{(0)}$ and generate a sequence of *iterates* $x^{(1)}, x^{(2)}, \ldots$. For each iteration $k = 0, 1, 2, \ldots$, the next point $x^{(k+1)}$ depends on $x^{(k)}$ and the objective function f. The algorithm may use only the value of f at specific points, or perhaps its first derivative f', or even its second derivative f''. In this chapter, we study several algorithms:

- Golden section method (uses only f)
- Fibonacci method (uses only f)

An Introduction to Optimization, Fourth Edition.
By E. K. P. Chong and S. H. Żak.

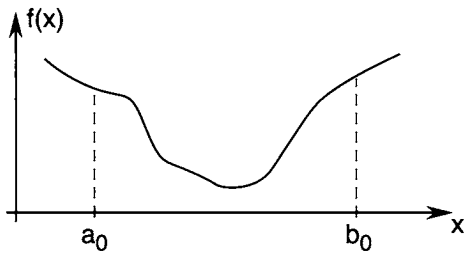

Figure 7.1 Unimodal function.

- Bisection method (uses only f')
- Secant method (uses only f')
- Newton's method (uses f' and f'')

The exposition here is based on [27].

7.2 Golden Section Search

The search methods we discuss in this and the next two sections allow us to determine the minimizer of an objective function $f : \mathbb{R} \to \mathbb{R}$ over a closed interval, say $[a_0, b_0]$. The only property that we assume of the objective function f is that it is *unimodal*, which means that f has only one local minimizer. An example of such a function is depicted in Figure 7.1.

The methods we discuss are based on evaluating the objective function at different points in the interval $[a_0, b_0]$. We choose these points in such a way that an approximation to the minimizer of f may be achieved in as few evaluations as possible. Our goal is to narrow the range progressively until the minimizer is "boxed in" with sufficient accuracy.

Consider a unimodal function f of one variable and the interval $[a_0, b_0]$. If we evaluate f at only one intermediate point of the interval, we cannot narrow the range within which we know the minimizer is located. We have to evaluate f at two intermediate points, as illustrated in Figure 7.2. We choose the intermediate points in such a way that the reduction in the range is symmetric, in the sense that

$$a_1 - a_0 = b_0 - b_1 = \rho(b_0 - a_0),$$

where

$$\rho < \frac{1}{2}.$$

We then evaluate f at the intermediate points. If $f(a_1) < f(b_1)$, then the minimizer must lie in the range $[a_0, b_1]$. If, on the other hand, $f(a_1) \geq f(b_1)$, then the minimizer is located in the range $[a_1, b_0]$ (see Figure 7.3).

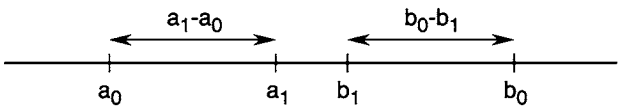

Figure 7.2 Evaluating the objective function at two intermediate points.

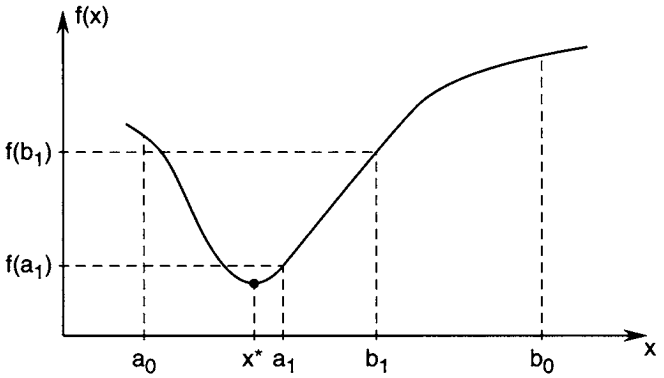

Figure 7.3 The case where $f(a_1) < f(b_1)$; the minimizer $x^* \in [a_0, b_1]$.

Starting with the reduced range of uncertainty, we can repeat the process and similarly find two new points, say a_2 and b_2, using the same value of $\rho < \frac{1}{2}$ as before. However, we would like to minimize the number of objective function evaluations while reducing the width of the uncertainty interval. Suppose, for example, that $f(a_1) < f(b_1)$, as in Figure 7.3. Then, we know that $x^* \in [a_0, b_1]$. Because a_1 is already in the uncertainty interval and $f(a_1)$ is already known, we can make a_1 coincide with b_2. Thus, only one new evaluation of f at a_2 would be necessary. To find the value of ρ that results in only one new evaluation of f, see Figure 7.4. Without loss of generality, imagine that the original range $[a_0, b_0]$ is of unit length. Then, to have only one new evaluation of f it is enough to choose ρ so that

$$\rho(b_1 - a_0) = b_1 - b_2.$$

Because $b_1 - a_0 = 1 - \rho$ and $b_1 - b_2 = 1 - 2\rho$, we have

$$\rho(1 - \rho) = 1 - 2\rho.$$

We write the quadratic equation above as

$$\rho^2 - 3\rho + 1 = 0.$$

The solutions are

$$\rho_1 = \frac{3 + \sqrt{5}}{2}, \qquad \rho_2 = \frac{3 - \sqrt{5}}{2}.$$

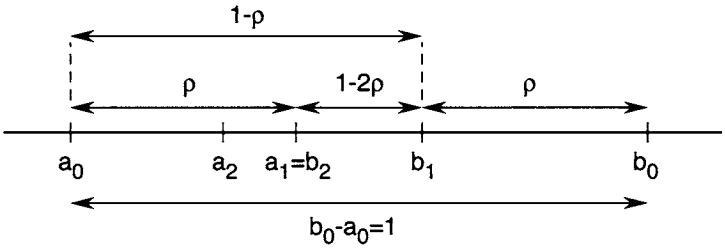

Figure 7.4 Finding value of ρ resulting in only one new evaluation of f.

Because we require that $\rho < \frac{1}{2}$, we take

$$\rho = \frac{3 - \sqrt{5}}{2} \approx 0.382.$$

Observe that

$$1 - \rho = \frac{\sqrt{5} - 1}{2}$$

and

$$\frac{\rho}{1 - \rho} = \frac{3 - \sqrt{5}}{\sqrt{5} - 1} = \frac{\sqrt{5} - 1}{2} = \frac{1 - \rho}{1},$$

that is,

$$\frac{\rho}{1 - \rho} = \frac{1 - \rho}{1}.$$

Thus, dividing a range in the ratio of ρ to $1 - \rho$ has the effect that the ratio of the shorter segment to the longer equals the ratio of the longer to the sum of the two. This rule was referred to by ancient Greek geometers as the *golden section.*

Using the golden section rule means that at every stage of the uncertainty range reduction (except the first), the objective function f need only be evaluated at one new point. The uncertainty range is reduced by the ratio $1 - \rho \approx 0.61803$ at every stage. Hence, N steps of reduction using the golden section method reduces the range by the factor

$$(1 - \rho)^N \approx (0.61803)^N.$$

Example 7.1 Suppose that we wish to use the golden section search method to find the value of x that minimizes

$$f(x) = x^4 - 14x^3 + 60x^2 - 70x$$

in the interval $[0, 2]$ (this function comes from an example in [21]). We wish to locate this value of x to within a range of 0.3.

After N stages the range $[0, 2]$ is reduced by $(0.61803)^N$. So, we choose N so that

$$(0.61803)^N \leq 0.3/2.$$

Four stages of reduction will do; that is, $N = 4$.

Iteration 1. We evaluate f at two intermediate points a_1 and b_1. We have

$$a_1 = a_0 + \rho(b_0 - a_0) = 0.7639,$$
$$b_1 = a_0 + (1 - \rho)(b_0 - a_0) = 1.236,$$

where $\rho = (3 - \sqrt{5})/2$. We compute

$$f(a_1) = -24.36,$$
$$f(b_1) = -18.96.$$

Thus, $f(a_1) < f(b_1)$, so the uncertainty interval is reduced to

$$[a_0, b_1] = [0, 1.236].$$

Iteration 2. We choose b_2 to coincide with a_1, and so f need only be evaluated at one new point,

$$a_2 = a_0 + \rho(b_1 - a_0) = 0.4721.$$

We have

$$f(a_2) = -21.10,$$
$$f(b_2) = f(a_1) = -24.36.$$

Now, $f(b_2) < f(a_2)$, so the uncertainty interval is reduced to

$$[a_2, b_1] = [0.4721, 1.236].$$

Iteration 3. We set $a_3 = b_2$ and compute b_3:

$$b_3 = a_2 + (1 - \rho)(b_1 - a_2) = 0.9443.$$

We have

$$f(a_3) = f(b_2) = -24.36,$$
$$f(b_3) = -23.59.$$

So $f(b_3) > f(a_3)$. Hence, the uncertainty interval is further reduced to

$$[a_2, b_3] = [0.4721, 0.9443].$$

Iteration 4. We set $b_4 = a_3$ and

$$a_4 = a_2 + \rho(b_3 - a_2) = 0.6525.$$

We have

$$f(a_4) = -23.84,$$
$$f(b_4) = f(a_3) = -24.36.$$

Hence, $f(a_4) > f(b_4)$. Thus, the value of x that minimizes f is located in the interval

$$[a_4, b_3] = [0.6525, 0.9443].$$

Note that $b_3 - a_4 = 0.292 < 0.3$. ∎

7.3 Fibonacci Method

Recall that the golden section method uses the same value of ρ throughout. Suppose now that we are allowed to vary the value ρ from stage to stage, so that at the kth stage in the reduction process we use a value ρ_k, at the next stage we use a value ρ_{k+1}, and so on.

As in the golden section search, our goal is to select successive values of ρ_k, $0 \leq \rho_k \leq 1/2$, such that only one new function evaluation is required at each stage. To derive the strategy for selecting evaluation points, consider Figure 7.5. From this figure we see that it is sufficient to choose the ρ_k such that

$$\rho_{k+1}(1 - \rho_k) = 1 - 2\rho_k.$$

After some manipulations, we obtain

$$\rho_{k+1} = 1 - \frac{\rho_k}{1 - \rho_k}.$$

There are many sequences $\rho_1, \rho_2, \ldots$ that satisfy the law of formation above and the condition that $0 \leq \rho_k \leq 1/2$. For example, the sequence $\rho_1 = \rho_2 = \rho_3 = \cdots = (3 - \sqrt{5})/2$ satisfies the conditions above and gives rise to the golden section method.

Suppose that we are given a sequence $\rho_1, \rho_2, \ldots$ satisfying the conditions above and we use this sequence in our search algorithm. Then, after N iterations of the algorithm, the uncertainty range is reduced by a factor of

$$(1 - \rho_1)(1 - \rho_2) \cdots (1 - \rho_N).$$

Depending on the sequence $\rho_1, \rho_2, \ldots$, we get a different reduction factor. The natural question is as follows: What sequence $\rho_1, \rho_2, \ldots$ minimizes the reduction factor above? This problem is a constrained optimization problem that can be stated formally as

$$\begin{aligned} \text{minimize} \quad & (1 - \rho_1)(1 - \rho_2) \cdots (1 - \rho_N) \\ \text{subject to} \quad & \rho_{k+1} = 1 - \frac{\rho_k}{1 - \rho_k}, \quad k = 1, \ldots, N - 1 \\ & 0 \leq \rho_k \leq \frac{1}{2}, \quad k = 1, \ldots, N. \end{aligned}$$

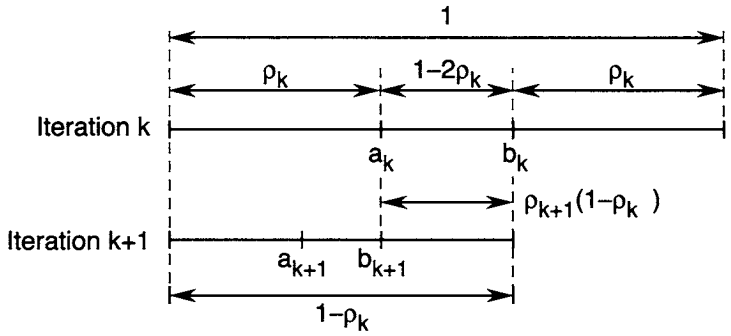

Figure 7.5 Selecting evaluation points.

Before we give the solution to the optimization problem above, we need to introduce the *Fibonacci sequence* $F_1, F_2, F_3, \ldots$. This sequence is defined as follows. First, let $F_{-1} = 0$ and $F_0 = 1$ by convention. Then, for $k \geq 0$,

$$F_{k+1} = F_k + F_{k-1}.$$

Some values of elements in the Fibonacci sequence are:

F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8
1	2	3	5	8	13	21	34

It turns out that the solution to the optimization problem above is

$$\begin{aligned}
\rho_1 &= 1 - \frac{F_N}{F_{N+1}}, \\
\rho_2 &= 1 - \frac{F_{N-1}}{F_N}, \\
&\vdots \\
\rho_k &= 1 - \frac{F_{N-k+1}}{F_{N-k+2}}, \\
&\vdots \\
\rho_N &= 1 - \frac{F_1}{F_2},
\end{aligned}$$

where the F_k are the elements of the Fibonacci sequence. The resulting algorithm is called the *Fibonacci search method.* We present a proof for the optimality of the Fibonacci search method later in this section.

In the Fibonacci search method, the uncertainty range is reduced by the factor

$$(1-\rho_1)(1-\rho_2)\cdots(1-\rho_N) = \frac{F_N}{F_{N+1}}\frac{F_{N-1}}{F_N}\cdots\frac{F_1}{F_2} = \frac{F_1}{F_{N+1}} = \frac{1}{F_{N+1}}.$$

Because the Fibonacci method uses the optimal values of $\rho_1, \rho_2, \ldots$, the reduction factor above is less than that of the golden section method. In other words, the Fibonacci method is better than the golden section method in that it gives a smaller final uncertainty range.

We point out that there is an anomaly in the final iteration of the Fibonacci search method, because

$$\rho_N = 1 - \frac{F_1}{F_2} = \frac{1}{2}.$$

Recall that we need two intermediate points at each stage, one that comes from a previous iteration and another that is a new evaluation point. However, with $\rho_N = 1/2$, the two intermediate points coincide in the middle of the uncertainty interval, and therefore we cannot further reduce the uncertainty range. To get around this problem, we perform the new evaluation for the last iteration using $\rho_N = 1/2 - \varepsilon$, where ε is a small number. In other words, the new evaluation point is just to the left or right of the midpoint of the uncertainty interval. This modification to the Fibonacci method is, of course, of no significant practical consequence.

As a result of the modification above, the reduction in the uncertainty range at the last iteration may be either

$$1 - \rho_N = \frac{1}{2}$$

or

$$1 - (\rho_N - \varepsilon) = \frac{1}{2} + \varepsilon = \frac{1 + 2\varepsilon}{2},$$

depending on which of the two points has the smaller objective function value. Therefore, in the worst case, the reduction factor in the uncertainty range for the Fibonacci method is

$$\frac{1 + 2\varepsilon}{F_{N+1}}.$$

Example 7.2 Consider the function

$$f(x) = x^4 - 14x^3 + 60x^2 - 70x.$$

Suppose that we wish to use the Fibonacci search method to find the value of x that minimizes f over the range $[0, 2]$, and locate this value of x to within the range 0.3.

After N steps the range is reduced by $(1 + 2\varepsilon)/F_{N+1}$ in the worst case. We need to choose N such that

$$\frac{1 + 2\varepsilon}{F_{N+1}} \leq \frac{\text{final range}}{\text{initial range}} = \frac{0.3}{2} = 0.15.$$

Thus, we need

$$F_{N+1} \geq \frac{1 + 2\varepsilon}{0.15}.$$

If we choose $\varepsilon \leq 0.1$, then $N = 4$ will do.

Iteration 1. We start with

$$1 - \rho_1 = \frac{F_4}{F_5} = \frac{5}{8}.$$

We then compute

$$\begin{aligned} a_1 &= a_0 + \rho_1(b_0 - a_0) = \frac{3}{4}, \\ b_1 &= a_0 + (1 - \rho_1)(b_0 - a_0) = \frac{5}{4}, \\ f(a_1) &= -24.34, \\ f(b_1) &= -18.65, \\ f(a_1) &< f(b_1). \end{aligned}$$

The range is reduced to

$$[a_0, b_1] = \left[0, \frac{5}{4}\right].$$

Iteration 2. We have

$$\begin{aligned} 1 - \rho_2 &= \frac{F_3}{F_4} = \frac{3}{5}, \\ a_2 &= a_0 + \rho_2(b_1 - a_0) = \frac{1}{2}, \\ b_2 &= a_1 = \frac{3}{4}, \\ f(a_2) &= -21.69, \\ f(b_2) &= f(a_1) = -24.34, \\ f(a_2) &> f(b_2), \end{aligned}$$

so the range is reduced to

$$[a_2, b_1] = \left[\frac{1}{2}, \frac{5}{4}\right].$$

Iteration 3. We compute

$$\begin{aligned} 1 - \rho_3 &= \frac{F_2}{F_3} = \frac{2}{3}, \\ a_3 &= b_2 = \frac{3}{4}, \\ b_3 &= a_2 + (1 - \rho_3)(b_1 - a_2) = 1, \\ f(a_3) &= f(b_2) = -24.34, \\ f(b_3) &= -23, \\ f(a_3) &< f(b_3). \end{aligned}$$

The range is reduced to

$$[a_2, b_3] = \left[\frac{1}{2}, 1\right].$$

Iteration 4. We choose $\varepsilon = 0.05$. We have

$$\begin{aligned} 1 - \rho_4 &= \frac{F_1}{F_2} = \frac{1}{2}, \\ a_4 &= a_2 + (\rho_4 - \varepsilon)(b_3 - a_2) = 0.725, \\ b_4 &= a_3 = \frac{3}{4}, \\ f(a_4) &= -24.27, \\ f(b_4) &= f(a_3) = -24.34, \\ f(a_4) &> f(b_4). \end{aligned}$$

The range is reduced to

$$[a_4, b_3] = [0.725, 1].$$

Note that $b_3 - a_4 = 0.275 < 0.3$. ∎

We now turn to a proof of the optimality of the Fibonacci search method. Skipping the rest of this section does not affect the continuity of the presentation.

To begin, recall that we wish to prove that the values of $\rho_1, \rho_2, \ldots, \rho_N$ used in the Fibonacci method, where $\rho_k = 1 - F_{N-k+1}/F_{N-k+2}$, solve the optimization problem

$$\begin{aligned} \text{minimize} \quad & (1 - \rho_1)(1 - \rho_2) \cdots (1 - \rho_N) \\ \text{subject to} \quad & \rho_{k+1} = 1 - \frac{\rho_k}{1 - \rho_k}, \ k = 1, \ldots, N - 1 \\ & 0 \leq \rho_k \leq \frac{1}{2}, \ k = 1, \ldots, N. \end{aligned}$$

It is easy to check that the values of $\rho_1, \rho_2, \ldots$ above for the Fibonacci search method satisfy the feasibility conditions in the optimization problem above (see Exercise 7.4). Recall that the Fibonacci method has an overall reduction factor of $(1 - \rho_1) \cdots (1 - \rho_N) = 1/F_{N+1}$. To prove that the Fibonacci search method is optimal, we show that for any feasible values of $\rho_1, \ldots, \rho_N$, we have $(1 - \rho_1) \cdots (1 - \rho_N) \geq 1/F_{N+1}$.

It is more convenient to work with $r_k = 1 - \rho_k$ rather than ρ_k. The optimization problem stated in terms of r_k is

$$\begin{aligned} \text{minimize} \quad & r_1 \cdots r_N \\ \text{subject to} \quad & r_{k+1} = \frac{1}{r_k} - 1, \ k = 1, \ldots, N - 1 \\ & \frac{1}{2} \leq r_k \leq 1, \ k = 1, \ldots, N. \end{aligned}$$

Note that if $r_1, r_2, \ldots$ satisfy $r_{k+1} = \frac{1}{r_k} - 1$, then $r_k \geq 1/2$ if and only if $r_{k+1} \leq 1$. Also, $r_k \geq 1/2$ if and only if $r_{k-1} \leq 2/3 \leq 1$. Therefore, in the constraints above, we may remove the constraint $r_k \leq 1$, because it is implied implicitly by $r_k \geq 1/2$ and the other constraints. Therefore, the constraints above reduce to

$$r_{k+1} = \frac{1}{r_k} - 1, \quad k = 1, \ldots, N-1,$$
$$r_k \geq \frac{1}{2}, \quad k = 1, \ldots, N.$$

To proceed, we need the following technical lemmas. In the statements of the lemmas, we assume that $r_1, r_2, \ldots$ is a sequence that satisfies

$$r_{k+1} = \frac{1}{r_k} - 1, \qquad r_k \geq \frac{1}{2}, \qquad k = 1, 2, \ldots .$$

Lemma 7.1 *For $k \geq 2$,*

$$r_k = -\frac{F_{k-2} - F_{k-1} r_1}{F_{k-3} - F_{k-2} r_1}.$$

□

Proof. We proceed by induction. For $k = 2$ we have

$$r_2 = \frac{1}{r_1} - 1 = \frac{1 - r_1}{r_1} = -\frac{F_0 - F_1 r_1}{F_{-1} - F_0 r_1}$$

and hence the lemma holds for $k = 2$. Suppose now that the lemma holds for $k \geq 2$. We show that it also holds for $k + 1$. We have

$$\begin{aligned} r_{k+1} &= \frac{1}{r_k} - 1 \\ &= \frac{-F_{k-3} + F_{k-2} r_1}{F_{k-2} - F_{k-1} r_1} - \frac{F_{k-2} - F_{k-1} r_1}{F_{k-2} - F_{k-1} r_1} \\ &= -\frac{F_{k-2} + F_{k-3} - (F_{k-1} + F_{k-2}) r_1}{F_{k-2} - F_{k-1} r_1} \\ &= -\frac{F_{k-1} - F_k r_1}{F_{k-2} - F_{k-1} r_1}, \end{aligned}$$

where we used the formation law for the Fibonacci sequence. ∎

Lemma 7.2 *For $k \geq 2$,*

$$(-1)^k (F_{k-2} - F_{k-1} r_1) > 0.$$

□

Proof. We proceed by induction. For $k = 2$, we have

$$(-1)^2(F_0 - F_1 r_1) = 1 - r_1.$$

But $r_1 = 1/(1 + r_2) \leq 2/3$, and hence $1 - r_1 > 0$. Therefore, the result holds for $k = 2$. Suppose now that the lemma holds for $k \geq 2$. We show that it also holds for $k + 1$. We have

$$(-1)^{k+1}(F_{k-1} - F_k r_1) = (-1)^{k+1} r_{k+1} \frac{1}{r_{k+1}} (F_{k-1} - F_k r_1).$$

By Lemma 7.1,

$$r_{k+1} = -\frac{F_{k-1} - F_k r_1}{F_{k-2} - F_{k-1} r_1}.$$

Substituting for $1/r_{k+1}$, we obtain

$$(-1)^{k+1}(F_{k-1} - F_k r_1) = r_{k+1}(-1)^k (F_{k-2} - F_{k-1} r_1) > 0,$$

which completes the proof. ∎

Lemma 7.3 *For $k \geq 2$,*

$$(-1)^{k+1} r_1 \geq (-1)^{k+1} \frac{F_k}{F_{k+1}}.$$

□

Proof. Because $r_{k+1} = \frac{1}{r_k} - 1$ and $r_k \geq \frac{1}{2}$, we have $r_{k+1} \leq 1$. Substituting for r_{k+1} from Lemma 7.1, we get

$$-\frac{F_{k-1} - F_k r_1}{F_{k-2} - F_{k-1} r_1} \leq 1.$$

Multiplying the numerator and denominator by $(-1)^k$ yields

$$\frac{(-1)^{k+1}(F_{k-1} - F_k r_1)}{(-1)^k (F_{k-2} - F_{k-1} r_1)} \leq 1.$$

By Lemma 7.2, $(-1)^k(F_{k-2} - F_{k-1} r_1) > 0$, and therefore we can multiply both sides of the inequality above by $(-1)^k(F_{k-2} - F_{k-1} r_1)$ to obtain

$$(-1)^{k+1}(F_{k-1} - F_k r_1) \leq (-1)^k (F_{k-2} - F_{k-1} r_1).$$

Rearranging yields

$$(-1)^{k+1}(F_{k-1} + F_k) r_1 \geq (-1)^{k+1}(F_{k-2} + F_{k-1}).$$

Using the law of formation of the Fibonacci sequence, we get

$$(-1)^{k+1} F_{k+1} r_1 \geq (-1)^{k+1} F_k,$$

which upon dividing by F_{k+1} on both sides gives the desired result. ∎

We are now ready to prove the optimality of the Fibonacci search method and the uniqueness of this optimal solution.

Theorem 7.1 *Let $r_1, \ldots, r_N$, $N \geq 2$, satisfy the constraints*

$$r_{k+1} = \frac{1}{r_k} - 1, \quad k = 1, \ldots, N-1,$$
$$r_k \geq \frac{1}{2}, \quad k = 1, \ldots, N.$$

Then,

$$r_1 \cdots r_N \geq \frac{1}{F_{N+1}}.$$

Furthermore,

$$r_1 \cdots r_N = \frac{1}{F_{N+1}}$$

if and only if $r_k = F_{N-k+1}/F_{N-k+2}$, $k = 1, \ldots, N$. In other words, the values of $r_1, \ldots, r_N$ used in the Fibonacci search method form a unique solution to the optimization problem. □

Proof. By substituting expressions for $r_1, \ldots, r_N$ from Lemma 7.1 and performing the appropriate cancellations, we obtain

$$r_1 \cdots r_N = (-1)^N (F_{N-2} - F_{N-1} r_1) = (-1)^N F_{N-2} + F_{N-1}(-1)^{N+1} r_1.$$

Using Lemma 7.3 yields

$$\begin{aligned} r_1 \cdots r_N &\geq (-1)^N F_{N-2} + F_{N-1}(-1)^{N+1} \frac{F_N}{F_{N+1}} \\ &= (-1)^N (F_{N-2} F_{N+1} - F_{N-1} F_N) \frac{1}{F_{N+1}}. \end{aligned}$$

By Exercise 7.5, it is readily checked that the following identity holds: $(-1)^N (F_{N-2} F_{N+1} - F_{N-1} F_N) = 1$. Hence,

$$r_1 \cdots r_N \geq \frac{1}{F_{N+1}}.$$

From the above we see that

$$r_1 \cdots r_N = \frac{1}{F_{N+1}}$$

if and only if

$$r_1 = \frac{F_N}{F_{N+1}}.$$

This is simply the value of r_1 for the Fibonacci search method. Note that fixing r_1 determines $r_2, \ldots, r_N$ uniquely. ∎

For further discussion on the Fibonacci search method and its variants, see [133].

7.4 Bisection Method

Again we consider finding the minimizer of an objective function $f : \mathbb{R} \to \mathbb{R}$ over an interval $[a_0, b_0]$. As before, we assume that the objective function f is unimodal. Further, suppose that f is continuously differentiable and that we can use values of the derivative f' as a basis for reducing the uncertainty interval.

The *bisection method* is a simple algorithm for successively reducing the uncertainty interval based on evaluations of the derivative. To begin, let $x^{(0)} = (a_0 + b_0)/2$ be the midpoint of the initial uncertainty interval. Next, evaluate $f'(x^{(0)})$. If $f'(x^{(0)}) > 0$, then we deduce that the minimizer lies to the *left* of $x^{(0)}$. In other words, we reduce the uncertainty interval to $[a_0, x^{(0)}]$. On the other hand, if $f'(x^{(0)}) < 0$, then we deduce that the minimizer lies to the *right* of $x^{(0)}$. In this case, we reduce the uncertainty interval to $[x^{(0)}, b_0]$. Finally, if $f'(x^{(0)}) = 0$, then we declare $x^{(0)}$ to be the minimizer and terminate our search.

With the new uncertainty interval computed, we repeat the process iteratively. At each iteration k, we compute the midpoint of the uncertainty interval. Call this point $x^{(k)}$. Depending on the sign of $f'(x^{(k)})$ (assuming that it is nonzero), we reduce the uncertainty interval to the left or right of $x^{(k)}$. If at any iteration k we find that $f'(x^{(k)}) = 0$, then we declare $x^{(k)}$ to be the minimizer and terminate our search.

Two salient features distinguish the bisection method from the golden section and Fibonacci methods. First, instead of using values of f, the bisection methods uses values of f'. Second, at each iteration, the length of the uncertainty interval is reduced by a factor of $1/2$. Hence, after N steps, the range is reduced by a factor of $(1/2)^N$. This factor is smaller than in the golden section and Fibonacci methods.

Example 7.3 Recall Example 7.1 where we wish to find the minimizer of

$$f(x) = x^4 - 14x^3 + 60x^2 - 70x$$

in the interval $[0, 2]$ to within a range of 0.3. The golden section method requires at least four stages of reduction. If, instead, we use the bisection method, we would choose N so that

$$(0.5)^N \leq 0.3/2.$$

In this case, only three stages of reduction are needed. ∎

7.5 Newton's Method

Suppose again that we are confronted with the problem of minimizing a function f of a single real variable x. We assume now that at each measurement

point $x^{(k)}$ we can determine $f(x^{(k)})$, $f'(x^{(k)})$, and $f''(x^{(k)})$. We can fit a quadratic function through $x^{(k)}$ that matches its first and second derivatives with that of the function f. This quadratic has the form

$$q(x) = f(x^{(k)}) + f'(x^{(k)})(x - x^{(k)}) + \frac{1}{2}f''(x^{(k)})(x - x^{(k)})^2.$$

Note that $q(x^{(k)}) = f(x^{(k)})$, $q'(x^{(k)}) = f'(x^{(k)})$, and $q''(x^{(k)}) = f''(x^{(k)})$. Then, instead of minimizing f, we minimize its approximation q. The first-order necessary condition for a minimizer of q yields

$$0 = q'(x) = f'(x^{(k)}) + f''(x^{(k)})(x - x^{(k)}).$$

Setting $x = x^{(k+1)}$, we obtain

$$x^{(k+1)} = x^{(k)} - \frac{f'(x^{(k)})}{f''(x^{(k)})}.$$

Example 7.4 Using Newton's method, we will find the minimizer of

$$f(x) = \frac{1}{2}x^2 - \sin x.$$

Suppose that the initial value is $x^{(0)} = 0.5$, and that the required accuracy is $\epsilon = 10^{-5}$, in the sense that we stop when $|x^{(k+1)} - x^{(k)}| < \epsilon$.

We compute

$$f'(x) = x - \cos x, \qquad f''(x) = 1 + \sin x.$$

Hence,

$$\begin{aligned} x^{(1)} &= 0.5 - \frac{0.5 - \cos 0.5}{1 + \sin 0.5} \\ &= 0.5 - \frac{-0.3775}{1.479} \\ &= 0.7552. \end{aligned}$$

Proceeding in a similar manner, we obtain

$$\begin{aligned} x^{(2)} &= x^{(1)} - \frac{f'(x^{(1)})}{f''(x^{(1)})} = x^{(1)} - \frac{0.02710}{1.685} = 0.7391, \\ x^{(3)} &= x^{(2)} - \frac{f'(x^{(2)})}{f''(x^{(2)})} = x^{(2)} - \frac{9.461 \times 10^{-5}}{1.673} = 0.7390, \\ x^{(4)} &= x^{(3)} - \frac{f'(x^{(3)})}{f''(x^{(3)})} = x^{(3)} - \frac{1.17 \times 10^{-9}}{1.673} = 0.7390. \end{aligned}$$

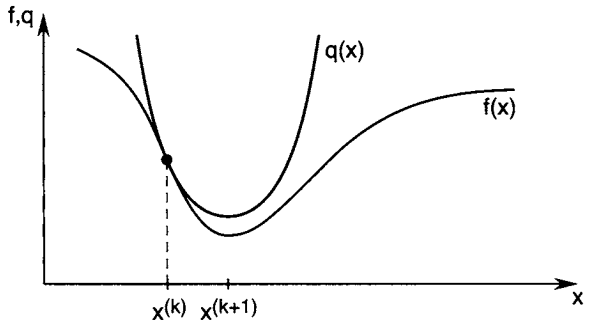

Figure 7.6 Newton's algorithm with $f''(x) > 0$.

Note that $|x^{(4)} - x^{(3)}| < \epsilon = 10^{-5}$. Furthermore, $f'(x^{(4)}) = -8.6 \times 10^{-6} \approx 0$. Observe that $f''(x^{(4)}) = 1.673 > 0$, so we can assume that $x^* \approx x^{(4)}$ is a strict minimizer. ∎

Newton's method works well if $f''(x) > 0$ everywhere (see Figure 7.6). However, if $f''(x) < 0$ for some x, Newton's method may fail to converge to the minimizer (see Figure 7.7).

Newton's method can also be viewed as a way to drive the first derivative of f to zero. Indeed, if we set $g(x) = f'(x)$, then we obtain a formula for iterative solution of the equation $g(x) = 0$:

$$x^{(k+1)} = x^{(k)} - \frac{g(x^{(k)})}{g'(x^{(k)})}.$$

In other words, we can use Newton's method for zero finding.

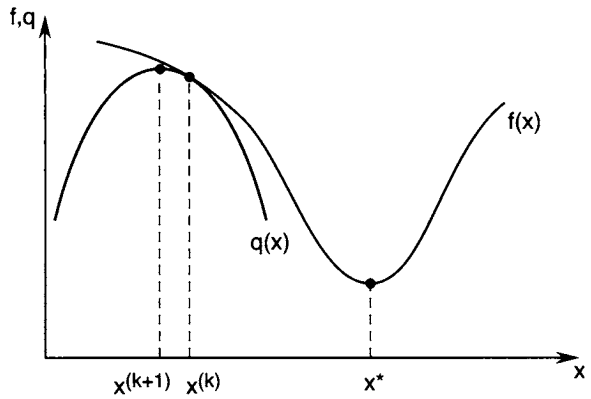

Figure 7.7 Newton's algorithm with $f''(x) < 0$.

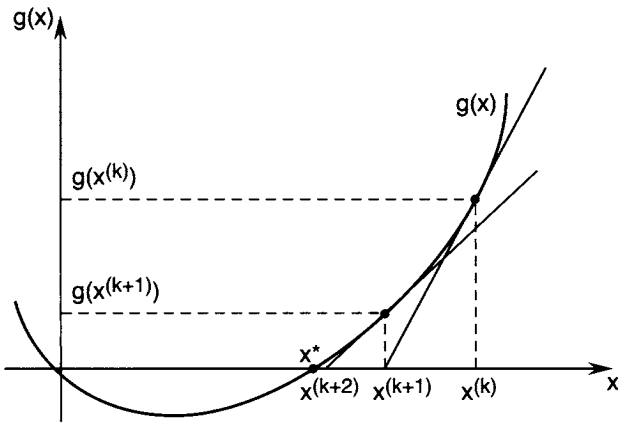

Figure 7.8 Newton's method of tangents.

Example 7.5 We apply Newton's method to improve a first approximation, $x^{(0)} = 12$, to the root of the equation

$$g(x) = x^3 - 12.2x^2 + 7.45x + 42 = 0.$$

We have $g'(x) = 3x^2 - 24.4x + 7.45$.

Performing two iterations yields

$$x^{(1)} = 12 - \frac{102.6}{146.65} = 11.33,$$
$$x^{(2)} = 11.33 - \frac{14.73}{116.11} = 11.21.$$

∎

Newton's method for solving equations of the form $g(x) = 0$ is also referred to as *Newton's method of tangents*. This name is easily justified if we look at a geometric interpretation of the method when applied to the solution of the equation $g(x) = 0$ (see Figure 7.8).

If we draw a tangent to $g(x)$ at the given point $x^{(k)}$, then the tangent line intersects the x-axis at the point $x^{(k+1)}$, which we expect to be closer to the root x^* of $g(x) = 0$. Note that the slope of $g(x)$ at $x^{(k)}$ is

$$g'(x^{(k)}) = \frac{g(x^{(k)})}{x^{(k)} - x^{(k+1)}}.$$

Hence,

$$x^{(k+1)} = x^{(k)} - \frac{g(x^{(k)})}{g'(x^{(k)})}.$$

Newton's method of tangents may fail if the first approximation to the root is such that the ratio $g(x^{(0)})/g'(x^{(0)})$ is not small enough (see Figure 7.9). Thus, an initial approximation to the root is very important.

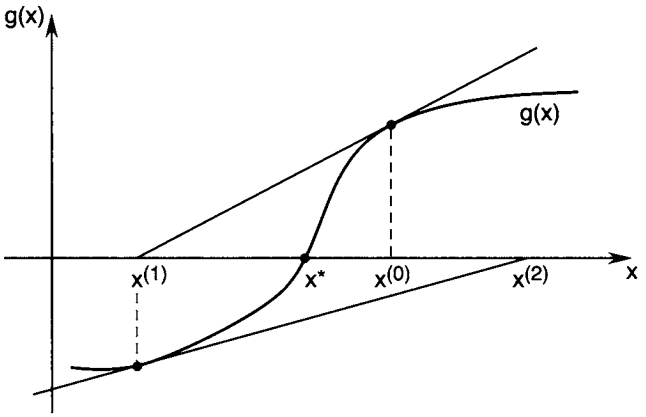

Figure 7.9 Example where Newton's method of tangents fails to converge to the root x^* of $g(x) = 0$.

7.6 Secant Method

Newton's method for minimizing f uses second derivatives of f:

$$x^{(k+1)} = x^{(k)} - \frac{f'(x^{(k)})}{f''(x^{(k)})}.$$

If the second derivative is not available, we may attempt to approximate it using first derivative information. In particular, we may approximate $f''(x^{(k)})$ above with

$$\frac{f'(x^{(k)}) - f'(x^{(k-1)})}{x^{(k)} - x^{(k-1)}}.$$

Using the foregoing approximation of the second derivative, we obtain the algorithm

$$x^{(k+1)} = x^{(k)} - \frac{x^{(k)} - x^{(k-1)}}{f'(x^{(k)}) - f'(x^{(k-1)})} f'(x^{(k)}),$$

called the *secant method.* Note that the algorithm requires two initial points to start it, which we denote $x^{(-1)}$ and $x^{(0)}$. The secant algorithm can be represented in the following equivalent form:

$$x^{(k+1)} = \frac{f'(x^{(k)})x^{(k-1)} - f'(x^{(k-1)})x^{(k)}}{f'(x^{(k)}) - f'(x^{(k-1)})}.$$

Observe that, like Newton's method, the secant method does not directly involve values of $f(x^{(k)})$. Instead, it tries to drive the derivative f' to zero. In fact, as we did for Newton's method, we can interpret the secant method as an algorithm for solving equations of the form $g(x) = 0$. Specifically, the

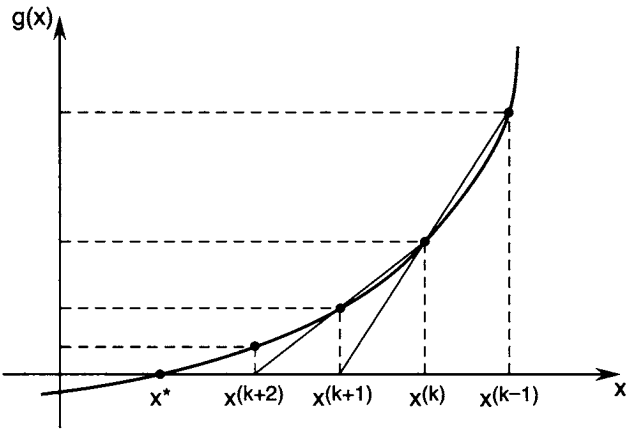

Figure 7.10 Secant method for root finding.

secant algorithm for finding a root of the equation $g(x) = 0$ takes the form

$$x^{(k+1)} = x^{(k)} - \frac{x^{(k)} - x^{(k-1)}}{g(x^{(k)}) - g(x^{(k-1)})} g(x^{(k)}),$$

or, equivalently,

$$x^{(k+1)} = \frac{g(x^{(k)})x^{(k-1)} - g(x^{(k-1)})x^{(k)}}{g(x^{(k)}) - g(x^{(k-1)})}.$$

The secant method for root finding is illustrated in Figure 7.10 (compare this with Figure 7.8). Unlike Newton's method, which uses the slope of g to determine the next point, the secant method uses the "secant" between the $(k-1)$th and kth points to determine the $(k+1)$th point.

Example 7.6 We apply the secant method to find the root of the equation

$$g(x) = x^3 - 12.2x^2 + 7.45x + 42 = 0.$$

We perform two iterations, with starting points $x^{(-1)} = 13$ and $x^{(0)} = 12$. We obtain

$$x^{(1)} = 11.40,$$
$$x^{(2)} = 11.25.$$

∎

Example 7.7 Suppose that the voltage across a resistor in a circuit decays according to the model $V(t) = e^{-Rt}$, where $V(t)$ is the voltage at time t and R is the resistance value.

Given measurements $V_1, \ldots, V_n$ of the voltage at times $t_1, \ldots, t_n$, respectively, we wish to find the best estimate of R. By the *best estimate* we mean the value of R that minimizes the total squared error between the measured voltages and the voltages predicted by the model.

We derive an algorithm to find the best estimate of R using the secant method. The objective function is

$$f(R) = \sum_{i=1}^{n} (V_i - e^{-Rt_i})^2.$$

Hence, we have

$$f'(R) = 2 \sum_{i=1}^{n} (V_i - e^{-Rt_i}) e^{-Rt_i} t_i.$$

The secant algorithm for the problem is

$$R_{k+1} = R_k - \frac{R_k - R_{k-1}}{\sum_{i=1}^{n} (V_i - e^{-R_k t_i}) e^{-R_k t_i} t_i - (V_i - e^{-R_{k-1} t_i}) e^{-R_{k-1} t_i} t_i} \times \sum_{i=1}^{n} (V_i - e^{-R_k t_i}) e^{-R_k t_i} t_i.$$

∎

For further reading on the secant method, see [32]. Newton's method and the secant method are instances of *quadratic fit* methods. In Newton's method, $x^{(k+1)}$ is the stationary point of a quadratic function that fits f' and f'' at $x^{(k)}$. In the secant method, $x^{(k+1)}$ is the stationary point of a quadratic function that fits f' at $x^{(k)}$ and $x^{(k-1)}$. The secant method uses only f' (and not f'') but needs values from *two* previous points. We leave it to the reader to verify that if we set $x^{(k+1)}$ to be the stationary point of a quadratic function that fits f at $x^{(k)}$, $x^{(k-1)}$, and $x^{(k-2)}$, we obtain a quadratic fit method that uses only values of f:

$$x^{(k+1)} = \frac{\sigma_{12} f(x^{(k)}) + \sigma_{20} f(x^{(k-1)}) + \sigma_{01} f(x^{(k-2)})}{2(\delta_{12} f(x^{(k)}) + \delta_{20} f(x^{(k-1)}) + \delta_{01} f(x^{(k-2)}))}$$

where $\sigma_{ij} = (x^{(k-i)})^2 - (x^{(k-j)})^2$ and $\delta_{ij} = x^{(k-i)} - x^{(k-j)}$ (see Exercise 7.9) This method does not use f' or f'', but needs values of f from *three* previous points. Three points are needed to initialize the iterations. The method is also sometimes called *inverse parabolic interpolation.*

An approach similar to fitting (or interpolation) based on higher-order polynomials is possible. For example, we could set $x^{(k+1)}$ to be a stationary point of a *cubic* function that fits f' at $x^{(k)}$, $x^{(k-1)}$, and $x^{(k-2)}$.

It is often practically advantageous to combine multiple methods, to overcome the limitations in any one method. For example, the golden section method is more robust but slower than inverse parabolic interpolation. *Brent's method* combines the two [17], resulting in a method that is faster than the golden section method but still retains its robustness properties.

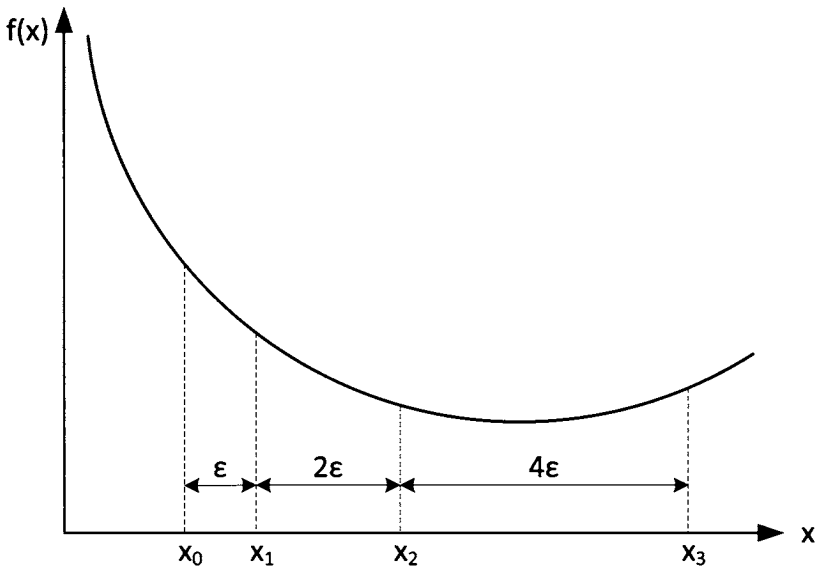

Figure 7.11 An illustration of the process of bracketing a minimizer.

7.7 Bracketing

Many of the methods we have described rely on an initial interval in which the minimizer is known to lie. This interval is also called a *bracket*, and procedures for finding such a bracket are called *bracketing* methods.

To find a bracket $[a, b]$ containing the minimizer, assuming unimodality, it suffices to find three points $a < c < b$ such that $f(c) < f(a)$ and $f(c) < f(b)$. A simple bracketing procedure is as follows. First, we pick three arbitrary points $x_0 < x_1 < x_2$. If $f(x_1) < f(x_0)$ and $f(x_1) < f(x_2)$, then we are done—the desired bracket is $[x_0, x_2]$. If not, say $f(x_0) > f(x_1) > f(x_2)$, then we pick a point $x_3 > x_2$ and check if $f(x_2) < f(x_3)$. If it holds, then again we are done—the desired bracket is $[x_1, x_3]$. Otherwise, we continue with this process until the function increases. Typically, each new point chosen involves an expansion in distance between successive test points. For example, we could double the distance between successive points, as illustrated in Figure 7.11. An analogous process applies if the initial three points are such that $f(x_0) < f(x_1) < f(x_2)$.

In the procedure described above, when the bracketing process terminates, we have three points x_{k-2}, x_{k-1}, and x_k such that $f(x_{k-1}) < f(x_{k-2})$ and $f(x_{k-1}) < f(x_k)$. The desired bracket is then $[x_{k-2}, x_k]$, which we can then use to initialize any of a number of search methods, including the golden section, Fibonacci, and bisection methods. Note that at this point, we have already evaluated $f(x_{k-2})$, $f(x_{k-1})$, and $f(x_k)$. If function evaluations are expensive to obtain, it would help if the point x_{k-1} inside the bracket also

coincides with one of the points used in the search method. For example, if we intend to use the golden section method, then it would help if $x_{k-1} - x_{k-2} = \rho(x_k - x_{k-2})$, where $\rho = (3 - \sqrt{5})/2$. In this case, x_{k-1} would be one of the two points within the initial interval used in the golden section method. This is achieved if each successive point x_k is chosen such that $x_k = x_{k-1} + (2 - \rho)(x_{k-1} - x_{k-2})$. In this case, the expansion in the distance between successive points is a factor $2 - \rho \approx 1.618$, which is less than double.

7.8 Line Search in Multidimensional Optimization

One-dimensional search methods play an important role in multidimensional optimization problems. In particular, iterative algorithms for solving such optimization problems (to be discussed in the following chapters) typically involve a *line search* at every iteration. To be specific, let $f : \mathbb{R}^n \to \mathbb{R}$ be a function that we wish to minimize. Iterative algorithms for finding a minimizer of f are of the form

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \alpha_k \boldsymbol{d}^{(k)},$$

where $\boldsymbol{x}^{(0)}$ is a given initial point and $\alpha_k \geq 0$ is chosen to minimize

$$\phi_k(\alpha) = f(\boldsymbol{x}^{(k)} + \alpha \boldsymbol{d}^{(k)}).$$

The vector $\boldsymbol{d}^{(k)}$ is called the *search direction* and α_k is called the *step size*. Figure 7.12 illustrates a line search within a multidimensional setting. Note that choice of α_k involves a one-dimensional minimization. This choice ensures that under appropriate conditions,

$$f(\boldsymbol{x}^{(k+1)}) < f(\boldsymbol{x}^{(k)}).$$

Any of the one-dimensional methods discussed in this chapter (including bracketing) can be used to minimize ϕ_k. We may, for example, use the secant method to find α_k. In this case we need the derivative of ϕ_k, which is

$$\phi_k'(\alpha) = \boldsymbol{d}^{(k)\top} \nabla f(\boldsymbol{x}^{(k)} + \alpha \boldsymbol{d}^{(k)}).$$

This is obtained using the chain rule. Therefore, applying the secant method for the line search requires the gradient ∇f, the initial line-search point $\boldsymbol{x}^{(k)}$, and the search direction $\boldsymbol{d}^{(k)}$ (see Exercise 7.11). Of course, other one-dimensional search methods may be used for line search (see, e.g., [43] and [88]).

Line-search algorithms used in practice involve considerations that we have not yet discussed thus far. First, determining the value of α_k that exactly minimizes ϕ_k may be computationally demanding; even worse, the minimizer of ϕ_k may not even exist. Second, practical experience suggests that it is better to allocate more computational time on iterating the multidimensional

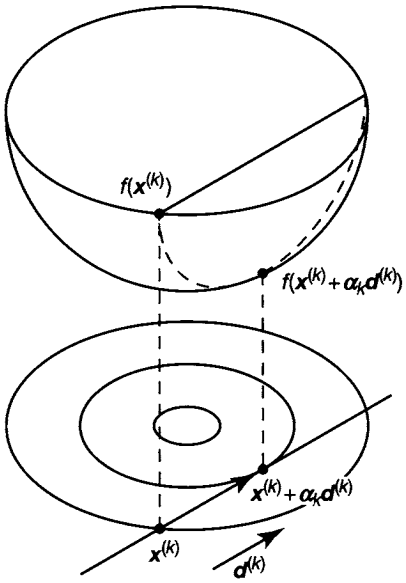

Figure 7.12 Line search in multidimensional optimization.

optimization algorithm rather than performing exact line searches. These considerations led to the development of conditions for terminating line-search algorithms that would result in low-accuracy line searches while still securing a sufficient decrease in the value of the f from one iteration to the next. The basic idea is that we have to ensure that the step size α_k is not too small or too large.

Some commonly used termination conditions are as follows. First, let $\varepsilon \in (0,1)$, $\gamma > 1$, and $\eta \in (\varepsilon, 1)$ be given constants. The *Armijo condition* ensures that α_k is not too large by requiring that

$$\phi_k(\alpha_k) \leq \phi_k(0) + \varepsilon\alpha_k\phi_k'(0).$$

Further, it ensures that α_k is not too small by requiring that

$$\phi_k(\gamma\alpha_k) \geq \phi_k(0) + \varepsilon\gamma\alpha_k\phi_k'(0).$$

The *Goldstein condition* differs from Armijo in the second inequality:

$$\phi_k(\alpha_k) \geq \phi_k(0) + \eta\alpha_k\phi_k'(0).$$

The first Armijo inequality together with the Goldstein condition are often jointly called the *Armijo-Goldstein condition.* The *Wolfe condition* differs from Goldstein in that it involves only ϕ_k':

$$\phi_k'(\alpha_k) \geq \eta\phi_k'(0).$$

A stronger variation of this is the *strong Wolfe condition*:

$$|\phi_k'(\alpha_k)| \leq \eta |\phi_k'(0)|.$$

A simple practical (inexact) line-search method is the *Armijo backtracking algorithm*, described as follows. We start with some candidate value for the step size α_k. If this candidate value satisfies a prespecified termination condition (usually the first Armijo inequality), then we stop and use it as the step size. Otherwise, we iteratively *decrease* the value by multiplying it by some constant factor $\tau \in (0,1)$ (typically $\tau = 0.5$) and re-check the termination condition. If $\alpha^{(0)}$ is the initial candidate value, then after m iterations the value obtained is $\alpha_k = \tau^m \alpha^{(0)}$. The algorithm *backtracks* from the initial value until the termination condition holds. In other words, the algorithm produces a value for the step size of the form $\alpha_k = \tau^m \alpha^{(0)}$ with m being the smallest value in $\{0, 1, 2, \ldots\}$ for which α_k satisfies the termination condition.

For more information on practical line-search methods, we refer the reader to [43, pp. 26–40], [96, Sec. 10.5], [11, App. C], [49], and [50].[1]

EXERCISES

7.1 Suppose that we have a unimodal function over the interval $[5, 8]$. Give an example of a desired final uncertainty range where the golden section method requires at least four iterations, whereas the Fibonacci method requires only three. You may choose an arbitrarily small value of ε for the Fibonacci method.

7.2 Let $f(x) = x^2 + 4\cos x$, $x \in \mathbb{R}$. We wish to find the minimizer x^* of f over the interval $[1, 2]$. (*Calculator users:* Note that in $\cos x$, the argument x is in radians.)

a. Plot $f(x)$ versus x over the interval $[1, 2]$.

b. Use the golden section method to locate x^* to within an uncertainty of 0.2. Display all intermediate steps using a table:

Iteration k	a_k	b_k	$f(a_k)$	$f(b_k)$	New uncertainty interval
1	?	?	?	?	[?,?]
2	?	?	?	?	[?,?]
⋮	⋮	⋮	⋮	⋮	⋮

c. Repeat part b using the Fibonacci method, with $\varepsilon = 0.05$. Display all intermediate steps using a table:

[1] We thank Dennis M. Goodman for furnishing us with references [49] and [50].

Iteration k	ρ_k	a_k	b_k	$f(a_k)$	$f(b_k)$	New uncertainty interval
1	?	?	?	?	?	[?,?]
2	?	?	?	?	?	[?,?]
⋮	⋮	⋮	⋮	⋮	⋮	⋮

d. Apply Newton's method, using the same number of iterations as in part b, with $x^{(0)} = 1$.

7.3 Let $f(x) = 8e^{1-x} + 7\log(x)$, where "log" represents the natural logarithm function.

a. Use MATLAB to plot $f(x)$ versus x over the interval $[1, 2]$, and verify that f is unimodal over $[1, 2]$.

b. Write a simple MATLAB program to implement the golden section method that locates the minimizer of f over $[1, 2]$ to within an uncertainty of 0.23. Display all intermediate steps using a table as in Exercise 7.2.

c. Repeat part b using the Fibonacci method, with $\varepsilon = 0.05$. Display all intermediate steps using a table as in Exercise 7.2.

7.4 Suppose that $\rho_1, \ldots, \rho_N$ are the values used in the Fibonacci search method. Show that for each $k = 1, \ldots, N$, $0 \leq \rho_k \leq 1/2$, and for each $k = 1, \ldots, N-1$,

$$\rho_{k+1} = 1 - \frac{\rho_k}{1 - \rho_k}.$$

7.5 Show that if $F_0, F_1, \ldots$ is the Fibonacci sequence, then for each $k = 2, 3, \ldots$,

$$F_{k-2}F_{k+1} - F_{k-1}F_k = (-1)^k.$$

7.6 Show that the Fibonacci sequence can be calculated using the formula

$$F_n = \frac{1}{\sqrt{5}} \left(\left(\frac{1+\sqrt{5}}{2} \right)^{n+1} - \left(\frac{1-\sqrt{5}}{2} \right)^{n+1} \right).$$

7.7 Suppose that we have an efficient way of calculating exponentials. Based on this, use Newton's method to devise a method to approximate $\log(2)$ [where "log" is the natural logarithm function]. Use an initial point of $x^{(0)} = 1$, and perform two iterations.

7.8 Consider the problem of finding the zero of $g(x) = (e^x - 1)/(e^x + 1)$, $x \in \mathbb{R}$, where e^x is the exponential of x. (Note that 0 is the unique zero of g.)

a. Write down the algorithm for Newton's method of tangents applied to this problem. Simplify using the identity $\sinh x = (e^x - e^{-x})/2$.

b. Find an initial condition $x^{(0)}$ such that the algorithm cycles [i.e., $x^{(0)} = x^{(2)} = x^{(4)} = \cdots$]. You need not explicitly calculate the initial condition; it suffices to provide an equation that the initial condition must satisfy. *Hint:* Draw a graph of g.

c. For what values of the initial condition does the algorithm converge?

7.9 Derive a one-dimensional search (minimization) algorithm based on quadratic fit with only objective function values. Specifically, derive an algorithm that computes $x^{(k+1)}$ based on $x^{(k)}$, $x^{(k-1)}$, $x^{(k-2)}$, $f(x^{(k)})$, $f(x^{(k-1)})$, and $f(x^{(k-2)})$.
Hint: To simplify, use the notation $\sigma_{ij} = (x^{(k-i)})^2 - (x^{(k-j)})^2$ and $\delta_{ij} = x^{(k-i)} - x^{(k-j)}$. You might also find it useful to experiment with your algorithm by writing a MATLAB program. Note that three points are needed to initialize the algorithm.

7.10 The objective of this exercise is to implement the secant method using MATLAB.

a. Write a simple MATLAB program to implement the secant method to locate the root of the equation $g(x) = 0$. For the stopping criterion, use the condition $|x^{(k+1)} - x^{(k)}| < |x^{(k)}|\varepsilon$, where $\varepsilon > 0$ is a given constant.

b. Let $g(x) = (2x-1)^2 + 4(4-1024x)^4$. Find the root of $g(x) = 0$ using the secant method with $x^{(-1)} = 0$, $x^{(0)} = 1$, and $\varepsilon = 10^{-5}$. Also determine the value of g at the solution obtained.

7.11 Write a MATLAB function that implements a line search algorithm using the secant method. The arguments to this function are the name of the M-file for the gradient, the current point, and the search direction. For example, the function may be called `linesearch_secant` and be used by the function call `alpha=linesearch_secant('grad',x,d)`, where `grad.m` is the M-file containing the gradient, `x` is the starting line search point, `d` is the search direction, and `alpha` is the value returned by the function [which we use in the following chapters as the step size for iterative algorithms (see, e.g., Exercises 8.25 and 10.11)].

Note: In the solutions manual, we used the stopping criterion $|\boldsymbol{d}^\top \nabla f(\boldsymbol{x} + \alpha \boldsymbol{d})| \leq \varepsilon |\boldsymbol{d}^\top \nabla f(\boldsymbol{x})|$, where $\varepsilon > 0$ is a prespecified number, ∇f is the gradient, $\boldsymbol{x}$ is the starting line search point, and $\boldsymbol{d}$ is the search direction. The rationale for the stopping criterion above is that we want to reduce the directional derivative of f in the direction $\boldsymbol{d}$ by the specified fraction ε. We used a value of $\varepsilon = 10^{-4}$ and initial conditions of 0 and 0.001.

7.12 Consider using a gradient algorithm to minimize the function

$$f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \boldsymbol{x}$$

with the initial guess $\boldsymbol{x}^{(0)} = [0.8, -0.25]^\top$.

a. To initialize the line search, apply the bracketing procedure in Figure 7.11 along the line starting at $\boldsymbol{x}^{(0)}$ in the direction of the negative gradient. Use $\varepsilon = 0.075$.

b. Apply the golden section method to reduce the width of the uncertainty region to 0.01. Organize the results of your computation in a table format similar to that of Exercise 7.2.

c. Repeat the above using the Fibonacci method.

CHAPTER 8

GRADIENT METHODS

8.1 Introduction

In this chapter we consider a class of search methods for real-valued functions on $\mathbb{R}^n$. These methods use the gradient of the given function. In our discussion we use such terms as *level sets*, *normal vectors*, and *tangent vectors*. These notions were discussed in some detail in Part I.

Recall that a level set of a function $f : \mathbb{R}^n \to \mathbb{R}$ is the set of points $\boldsymbol{x}$ satisfying $f(\boldsymbol{x}) = c$ for some constant c. Thus, a point $\boldsymbol{x}_0 \in \mathbb{R}^n$ is on the level set corresponding to level c if $f(\boldsymbol{x}_0) = c$. In the case of functions of two real variables, $f : \mathbb{R}^2 \to \mathbb{R}$, the notion of the level set is illustrated in Figure 8.1.

The gradient of f at $\boldsymbol{x}_0$, denoted $\nabla f(\boldsymbol{x}_0)$, if it is not a zero vector, is orthogonal to the tangent vector to an arbitrary smooth curve passing through $\boldsymbol{x}_0$ on the level set $f(\boldsymbol{x}) = c$. Thus, the direction of maximum rate of increase of a real-valued differentiable function at a point is orthogonal to the level set of the function through that point. In other words, the gradient acts in such a direction that for a given small displacement, the function f increases more in the direction of the gradient than in any other direction. To prove this statement, recall that $\langle \nabla f(\boldsymbol{x}), \boldsymbol{d} \rangle$, $\|\boldsymbol{d}\| = 1$, is the rate of increase of f in

An Introduction to Optimization, Fourth Edition.
By E. K. P. Chong and S. H. Żak.

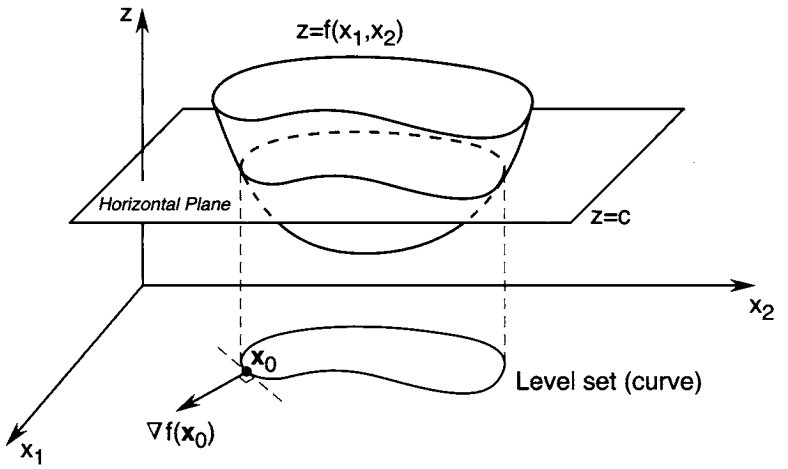

Figure 8.1 Constructing a level set corresponding to level c for f.

the direction $\boldsymbol{d}$ at the point $\boldsymbol{x}$. By the Cauchy-Schwarz inequality,

$$\langle \nabla f(\boldsymbol{x}), \boldsymbol{d} \rangle \leq \|\nabla f(\boldsymbol{x})\|$$

because $\|\boldsymbol{d}\| = 1$. But if $\boldsymbol{d} = \nabla f(\boldsymbol{x})/\|\nabla f(\boldsymbol{x})\|$, then

$$\left\langle \nabla f(\boldsymbol{x}), \frac{\nabla f(\boldsymbol{x})}{\|\nabla f(\boldsymbol{x})\|} \right\rangle = \|\nabla f(\boldsymbol{x})\|.$$

Thus, the direction in which $\nabla f(\boldsymbol{x})$ points is the direction of maximum rate of increase of f at $\boldsymbol{x}$. The direction in which $-\nabla f(\boldsymbol{x})$ points is the direction of maximum rate of decrease of f at $\boldsymbol{x}$. Hence, the direction of negative gradient is a good direction to search if we want to find a function minimizer.

We proceed as follows. Let $\boldsymbol{x}^{(0)}$ be a starting point, and consider the point $\boldsymbol{x}^{(0)} - \alpha \nabla f(\boldsymbol{x}^{(0)})$. Then, by Taylor's theorem, we obtain

$$f(\boldsymbol{x}^{(0)} - \alpha \nabla f(\boldsymbol{x}^{(0)})) = f(\boldsymbol{x}^{(0)}) - \alpha \|\nabla f(\boldsymbol{x}^{(0)})\|^2 + o(\alpha).$$

Thus, if $\nabla f(\boldsymbol{x}^{(0)}) \neq \boldsymbol{0}$, then for sufficiently small $\alpha > 0$, we have

$$f(\boldsymbol{x}^{(0)} - \alpha \nabla f(\boldsymbol{x}^{(0)})) < f(\boldsymbol{x}^{(0)}).$$

This means that the point $\boldsymbol{x}^{(0)} - \alpha \nabla f(\boldsymbol{x}^{(0)})$ is an improvement over the point $\boldsymbol{x}^{(0)}$ if we are searching for a minimizer.

To formulate an algorithm that implements this idea, suppose that we are given a point $\boldsymbol{x}^{(k)}$. To find the next point $\boldsymbol{x}^{(k+1)}$, we start at $\boldsymbol{x}^{(k)}$ and move by an amount $-\alpha_k \nabla f(\boldsymbol{x}^{(k)})$, where α_k is a positive scalar called the *step size*. This procedure leads to the following iterative algorithm:

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha_k \nabla f(\boldsymbol{x}^{(k)}).$$

We refer to this as a *gradient descent algorithm* (or simply a *gradient algorithm*). The gradient varies as the search proceeds, tending to zero as we approach the minimizer. We have the option of either taking very small steps and reevaluating the gradient at every step, or we can take large steps each time. The first approach results in a laborious method of reaching the minimizer, whereas the second approach may result in a more zigzag path to the minimizer. The advantage of the second approach is possibly fewer gradient evaluations. Among many different methods that use this philosophy the most popular is the method of *steepest descent*, which we discuss next.

Gradient methods are simple to implement and often perform well. For this reason, they are used widely in practical applications. For a discussion of applications of the steepest descent method to the computation of optimal controllers, we recommend [85, pp. 481–515]. In Chapter 13 we apply a gradient method to the training of a class of neural networks.

8.2 The Method of Steepest Descent

The method of steepest descent is a gradient algorithm where the step size α_k is chosen to achieve the maximum amount of decrease of the objective function at each individual step. Specifically, α_k is chosen to minimize $\phi_k(\alpha) \triangleq f(\boldsymbol{x}^{(k)} - \alpha \nabla f(\boldsymbol{x}^{(k)}))$. In other words,

$$\alpha_k = \arg\min_{\alpha \geq 0} f(\boldsymbol{x}^{(k)} - \alpha \nabla f(\boldsymbol{x}^{(k)})).$$

To summarize, the steepest descent algorithm proceeds as follows: At each step, starting from the point $\boldsymbol{x}^{(k)}$, we conduct a line search in the direction $-\nabla f(\boldsymbol{x}^{(k)})$ until a minimizer, $\boldsymbol{x}^{(k+1)}$, is found. A typical sequence resulting from the method of steepest descent is depicted in Figure 8.2.

Observe that the method of steepest descent moves in orthogonal steps, as stated in the following proposition.

Proposition 8.1 *If $\{\boldsymbol{x}^{(k)}\}_{k=0}^{\infty}$ is a steepest descent sequence for a given function $f : \mathbb{R}^n \to \mathbb{R}$, then for each k the vector $\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)}$ is orthogonal to the vector $\boldsymbol{x}^{(k+2)} - \boldsymbol{x}^{(k+1)}$.* □

Proof. From the iterative formula of the method of steepest descent it follows that

$$\langle \boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)}, \boldsymbol{x}^{(k+2)} - \boldsymbol{x}^{(k+1)} \rangle = \alpha_k \alpha_{k+1} \langle \nabla f(\boldsymbol{x}^{(k)}), \nabla f(\boldsymbol{x}^{(k+1)}) \rangle.$$

To complete the proof it is enough to show that

$$\langle \nabla f(\boldsymbol{x}^{(k)}), \nabla f(\boldsymbol{x}^{(k+1)}) \rangle = 0.$$

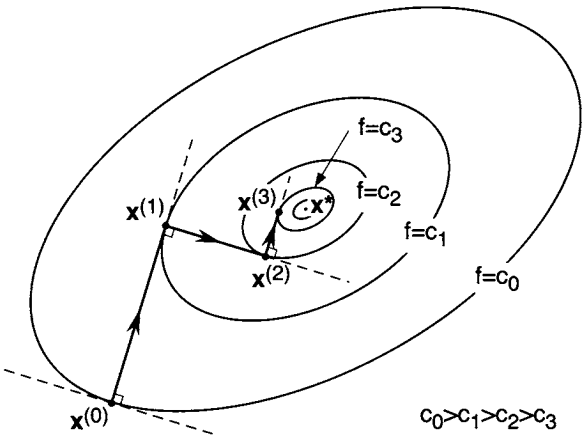

Figure 8.2 Typical sequence resulting from the method of steepest descent.

To this end, observe that α_k is a nonnegative scalar that minimizes $\phi_k(\alpha) \triangleq f(\boldsymbol{x}^{(k)} - \alpha \nabla f(\boldsymbol{x}^{(k)}))$. Hence, using the FONC and the chain rule gives us

$$\begin{aligned} 0 &= \phi_k'(\alpha_k) \\ &= \frac{d\phi_k}{d\alpha}(\alpha_k) \\ &= \nabla f(\boldsymbol{x}^{(k)} - \alpha_k \nabla f(\boldsymbol{x}^{(k)}))^\top (-\nabla f(\boldsymbol{x}^{(k)})) \\ &= -\langle \nabla f(\boldsymbol{x}^{(k+1)}), \nabla f(\boldsymbol{x}^{(k)}) \rangle, \end{aligned}$$

which completes the proof. ∎

The proposition above implies that $\nabla f(\boldsymbol{x}^{(k)})$ is parallel to the tangent plane to the level set $\{f(\boldsymbol{x}) = f(\boldsymbol{x}^{(k+1)})\}$ at $\boldsymbol{x}^{(k+1)}$. Note that as each new point is generated by the steepest descent algorithm, the corresponding value of the function f decreases in value, as stated below.

Proposition 8.2 *If $\{\boldsymbol{x}^{(k)}\}_{k=0}^{\infty}$ is the steepest descent sequence for $f : \mathbb{R}^n \to \mathbb{R}$ and if $\nabla f(\boldsymbol{x}^{(k)}) \neq \boldsymbol{0}$, then $f(\boldsymbol{x}^{(k+1)}) < f(\boldsymbol{x}^{(k)})$.* □

Proof. First recall that

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha_k \nabla f(\boldsymbol{x}^{(k)}),$$

where $\alpha_k \geq 0$ is the minimizer of

$$\phi_k(\alpha) = f(\boldsymbol{x}^{(k)} - \alpha \nabla f(\boldsymbol{x}^{(k)}))$$

over all $\alpha \geq 0$. Thus, for $\alpha \geq 0$, we have

$$\phi_k(\alpha_k) \leq \phi_k(\alpha).$$

By the chain rule,

$$\phi_k'(0) = \frac{d\phi_k}{d\alpha}(0) = -(\nabla f(\boldsymbol{x}^{(k)} - 0\nabla f(\boldsymbol{x}^{(k)})))^\top \nabla f(\boldsymbol{x}^{(k)}) = -\|\nabla f(\boldsymbol{x}^{(k)})\|^2 < 0$$

because $\nabla f(\boldsymbol{x}^{(k)}) \neq \boldsymbol{0}$ by assumption. Thus, $\phi_k'(0) < 0$ and this implies that there is an $\bar{\alpha} > 0$ such that $\phi_k(0) > \phi_k(\alpha)$ for all $\alpha \in (0, \bar{\alpha}]$. Hence,

$$f(\boldsymbol{x}^{(k+1)}) = \phi_k(\alpha_k) \leq \phi_k(\bar{\alpha}) < \phi_k(0) = f(\boldsymbol{x}^{(k)}),$$

which completes the proof. ∎

In Proposition 8.2, we proved that the algorithm possesses the *descent property*: $f(\boldsymbol{x}^{(k+1)}) < f(\boldsymbol{x}^{(k)})$ if $\nabla f(\boldsymbol{x}^{(k)}) \neq \boldsymbol{0}$. If for some k, we have $\nabla f(\boldsymbol{x}^{(k)}) = \boldsymbol{0}$, then the point $\boldsymbol{x}^{(k)}$ satisfies the FONC. In this case, $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)}$. We can use the above as the basis for a stopping (termination) criterion for the algorithm.

The condition $\nabla f(\boldsymbol{x}^{(k+1)}) = \boldsymbol{0}$, however, is not directly suitable as a practical stopping criterion, because the numerical computation of the gradient will rarely be identically equal to zero. A practical stopping criterion is to check if the norm $\|\nabla f(\boldsymbol{x}^{(k)})\|$ of the gradient is less than a prespecified threshold, in which case we stop. Alternatively, we may compute the absolute difference $|f(\boldsymbol{x}^{(k+1)}) - f(\boldsymbol{x}^{(k)})|$ between objective function values for every two successive iterations, and if the difference is less than some prespecified threshold, then we stop; that is, we stop when

$$|f(\boldsymbol{x}^{(k+1)}) - f(\boldsymbol{x}^{(k)})| < \varepsilon,$$

where $\varepsilon > 0$ is a prespecified threshold. Yet another alternative is to compute the norm $\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)}\|$ of the difference between two successive iterates, and we stop if the norm is less than a prespecified threshold:

$$\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)}\| < \varepsilon.$$

Alternatively, we may check "relative" values of the quantities above; for example,

$$\frac{|f(\boldsymbol{x}^{(k+1)}) - f(\boldsymbol{x}^{(k)})|}{|f(\boldsymbol{x}^{(k)})|} < \varepsilon$$

or

$$\frac{\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)}\|}{\|\boldsymbol{x}^{(k)}\|} < \varepsilon.$$

The two (relative) stopping criteria above are preferable to the previous (absolute) criteria because the relative criteria are "scale-independent." For example, scaling the objective function does not change the satisfaction of the criterion $|f(\boldsymbol{x}^{(k+1)}) - f(\boldsymbol{x}^{(k)})|/|f(\boldsymbol{x}^{(k)})| < \varepsilon$. Similarly, scaling the decision variable does not change the satisfaction of the criterion $\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)}\|/\|\boldsymbol{x}^{(k)})\| < \varepsilon$.

To avoid dividing by very small numbers, we can modify these stopping criteria as follows:

$$\frac{|f(\boldsymbol{x}^{(k+1)}) - f(\boldsymbol{x}^{(k)})|}{\max\{1, |f(\boldsymbol{x}^{(k)})|\}} < \varepsilon$$

or

$$\frac{\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)}\|}{\max\{1, \|\boldsymbol{x}^{(k)}\|\}} < \varepsilon.$$

Note that the stopping criteria above are relevant to all the iterative algorithms we discuss in this part.

Example 8.1 We use the method of steepest descent to find the minimizer of

$$f(x_1, x_2, x_3) = (x_1 - 4)^4 + (x_2 - 3)^2 + 4(x_3 + 5)^4.$$

The initial point is $\boldsymbol{x}^{(0)} = [4, 2, -1]^\top$. We perform three iterations.

We find that

$$\nabla f(\boldsymbol{x}) = [4(x_1 - 4)^3, 2(x_2 - 3), 16(x_3 + 5)^3]^\top.$$

Hence,

$$\nabla f(\boldsymbol{x}^{(0)}) = [0, -2, 1024]^\top.$$

To compute $\boldsymbol{x}^{(1)}$, we need

$$\begin{aligned}
\alpha_0 &= \arg\min_{\alpha \geq 0} f(\boldsymbol{x}^{(0)} - \alpha \nabla f(\boldsymbol{x}^{(0)})) \\
&= \arg\min_{\alpha \geq 0} (0 + (2 + 2\alpha - 3)^2 + 4(-1 - 1024\alpha + 5)^4) \\
&= \arg\min_{\alpha \geq 0} \phi_0(\alpha).
\end{aligned}$$

Using the secant method from Section 7.6, we obtain

$$\alpha_0 = 3.967 \times 10^{-3}.$$

For illustrative purpose, we show a plot of $\phi_0(\alpha)$ versus α in Figure 8.3, obtained using MATLAB. Thus,

$$\boldsymbol{x}^{(1)} = \boldsymbol{x}^{(0)} - \alpha_0 \nabla f(\boldsymbol{x}^{(0)}) = [4.000, 2.008, -5.062]^\top.$$

To find $\boldsymbol{x}^{(2)}$, we first determine

$$\nabla f(\boldsymbol{x}^{(1)}) = [0.000, -1.984, -0.003875]^\top.$$

Next, we find α_1, where

$$\begin{aligned}
\alpha_1 &= \arg\min_{\alpha \geq 0} (0 + (2.008 + 1.984\alpha - 3)^2 + 4(-5.062 + 0.003875\alpha + 5)^4) \\
&= \arg\min_{\alpha \geq 0} \phi_1(\alpha).
\end{aligned}$$

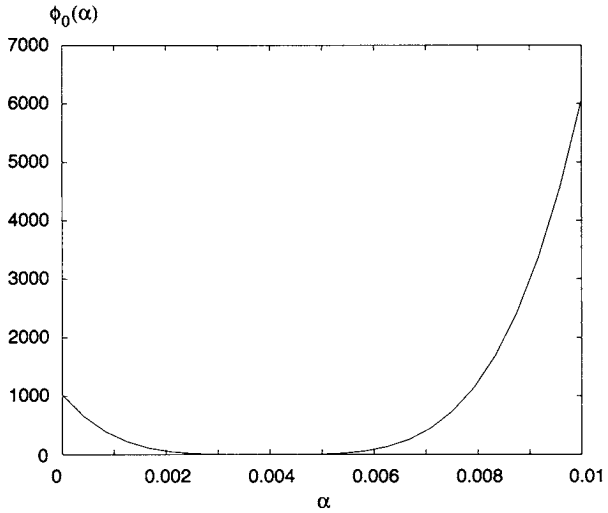

Figure 8.3 Plot of $\phi_0(\alpha)$ versus α.

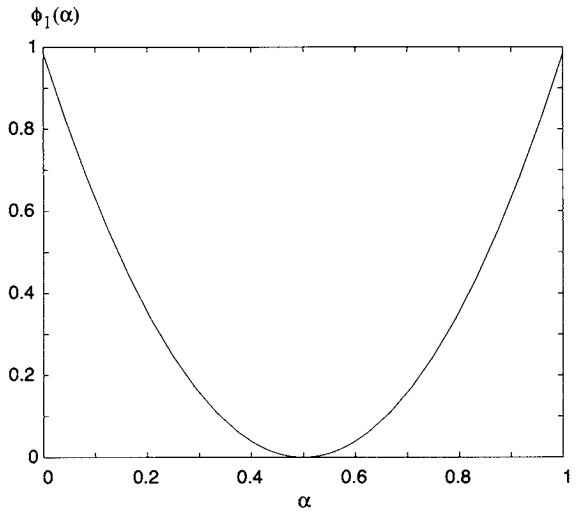

Figure 8.4 Plot of $\phi_1(\alpha)$ versus α.

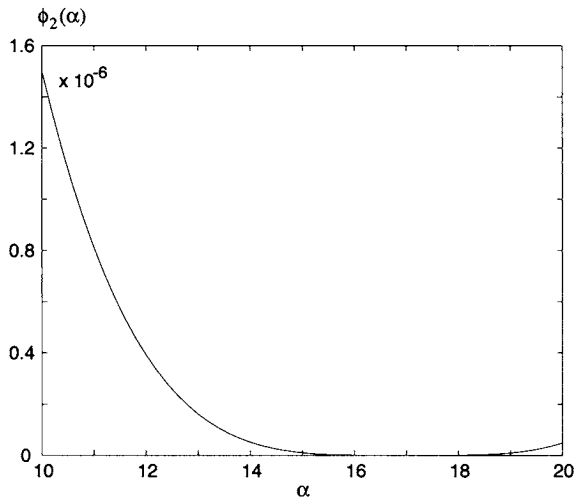

Figure 8.5 Plot of $\phi_2(\alpha)$ versus α.

Using the secant method again, we obtain $\alpha_1 = 0.5000$. Figure 8.4 depicts a plot of $\phi_1(\alpha)$ versus α. Thus,

$$\boldsymbol{x}^{(2)} = \boldsymbol{x}^{(1)} - \alpha_1 \nabla f(\boldsymbol{x}^{(1)}) = [4.000, 3.000, -5.060]^\top.$$

To find $\boldsymbol{x}^{(3)}$, we first determine

$$\nabla f(\boldsymbol{x}^{(2)}) = [0.000, 0.000, -0.003525]^\top$$

and

$$\begin{aligned}\alpha_2 &= \arg\min_{\alpha \geq 0}(0.000 + 0.000 + 4(-5.060 + 0.003525\alpha + 5)^4) \\ &= \arg\min_{\alpha \geq 0} \phi_2(\alpha).\end{aligned}$$

We proceed as in the previous iterations to obtain $\alpha_2 = 16.29$. A plot of $\phi_2(\alpha)$ versus α is shown in Figure 8.5.

The value of $\boldsymbol{x}^{(3)}$ is

$$\boldsymbol{x}^{(3)} = [4.000, 3.000, -5.002]^\top.$$

Note that the minimizer of f is $[4, 3, -5]^\top$, and hence it appears that we have arrived at the minimizer in only three iterations. The reader should be cautioned not to draw any conclusions from this example about the number of iterations required to arrive at a solution in general.

It goes without saying that numerical computations, such as those in this example, are performed in practice using a computer (rather than by hand).

The calculations above were written out explicitly, step by step, for the purpose of illustrating the operations involved in the steepest descent algorithm. The computations themselves were, in fact, carried out using a MATLAB program (see Exercise 8.25). ∎

Let us now see what the method of steepest descent does with a quadratic function of the form

$$f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{Q}\boldsymbol{x} - \boldsymbol{b}^\top \boldsymbol{x},$$

where $\boldsymbol{Q} \in \mathbb{R}^{n\times n}$ is a symmetric positive definite matrix, $\boldsymbol{b} \in \mathbb{R}^n$, and $\boldsymbol{x} \in \mathbb{R}^n$. The unique minimizer of f can be found by setting the gradient of f to zero, where

$$\nabla f(\boldsymbol{x}) = \boldsymbol{Q}\boldsymbol{x} - \boldsymbol{b},$$

because $D\left(\boldsymbol{x}^\top \boldsymbol{Q}\boldsymbol{x}\right) = \boldsymbol{x}^\top(\boldsymbol{Q} + \boldsymbol{Q}^\top) = 2\boldsymbol{x}^\top \boldsymbol{Q}$, and $D(\boldsymbol{b}^\top \boldsymbol{x}) = \boldsymbol{b}^\top$. There is no loss of generality in assuming $\boldsymbol{Q}$ to be a symmetric matrix. For if we are given a quadratic form $\boldsymbol{x}^\top \boldsymbol{A}\boldsymbol{x}$ and $\boldsymbol{A} \neq \boldsymbol{A}^\top$, then because the transposition of a scalar equals itself, we obtain

$$(\boldsymbol{x}^\top \boldsymbol{A}\boldsymbol{x})^\top = \boldsymbol{x}^\top \boldsymbol{A}^\top \boldsymbol{x} = \boldsymbol{x}^\top \boldsymbol{A}\boldsymbol{x}.$$

Hence,

$$\begin{aligned}\boldsymbol{x}^\top \boldsymbol{A}\boldsymbol{x} &= \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{A}\boldsymbol{x} + \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{A}^\top \boldsymbol{x} \\ &= \frac{1}{2}\boldsymbol{x}^\top(\boldsymbol{A} + \boldsymbol{A}^\top)\boldsymbol{x} \\ &\triangleq \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{Q}\boldsymbol{x}.\end{aligned}$$

Note that

$$(\boldsymbol{A} + \boldsymbol{A}^\top)^\top = \boldsymbol{Q}^\top = \boldsymbol{A} + \boldsymbol{A}^\top = \boldsymbol{Q}.$$

The Hessian of f is $\boldsymbol{F}(\boldsymbol{x}) = \boldsymbol{Q} = \boldsymbol{Q}^\top > 0$. To simplify the notation we write $\boldsymbol{g}^{(k)} = \nabla f(\boldsymbol{x}^{(k)})$. Then, the steepest descent algorithm for the quadratic function can be represented as

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha_k \boldsymbol{g}^{(k)},$$

where

$$\begin{aligned}\alpha_k &= \arg\min_{\alpha\geq 0} f(\boldsymbol{x}^{(k)} - \alpha \boldsymbol{g}^{(k)}) \\ &= \arg\min_{\alpha\geq 0}\left(\frac{1}{2}(\boldsymbol{x}^{(k)} - \alpha \boldsymbol{g}^{(k)})^\top \boldsymbol{Q}(\boldsymbol{x}^{(k)} - \alpha \boldsymbol{g}^{(k)}) - (\boldsymbol{x}^{(k)} - \alpha \boldsymbol{g}^{(k)})^\top \boldsymbol{b}\right).\end{aligned}$$

In the quadratic case, we can find an explicit formula for α_k. We proceed as follows. Assume that $\boldsymbol{g}^{(k)} \neq \boldsymbol{0}$, for if $\boldsymbol{g}^{(k)} = \boldsymbol{0}$, then $\boldsymbol{x}^{(k)} = \boldsymbol{x}^*$ and the

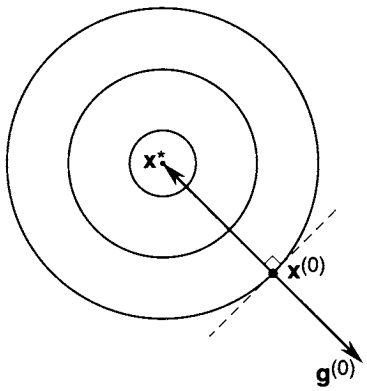

Figure 8.6 Steepest descent method applied to $f(x_1, x_2) = x_1^2 + x_2^2$.

algorithm stops. Because $\alpha_k \geq 0$ is a minimizer of $\phi_k(\alpha) = f(\boldsymbol{x}^{(k)} - \alpha \boldsymbol{g}^{(k)})$, we apply the FONC to $\phi_k(\alpha)$ to obtain

$$\phi_k'(\alpha) = (\boldsymbol{x}^{(k)} - \alpha \boldsymbol{g}^{(k)})^\top \boldsymbol{Q}(-\boldsymbol{g}^{(k)}) - \boldsymbol{b}^\top(-\boldsymbol{g}^{(k)}).$$

Therefore, $\phi_k'(\alpha) = 0$ if $\alpha \boldsymbol{g}^{(k)\top} \boldsymbol{Q} \boldsymbol{g}^{(k)} = (\boldsymbol{x}^{(k)\top} \boldsymbol{Q} - \boldsymbol{b}^\top)\boldsymbol{g}^{(k)}$. But

$$\boldsymbol{x}^{(k)\top} \boldsymbol{Q} - \boldsymbol{b}^\top = \boldsymbol{g}^{(k)\top}.$$

Hence,

$$\alpha_k = \frac{\boldsymbol{g}^{(k)\top} \boldsymbol{g}^{(k)}}{\boldsymbol{g}^{(k)\top} \boldsymbol{Q} \boldsymbol{g}^{(k)}}.$$

In summary, the method of steepest descent for the quadratic takes the form

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \frac{\boldsymbol{g}^{(k)\top} \boldsymbol{g}^{(k)}}{\boldsymbol{g}^{(k)\top} \boldsymbol{Q} \boldsymbol{g}^{(k)}} \boldsymbol{g}^{(k)},$$

where

$$\boldsymbol{g}^{(k)} = \nabla f(\boldsymbol{x}^{(k)}) = \boldsymbol{Q}\boldsymbol{x}^{(k)} - \boldsymbol{b}.$$

Example 8.2 Let

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

Then, starting from an arbitrary initial point $\boldsymbol{x}^{(0)} \in \mathbb{R}^2$, we arrive at the solution $\boldsymbol{x}^* = \boldsymbol{0} \in \mathbb{R}^2$ in only one step. See Figure 8.6.

However, if

$$f(x_1, x_2) = \frac{x_1^2}{5} + x_2^2,$$

then the method of steepest descent shuffles ineffectively back and forth when searching for the minimizer in a narrow valley (see Figure 8.7). This example illustrates a major drawback in the steepest descent method. More

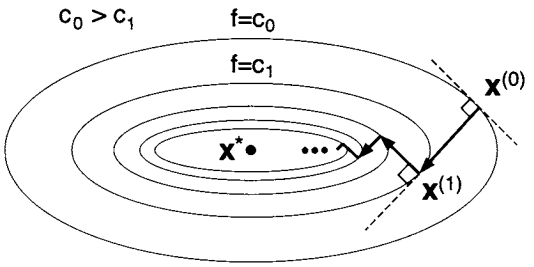

Figure 8.7 Steepest descent method in search for minimizer in a narrow valley.

sophisticated methods that alleviate this problem are discussed in subsequent chapters. ∎

To understand better the method of steepest descent, we examine its convergence properties in the next section.

8.3 Analysis of Gradient Methods

Convergence

The method of steepest descent is an example of an iterative algorithm. This means that the algorithm generates a sequence of points, each calculated on the basis of the points preceding it. The method is a *descent method* because as each new point is generated by the algorithm, the corresponding value of the objective function decreases in value (i.e., the algorithm possesses the descent property).

We say that an iterative algorithm is *globally convergent* if for any arbitrary starting point the algorithm is guaranteed to generate a sequence of points converging to a point that satisfies the FONC for a minimizer. When the algorithm is not globally convergent, it may still generate a sequence that converges to a point satisfying the FONC, provided that the initial point is sufficiently close to the point. In this case we say that the algorithm is *locally convergent.* How close to a solution point we need to start for the algorithm to converge depends on the local convergence properties of the algorithm. A related issue of interest pertaining to a given locally or globally convergent algorithm is the *rate of convergence*; that is, how fast the algorithm converges to a solution point.

In this section we analyze the convergence properties of descent gradient methods, including the method of steepest descent and gradient methods with fixed step size. We can investigate important convergence characteristics of a gradient method by applying the method to quadratic problems. The convergence analysis is more convenient if instead of working with f we deal

with

$$V(\boldsymbol{x}) = f(\boldsymbol{x}) + \frac{1}{2}\boldsymbol{x}^{*\top}\boldsymbol{Q}\boldsymbol{x}^* = \frac{1}{2}(\boldsymbol{x} - \boldsymbol{x}^*)^\top \boldsymbol{Q}(\boldsymbol{x} - \boldsymbol{x}^*),$$

where $\boldsymbol{Q} = \boldsymbol{Q}^\top > 0$. The solution point $\boldsymbol{x}^*$ is obtained by solving $\boldsymbol{Q}\boldsymbol{x} = \boldsymbol{b}$; that is, $\boldsymbol{x}^* = \boldsymbol{Q}^{-1}\boldsymbol{b}$. The function V differs from f only by a constant $\frac{1}{2}\boldsymbol{x}^{*\top}\boldsymbol{Q}\boldsymbol{x}^*$. We begin our analysis with the following useful lemma that applies to a general gradient algorithm.

Lemma 8.1 *The iterative algorithm*

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha_k \boldsymbol{g}^{(k)}$$

with $\boldsymbol{g}^{(k)} = \boldsymbol{Q}\boldsymbol{x}^{(k)} - \boldsymbol{b}$ *satisfies*

$$V(\boldsymbol{x}^{(k+1)}) = (1 - \gamma_k)\, V(\boldsymbol{x}^{(k)}),$$

where if $\boldsymbol{g}^{(k)} = \boldsymbol{0}$, *then* $\gamma_k = 1$, *and if* $\boldsymbol{g}^{(k)} \neq \boldsymbol{0}$, *then*

$$\gamma_k = \alpha_k \frac{\boldsymbol{g}^{(k)\top}\boldsymbol{Q}\boldsymbol{g}^{(k)}}{\boldsymbol{g}^{(k)\top}\boldsymbol{Q}^{-1}\boldsymbol{g}^{(k)}} \left(2\frac{\boldsymbol{g}^{(k)\top}\boldsymbol{g}^{(k)}}{\boldsymbol{g}^{(k)\top}\boldsymbol{Q}\boldsymbol{g}^{(k)}} - \alpha_k\right).$$

□

Proof. The proof is by direct computation. Note that if $\boldsymbol{g}^{(k)} = \boldsymbol{0}$, then the desired result holds trivially. In the remainder of the proof, assume that $\boldsymbol{g}^{(k)} \neq \boldsymbol{0}$. We first evaluate the expression

$$\frac{V(\boldsymbol{x}^{(k)}) - V(\boldsymbol{x}^{(k+1)})}{V(\boldsymbol{x}^{(k)})}.$$

To facilitate computations, let $\boldsymbol{y}^{(k)} = \boldsymbol{x}^{(k)} - \boldsymbol{x}^*$. Then, $V(\boldsymbol{x}^{(k)}) = \frac{1}{2}\boldsymbol{y}^{(k)\top}\boldsymbol{Q}\boldsymbol{y}^{(k)}$. Hence,

$$\begin{aligned} V(\boldsymbol{x}^{(k+1)}) &= \frac{1}{2}(\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*)^\top \boldsymbol{Q}(\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*) \\ &= \frac{1}{2}(\boldsymbol{x}^{(k)} - \boldsymbol{x}^* - \alpha_k \boldsymbol{g}^{(k)})^\top \boldsymbol{Q}(\boldsymbol{x}^{(k)} - \boldsymbol{x}^* - \alpha_k \boldsymbol{g}^{(k)}) \\ &= \frac{1}{2}\boldsymbol{y}^{(k)\top}\boldsymbol{Q}\boldsymbol{y}^{(k)} - \alpha_k \boldsymbol{g}^{(k)\top}\boldsymbol{Q}\boldsymbol{y}^{(k)} + \frac{1}{2}\alpha_k^2 \boldsymbol{g}^{(k)\top}\boldsymbol{Q}\boldsymbol{g}^{(k)}. \end{aligned}$$

Therefore,

$$\frac{V(\boldsymbol{x}^{(k)}) - V(\boldsymbol{x}^{(k+1)})}{V(\boldsymbol{x}^{(k)})} = \frac{2\alpha_k \boldsymbol{g}^{(k)\top}\boldsymbol{Q}\boldsymbol{y}^{(k)} - \alpha_k^2 \boldsymbol{g}^{(k)\top}\boldsymbol{Q}\boldsymbol{g}^{(k)}}{\boldsymbol{y}^{(k)\top}\boldsymbol{Q}\boldsymbol{y}^{(k)}}.$$

Because

$$\boldsymbol{g}^{(k)} = \boldsymbol{Q}\boldsymbol{x}^{(k)} - \boldsymbol{b} = \boldsymbol{Q}\boldsymbol{x}^{(k)} - \boldsymbol{Q}\boldsymbol{x}^* = \boldsymbol{Q}\boldsymbol{y}^{(k)},$$

we have

$$\boldsymbol{y}^{(k)\top}\boldsymbol{Q}\boldsymbol{y}^{(k)} = \boldsymbol{g}^{(k)\top}\boldsymbol{Q}^{-1}\boldsymbol{g}^{(k)},$$
$$\boldsymbol{g}^{(k)\top}\boldsymbol{Q}\boldsymbol{y}^{(k)} = \boldsymbol{g}^{(k)\top}\boldsymbol{g}^{(k)}.$$

Therefore, substituting the above, we get

$$\frac{V(\boldsymbol{x}^{(k)}) - V(\boldsymbol{x}^{(k+1)})}{V(\boldsymbol{x}^{(k)})} = \alpha_k \frac{\boldsymbol{g}^{(k)\top}\boldsymbol{Q}\boldsymbol{g}^{(k)}}{\boldsymbol{g}^{(k)\top}\boldsymbol{Q}^{-1}\boldsymbol{g}^{(k)}} \left(2\frac{\boldsymbol{g}^{(k)\top}\boldsymbol{g}^{(k)}}{\boldsymbol{g}^{(k)\top}\boldsymbol{Q}\boldsymbol{g}^{(k)}} - \alpha_k\right) = \gamma_k.$$

∎

Note that $\gamma_k \leq 1$ for all k, because $\gamma_k = 1 - V(\boldsymbol{x}^{(k+1)})/V(\boldsymbol{x}^{(k)})$ and V is a nonnegative function. If $\gamma_k = 1$ for some k, then $V(\boldsymbol{x}^{(k+1)}) = 0$, which is equivalent to $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^*$. In this case we also have that for all $i \geq k+1$, $\boldsymbol{x}^{(i)} = \boldsymbol{x}^*$ and $\gamma_i = 1$. It turns out that $\gamma_k = 1$ if and only if either $\boldsymbol{g}^{(k)} = \boldsymbol{0}$ or $\boldsymbol{g}^{(k)}$ is an eigenvector of $\boldsymbol{Q}$ (see Lemma 8.3).

We are now ready to state and prove our key convergence theorem for gradient methods. The theorem gives a necessary and sufficient condition for the sequence $\{\boldsymbol{x}^{(k)}\}$ generated by a gradient method to converge to $\boldsymbol{x}^*$; that is, $\boldsymbol{x}^{(k)} \to \boldsymbol{x}^*$ or $\lim_{k\to\infty} \boldsymbol{x}^{(k)} = \boldsymbol{x}^*$.

Theorem 8.1 *Let $\{\boldsymbol{x}^{(k)}\}$ be the sequence resulting from a gradient algorithm $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha_k \boldsymbol{g}^{(k)}$. Let γ_k be as defined in Lemma 8.1, and suppose that $\gamma_k > 0$ for all k. Then, $\{\boldsymbol{x}^{(k)}\}$ converges to $\boldsymbol{x}^*$ for any initial condition $\boldsymbol{x}^{(0)}$ if and only if*

$$\sum_{k=0}^{\infty} \gamma_k = \infty.$$

□

Proof. From Lemma 8.1 we have $V(\boldsymbol{x}^{(k+1)}) = (1-\gamma_k)\, V(\boldsymbol{x}^{(k)})$, from which we obtain

$$V(\boldsymbol{x}^{(k)}) = \left(\prod_{i=0}^{k-1}(1-\gamma_i)\right) V(\boldsymbol{x}^{(0)}).$$

Assume that $\gamma_k < 1$ for all k, for otherwise the result holds trivially. Note that $\boldsymbol{x}^{(k)} \to \boldsymbol{x}^*$ if and only if $V(\boldsymbol{x}^{(k)}) \to 0$. By the equation above we see that this occurs if and only if $\prod_{i=0}^{\infty}(1-\gamma_i) = 0$, which, in turn, holds if and only if $\sum_{i=0}^{\infty} -\log(1-\gamma_i) = \infty$ (we get this simply by taking logs). Note that by Lemma 8.1, $1 - \gamma_i \geq 0$ and $\log(1-\gamma_i)$ is well-defined [$\log(0)$ is taken to be $-\infty$]. Therefore, it remains to show that $\sum_{i=0}^{\infty} -\log(1-\gamma_i) = \infty$ if and only if

$$\sum_{i=0}^{\infty} \gamma_i = \infty.$$

We first show that $\sum_{i=0}^{\infty} \gamma_i = \infty$ implies that $\sum_{i=0}^{\infty} -\log(1-\gamma_i) = \infty$. For this, first observe that for any $x \in \mathbb{R}$, $x > 0$, we have $\log(x) \leq x - 1$

[this is easy to see simply by plotting $\log(x)$ and $x-1$ versus x]. Therefore, $\log(1-\gamma_i) \leq 1-\gamma_i-1 = -\gamma_i$, and hence $-\log(1-\gamma_i) \geq \gamma_i$. Thus, if $\sum_{i=0}^{\infty}\gamma_i = \infty$, then clearly $\sum_{i=0}^{\infty} -\log(1-\gamma_i) = \infty$.

Finally, we show that $\sum_{i=0}^{\infty} -\log(1-\gamma_i) = \infty$ implies that $\sum_{i=0}^{\infty}\gamma_i = \infty$. We proceed by contraposition. Suppose that $\sum_{i=0}^{\infty}\gamma_i < \infty$. Then, it must be that $\gamma_i \to 0$. Now observe that for $x \in \mathbb{R}$, $x \leq 1$ and x sufficiently close to 1, we have $\log(x) \geq 2(x-1)$ [as before, this is easy to see simply by plotting $\log(x)$ and $2(x-1)$ versus x]. Therefore, for sufficiently large i, $\log(1-\gamma_i) \geq 2(1-\gamma_i-1) = -2\gamma_i$, which implies that $-\log(1-\gamma_i) \leq 2\gamma_i$. Hence, $\sum_{i=0}^{\infty}\gamma_i < \infty$ implies that $\sum_{i=0}^{\infty} -\log(1-\gamma_i) < \infty$.

This completes the proof. ∎

The assumption in Theorem 8.1 that $\gamma_k > 0$ for all k is significant in that it corresponds to the algorithm having the descent property (see Exercise 8.23). Furthermore, the result of the theorem does not hold in general if we do not assume that $\gamma_k > 0$ for all k, as shown in the following example.

Example 8.3 We show, using a counterexample, that the assumption that $\gamma_k > 0$ in Theorem 8.1 is necessary for the result of the theorem to hold. Indeed, for each $k = 0, 1, 2, \ldots$, choose α_k in such a way that $\gamma_{2k} = -1/2$ and $\gamma_{2k+1} = 1/2$ (we can always do this if, for example, $\boldsymbol{Q} = \boldsymbol{I}_n$). From Lemma 8.1 we have

$$V(\boldsymbol{x}^{(2(k+1))}) = (1-1/2)(1+1/2)V(\boldsymbol{x}^{(2k)}) = (3/4)V(\boldsymbol{x}^{(2k)}).$$

Therefore, $V(\boldsymbol{x}^{(2k)}) \to 0$. Because $V(\boldsymbol{x}^{(2k+1)}) = (3/2)V(\boldsymbol{x}^{(2k)})$, we also have that $V(\boldsymbol{x}^{(2k+1)}) \to 0$. Hence, $V(\boldsymbol{x}^{(k)}) \to 0$, which implies that $\boldsymbol{x}^{(k)} \to 0$ (for all $\boldsymbol{x}^{(0)}$). On the other hand, it is clear that

$$\sum_{i=0}^{k}\gamma_i \leq \frac{1}{2}$$

for all k. Hence, the result of the theorem does not hold if $\gamma_k \leq 0$ for some k. ∎

Using the general theorem above, we can now establish the convergence of specific cases of the gradient algorithm, including the steepest descent algorithm and algorithms with fixed step size. In the analysis to follow, we use Rayleigh's inequality, which states that for any $\boldsymbol{Q} = \boldsymbol{Q}^{\top} > 0$, we have

$$\lambda_{\min}(\boldsymbol{Q})\|\boldsymbol{x}\|^2 \leq \boldsymbol{x}^{\top}\boldsymbol{Q}\boldsymbol{x} \leq \lambda_{\max}(\boldsymbol{Q})\|\boldsymbol{x}\|^2,$$

where $\lambda_{\min}(\boldsymbol{Q})$ denotes the minimal eigenvalue of $\boldsymbol{Q}$ and $\lambda_{\max}(\boldsymbol{Q})$ denotes the maximal eigenvalue of $\boldsymbol{Q}$. For $\boldsymbol{Q} = \boldsymbol{Q}^{\top} > 0$, we also have

$$\lambda_{\min}(\boldsymbol{Q}^{-1}) = \frac{1}{\lambda_{\max}(\boldsymbol{Q})},$$

$$\lambda_{\max}(\boldsymbol{Q}^{-1}) = \frac{1}{\lambda_{\min}(\boldsymbol{Q})},$$

and

$$\lambda_{\min}(\boldsymbol{Q}^{-1})\|\boldsymbol{x}\|^2 \leq \boldsymbol{x}^\top \boldsymbol{Q}^{-1}\boldsymbol{x} \leq \lambda_{\max}(\boldsymbol{Q}^{-1})\|\boldsymbol{x}\|^2.$$

Lemma 8.2 *Let $\boldsymbol{Q} = \boldsymbol{Q}^\top > 0$ be an $n \times n$ real symmetric positive definite matrix. Then, for any $\boldsymbol{x} \in \mathbb{R}^n$, we have*

$$\frac{\lambda_{\min}(\boldsymbol{Q})}{\lambda_{\max}(\boldsymbol{Q})} \leq \frac{(\boldsymbol{x}^\top \boldsymbol{x})^2}{(\boldsymbol{x}^\top \boldsymbol{Q}\boldsymbol{x})(\boldsymbol{x}^\top \boldsymbol{Q}^{-1}\boldsymbol{x})} \leq \frac{\lambda_{\max}(\boldsymbol{Q})}{\lambda_{\min}(\boldsymbol{Q})}.$$

□

Proof. Applying Rayleigh's inequality and using the properties of symmetric positive definite matrices listed previously, we get

$$\frac{(\boldsymbol{x}^\top \boldsymbol{x})^2}{(\boldsymbol{x}^\top \boldsymbol{Q}\boldsymbol{x})(\boldsymbol{x}^\top \boldsymbol{Q}^{-1}\boldsymbol{x})} \leq \frac{\|\boldsymbol{x}\|^4}{\lambda_{\min}(\boldsymbol{Q})\|\boldsymbol{x}\|^2 \lambda_{\min}(\boldsymbol{Q}^{-1})\|\boldsymbol{x}\|^2} = \frac{\lambda_{\max}(\boldsymbol{Q})}{\lambda_{\min}(\boldsymbol{Q})}$$

and

$$\frac{(\boldsymbol{x}^\top \boldsymbol{x})^2}{(\boldsymbol{x}^\top \boldsymbol{Q}\boldsymbol{x})(\boldsymbol{x}^\top \boldsymbol{Q}^{-1}\boldsymbol{x})} \geq \frac{\|\boldsymbol{x}\|^4}{\lambda_{\max}(\boldsymbol{Q})\|\boldsymbol{x}\|^2 \lambda_{\max}(\boldsymbol{Q}^{-1})\|\boldsymbol{x}\|^2} = \frac{\lambda_{\min}(\boldsymbol{Q})}{\lambda_{\max}(\boldsymbol{Q})}.$$

∎

We are now ready to establish the convergence of the steepest descent method.

Theorem 8.2 *In the steepest descent algorithm, we have $\boldsymbol{x}^{(k)} \to \boldsymbol{x}^*$ for any $\boldsymbol{x}^{(0)}$.* □

Proof. If $\boldsymbol{g}^{(k)} = \boldsymbol{0}$ for some k, then $\boldsymbol{x}^{(k)} = \boldsymbol{x}^*$ and the result holds. So assume that $\boldsymbol{g}^{(k)} \neq \boldsymbol{0}$ for all k. Recall that for the steepest descent algorithm,

$$\alpha_k = \frac{\boldsymbol{g}^{(k)\top}\boldsymbol{g}^{(k)}}{\boldsymbol{g}^{(k)\top}\boldsymbol{Q}\boldsymbol{g}^{(k)}}.$$

Substituting this expression for α_k in the formula for γ_k yields

$$\gamma_k = \frac{(\boldsymbol{g}^{(k)\top}\boldsymbol{g}^{(k)})^2}{(\boldsymbol{g}^{(k)\top}\boldsymbol{Q}\boldsymbol{g}^{(k)})(\boldsymbol{g}^{(k)\top}\boldsymbol{Q}^{-1}\boldsymbol{g}^{(k)})}.$$

Note that in this case $\gamma_k > 0$ for all k. Furthermore, by Lemma 8.2, we have $\gamma_k \geq (\lambda_{\min}(\boldsymbol{Q})/\lambda_{\max}(\boldsymbol{Q})) > 0$. Therefore, we have $\sum_{k=0}^{\infty} \gamma_k = \infty$, and hence by Theorem 8.1 we conclude that $\boldsymbol{x}^{(k)} \to \boldsymbol{x}^*$. ∎

Consider now a gradient method with fixed step size; that is, $\alpha_k = \alpha \in \mathbb{R}$ for all k. The resulting algorithm is of the form

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha \boldsymbol{g}^{(k)}.$$

We refer to the algorithm above as a *fixed-step-size* gradient algorithm. The algorithm is of practical interest because of its simplicity. In particular, the algorithm does not require a line search at each step to determine α_k, because the same step size α is used at each step. Clearly, the convergence of the algorithm depends on the choice of α, and we would not expect the algorithm to work for arbitrary α. The following theorem gives a necessary and sufficient condition on α for convergence of the algorithm.

Theorem 8.3 *For the fixed-step-size gradient algorithm,* $\boldsymbol{x}^{(k)} \to \boldsymbol{x}^*$ *for any* $\boldsymbol{x}^{(0)}$ *if and only if*

$$0 < \alpha < \frac{2}{\lambda_{\max}(\boldsymbol{Q})}.$$

□

Proof. $\Leftarrow$: By Rayleigh's inequality we have

$$\lambda_{\min}(\boldsymbol{Q})\boldsymbol{g}^{(k)\top}\boldsymbol{g}^{(k)} \leq \boldsymbol{g}^{(k)\top}\boldsymbol{Q}\boldsymbol{g}^{(k)} \leq \lambda_{\max}(\boldsymbol{Q})\boldsymbol{g}^{(k)\top}\boldsymbol{g}^{(k)}$$

and

$$\boldsymbol{g}^{(k)\top}\boldsymbol{Q}^{-1}\boldsymbol{g}^{(k)} \leq \frac{1}{\lambda_{\min}(\boldsymbol{Q})}\boldsymbol{g}^{(k)\top}\boldsymbol{g}^{(k)}.$$

Therefore, substituting the above into the formula for γ_k, we get

$$\gamma_k \geq \alpha\left(\lambda_{\min}(\boldsymbol{Q})\right)^2\left(\frac{2}{\lambda_{\max}(\boldsymbol{Q})} - \alpha\right) > 0.$$

Therefore, $\gamma_k > 0$ for all k, and $\sum_{k=0}^{\infty}\gamma_k = \infty$. Hence, by Theorem 8.1 we conclude that $\boldsymbol{x}^{(k)} \to \boldsymbol{x}^*$.

$\Rightarrow$: We use contraposition. Suppose that either $\alpha \leq 0$ or $\alpha \geq 2/\lambda_{\max}(\boldsymbol{Q})$. Let $\boldsymbol{x}^{(0)}$ be chosen such that $\boldsymbol{x}^{(0)} - \boldsymbol{x}^*$ is an eigenvector of $\boldsymbol{Q}$ corresponding to the eigenvalue $\lambda_{\max}(\boldsymbol{Q})$. Because

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha(\boldsymbol{Q}\boldsymbol{x}^{(k)} - \boldsymbol{b}) = \boldsymbol{x}^{(k)} - \alpha(\boldsymbol{Q}\boldsymbol{x}^{(k)} - \boldsymbol{Q}\boldsymbol{x}^*),$$

we obtain

$$\begin{aligned}
\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^* &= \boldsymbol{x}^{(k)} - \boldsymbol{x}^* - \alpha(\boldsymbol{Q}\boldsymbol{x}^{(k)} - \boldsymbol{Q}\boldsymbol{x}^*) \\
&= (\boldsymbol{I}_n - \alpha\boldsymbol{Q})(\boldsymbol{x}^{(k)} - \boldsymbol{x}^*) \\
&= (\boldsymbol{I}_n - \alpha\boldsymbol{Q})^{k+1}(\boldsymbol{x}^{(0)} - \boldsymbol{x}^*) \\
&= \left(1 - \alpha\lambda_{\max}(\boldsymbol{Q})\right)^{k+1}(\boldsymbol{x}^{(0)} - \boldsymbol{x}^*),
\end{aligned}$$

where in the last line we used the property that $\boldsymbol{x}^{(0)} - \boldsymbol{x}^*$ is an eigenvector of $\boldsymbol{Q}$. Taking norms on both sides, we get

$$\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*\| = |1 - \alpha\lambda_{\max}(\boldsymbol{Q})|^{k+1}\,\|\boldsymbol{x}^{(0)} - \boldsymbol{x}^*\|.$$

Because $\alpha \leq 0$ or $\alpha \geq 2/\lambda_{\max}(\boldsymbol{Q})$,

$$|1 - \alpha\lambda_{\max}(\boldsymbol{Q})| \geq 1.$$

Hence, $\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*\|$ cannot converge to 0, and thus the sequence $\{\boldsymbol{x}^{(k)}\}$ does not converge to $\boldsymbol{x}^*$. ∎

Example 8.4 Let the function f be given by

$$f(\boldsymbol{x}) = \boldsymbol{x}^\top \begin{bmatrix} 4 & 2\sqrt{2} \\ 0 & 5 \end{bmatrix} \boldsymbol{x} + \boldsymbol{x}^\top \begin{bmatrix} 3 \\ 6 \end{bmatrix} + 24.$$

We wish to find the minimizer of f using a fixed-step-size gradient algorithm

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha\nabla f(\boldsymbol{x}^{(k)}),$$

where $\alpha \in \mathbb{R}$ is a fixed step size.

To apply Theorem 8.3, we first symmetrize the matrix in the quadratic term of f to get

$$f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top \begin{bmatrix} 8 & 2\sqrt{2} \\ 2\sqrt{2} & 10 \end{bmatrix} \boldsymbol{x} + \boldsymbol{x}^\top \begin{bmatrix} 3 \\ 6 \end{bmatrix} + 24.$$

The eigenvalues of the matrix in the quadratic term are 6 and 12. Hence, using Theorem 8.3, the algorithm converges to the minimizer for all $\boldsymbol{x}^{(0)}$ if and only if α lies in the range $0 < \alpha < 2/12$. ∎

Convergence Rate

We now turn our attention to the issue of convergence rates of gradient algorithms. In particular, we focus on the steepest descent algorithm. We first present the following theorem.

Theorem 8.4 *In the method of steepest descent applied to the quadratic function, at every step k we have*

$$V(\boldsymbol{x}^{(k+1)}) \leq \frac{\lambda_{\max}(\boldsymbol{Q}) - \lambda_{\min}(\boldsymbol{Q})}{\lambda_{\max}(\boldsymbol{Q})} V(\boldsymbol{x}^{(k)}).$$

□

Proof. In the proof of Theorem 8.2, we showed that $\gamma_k \geq \lambda_{\min}(\boldsymbol{Q})/\lambda_{\max}(\boldsymbol{Q})$. Therefore,

$$\frac{V(\boldsymbol{x}^{(k)}) - V(\boldsymbol{x}^{(k+1)})}{V(\boldsymbol{x}^{(k)})} = \gamma_k \geq \frac{\lambda_{\min}(\boldsymbol{Q})}{\lambda_{\max}(\boldsymbol{Q})},$$

and the result follows. ∎

Theorem 8.4 is relevant to our consideration of the convergence rate of the steepest descent algorithm as follows. Let

$$r = \frac{\lambda_{\max}(\boldsymbol{Q})}{\lambda_{\min}(\boldsymbol{Q})} = \|\boldsymbol{Q}\|\|\boldsymbol{Q}^{-1}\|,$$

called the *condition number* of $\boldsymbol{Q}$. Then, it follows from Theorem 8.4 that

$$V(\boldsymbol{x}^{(k+1)}) \leq \left(1 - \frac{1}{r}\right) V(\boldsymbol{x}^{(k)}).$$

The term $(1 - 1/r)$ plays an important role in the convergence of $\{V(\boldsymbol{x}^{(k)})\}$ to 0 (and hence of $\{\boldsymbol{x}^{(k)}\}$ to $\boldsymbol{x}^*$). We refer to $(1 - 1/r)$ as the *convergence ratio*. Specifically, we see that the smaller the value of $(1 - 1/r)$, the smaller $V(\boldsymbol{x}^{(k+1)})$ will be relative to $V(\boldsymbol{x}^{(k)})$, and hence the "faster" $V(\boldsymbol{x}^{(k)})$ converges to 0, as indicated by the inequality above. The convergence ratio $(1 - 1/r)$ decreases as r decreases. If $r = 1$, then $\lambda_{\max}(\boldsymbol{Q}) = \lambda_{\min}(\boldsymbol{Q})$, corresponding to circular contours of f (see Figure 8.6). In this case the algorithm converges in a single step to the minimizer. As r increases, the speed of convergence of $\{V(\boldsymbol{x}^{(k)})\}$ (and hence of $\{\boldsymbol{x}^{(k)}\}$) decreases. The increase in r reflects that fact that the contours of f are more eccentric (see, e.g., Figure 8.7). We refer the reader to [88, pp. 238, 239] for an alternative approach to the analysis above.

To investigate the convergence properties of $\{\boldsymbol{x}^{(k)}\}$ further, we need the following definition.

Definition 8.1 Given a sequence $\{\boldsymbol{x}^{(k)}\}$ that converges to $\boldsymbol{x}^*$, that is, $\lim_{k\to\infty} \|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\| = 0$, we say that the *order of convergence* is p, where $p \in \mathbb{R}$, if

$$0 < \lim_{k\to\infty} \frac{\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*\|}{\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|^p} < \infty.$$

If for all $p > 0$,

$$\lim_{k\to\infty} \frac{\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*\|}{\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|^p} = 0,$$

then we say that the order of convergence is ∞. ∎

Note that in the definition above, 0/0 should be understood to be 0.

The order of convergence of a sequence is a measure of its rate of convergence; the higher the order, the faster the rate of convergence. The order of convergence is sometimes also called the *rate of convergence* (see, e.g., [96]). If $p = 1$ (first-order convergence) and $\lim_{k\to\infty} \|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*\|/\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\| = 1$, we say that the convergence is *sublinear*. If $p = 1$ and $\lim_{k\to\infty} \|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*\|/\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\| < 1$, we say that the convergence is *linear*. If $p > 1$, we say that the convergence is *superlinear*. If $p = 2$ (second-order convergence), we say that the convergence is *quadratic*.

Example 8.5 1. Suppose that $x^{(k)} = 1/k$ and thus $x^{(k)} \to 0$. Then,

$$\frac{|x^{(k+1)}|}{|x^{(k)}|^p} = \frac{1/(k+1)}{1/k^p} = \frac{k^p}{k+1}.$$

If $p < 1$, the sequence above converges to 0, whereas if $p > 1$, it grows to ∞. If $p = 1$, the sequence converges to 1. Hence, the order of convergence is 1 (i.e., we have linear convergence).

2. Suppose that $x^{(k)} = \gamma^k$, where $0 < \gamma < 1$, and thus $x^{(k)} \to 0$. Then,

$$\frac{|x^{(k+1)}|}{|x^{(k)}|^p} = \frac{\gamma^{k+1}}{(\gamma^k)^p} = \gamma^{k+1-kp} = \gamma^{k(1-p)+1}.$$

If $p < 1$, the sequence above converges to 0, whereas if $p > 1$, it grows to ∞. If $p = 1$, the sequence converges to γ (in fact, remains constant at γ). Hence, the order of convergence is 1.

3. Suppose that $x^{(k)} = \gamma^{(q^k)}$, where $q > 1$ and $0 < \gamma < 1$, and thus $x^{(k)} \to 0$. Then,

$$\frac{|x^{(k+1)}|}{|x^{(k)}|^p} = \frac{\gamma^{(q^{k+1})}}{(\gamma^{(q^k)})^p} = \gamma^{(q^{k+1}-pq^k)} = \gamma^{(q-p)q^k}.$$

If $p < q$, the sequence above converges to 0, whereas if $p > q$, it grows to ∞. If $p = q$, the sequence converges to 1 (in fact, remains constant at 1). Hence, the order of convergence is q.

4. Suppose that $x^{(k)} = 1$ for all k, and thus $x^{(k)} \to 1$ trivially. Then,

$$\frac{|x^{(k+1)} - 1|}{|x^{(k)} - 1|^p} = \frac{0}{0^p} = 0$$

for all p. Hence, the order of convergence is ∞.

∎

The order of convergence can be interpreted using the notion of the order symbol O, as follows. Recall that $a = O(h)$ ("big-oh of h") if there exists a constant c such that $|a| \leq c|h|$ for sufficiently small h. Then, the order of convergence is *at least* p if

$$\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*\| = O(\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|^p)$$

(see Theorem 8.5 below). For example, the order of convergence is at least 2 if

$$\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*\| = O(\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|^2)$$

(this fact is used in the analysis of Newton's algorithm in Chapter 9).

Theorem 8.5 *Let $\{\boldsymbol{x}^{(k)}\}$ be a sequence that converges to $\boldsymbol{x}^*$. If*

$$\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*\| = O(\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|^p),$$

then the order of convergence (if it exists) is at least p. □

Proof. Let s be the order of convergence of $\{\boldsymbol{x}^{(k)}\}$. Suppose that

$$\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*\| = O(\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|^p).$$

Then, there exists c such that for sufficiently large k,

$$\frac{\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*\|}{\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|^p} \leq c.$$

Hence,

$$\begin{aligned}\frac{\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*\|}{\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|^s} &= \frac{\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*\|}{\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|^p}\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|^{p-s} \\ &\leq c\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|^{p-s}.\end{aligned}$$

Taking limits yields

$$\lim_{k\to\infty} \frac{\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*\|}{\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|^s} \leq c \lim_{k\to\infty} \|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|^{p-s}.$$

Because by definition s is the order of convergence,

$$\lim_{k\to\infty} \frac{\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*\|}{\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|^s} > 0.$$

Combining the two inequalities above, we get

$$c \lim_{k\to\infty} \|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|^{p-s} > 0.$$

Therefore, because $\lim_{k\to\infty} \|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\| = 0$, we conclude that $s \geq p$; that is, the order of convergence is at least p. ∎

By an argument similar to the above, we can show that if

$$\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*\| = o(\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|^p),$$

then the order of convergence (if it exists) strictly exceeds p.

Example 8.6 Suppose that we are given a scalar sequence $\{x^{(k)}\}$ that converges with order of convergence p and satisfies

$$\lim_{k\to\infty} \frac{|x^{(k+1)} - 2|}{|x^{(k)} - 2|^3} = 0.$$

The limit of $\{x^{(k)}\}$ must be 2, because it is clear from the equation that $|x^{(k+1)} - 2| \to 0$. Also, we see that $|x^{(k+1)} - 2| = o(|x^{(k)} - 2|^3)$. Hence, we conclude that $p > 3$. ∎

It turns out that the order of convergence of any convergent sequence cannot be less than 1 (see Exercise 8.3). In the following, we provide an example where the order of convergence of a fixed-step-size gradient algorithm exceeds 1.

Example 8.7 Consider the problem of finding a minimizer of the function $f : \mathbb{R} \to \mathbb{R}$ given by

$$f(x) = x^2 - \frac{x^3}{3}.$$

Suppose that we use the algorithm $x^{(k+1)} = x^{(k)} - \alpha f'(x^{(k)})$ with step size $\alpha = 1/2$ and initial condition $x^{(0)} = 1$. (The notation f' represents the derivative of f.)

We first show that the algorithm converges to a local minimizer of f. Indeed, we have $f'(x) = 2x - x^2$. The fixed-step-size gradient algorithm with step size $\alpha = 1/2$ is therefore given by

$$x^{(k+1)} = x^{(k)} - \alpha f'(x^{(k)}) = \frac{1}{2}(x^{(k)})^2.$$

With $x^{(0)} = 1$, we can derive the expression $x^{(k)} = (1/2)^{2^k - 1}$. Hence, the algorithm converges to 0, a strict local minimizer of f.

Next, we find the order of convergence. Note that

$$\frac{|x^{(k+1)}|}{|x^{(k)}|^2} = \frac{1}{2}.$$

Therefore, the order of convergence is 2. ∎

Finally, we show that the steepest descent algorithm has an order of convergence of 1 in the *worst case*; that is, there are cases for which the order of convergence of the steepest descent algorithm is equal to 1. To proceed, we will need the following simple lemma.

Lemma 8.3 *In the steepest descent algorithm, if* $\boldsymbol{g}^{(k)} \neq \boldsymbol{0}$ *for all* k, *then* $\gamma_k = 1$ *if and only if* $\boldsymbol{g}^{(k)}$ *is an eigenvector of* $\boldsymbol{Q}$. □

Proof. Suppose that $\boldsymbol{g}^{(k)} \neq \boldsymbol{0}$ for all k. Recall that for the steepest descent algorithm,

$$\gamma_k = \frac{(\boldsymbol{g}^{(k)\top}\boldsymbol{g}^{(k)})^2}{(\boldsymbol{g}^{(k)\top}\boldsymbol{Q}\boldsymbol{g}^{(k)})(\boldsymbol{g}^{(k)\top}\boldsymbol{Q}^{-1}\boldsymbol{g}^{(k)})}.$$

Sufficiency is easy to show by verification. To show necessity, suppose that $\gamma_k = 1$. Then, $V(\boldsymbol{x}^{(k+1)}) = 0$, which implies that $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^*$. Therefore,

$$\boldsymbol{x}^* = \boldsymbol{x}^{(k)} - \alpha_k \boldsymbol{g}^{(k)}.$$

Premultiplying by $\boldsymbol{Q}$ and subtracting $\boldsymbol{b}$ from both sides yields

$$\mathbf{0} = \boldsymbol{g}^{(k)} - \alpha_k \boldsymbol{Q}\boldsymbol{g}^{(k)},$$

which can be rewritten as

$$\boldsymbol{Q}\boldsymbol{g}^{(k)} = \frac{1}{\alpha_k}\boldsymbol{g}^{(k)}.$$

Hence, $\boldsymbol{g}^{(k)}$ is an eigenvector of $\boldsymbol{Q}$. ∎

By the lemma, if $\boldsymbol{g}^{(k)}$ is not an eigenvector of $\boldsymbol{Q}$, then $\gamma_k < 1$ (recall that γ_k cannot exceed 1). We use this fact in the proof of the following result on the worst-case order of convergence of the steepest descent algorithm.

Theorem 8.6 *Let $\{\boldsymbol{x}^{(k)}\}$ be a convergent sequence of iterates of the steepest descent algorithm applied to a function f. Then, the order of convergence of $\{\boldsymbol{x}^{(k)}\}$ is 1 in the worst case; that is, there exist a function f and an initial condition $\boldsymbol{x}^{(0)}$ such that the order of convergence of $\{\boldsymbol{x}^{(k)}\}$ is equal to 1.* □

Proof. Let $f : \mathbb{R}^n \to \mathbb{R}$ be a quadratic function with Hessian $\boldsymbol{Q}$. Assume that the maximum and minimum eigenvalues of $\boldsymbol{Q}$ satisfy $\lambda_{\max}(\boldsymbol{Q}) > \lambda_{\min}(\boldsymbol{Q})$. To show that the order of convergence of $\{\boldsymbol{x}^{(k)}\}$ is 1, it suffices to show that there exists $\boldsymbol{x}^{(0)}$ such that

$$\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*\| \geq c\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|$$

for some $c > 0$ (see Exercise 8.2). Indeed, by Rayleigh's inequality,

$$\begin{aligned} V(\boldsymbol{x}^{(k+1)}) &= \frac{1}{2}(\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*)^\top \boldsymbol{Q}(\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*) \\ &\leq \frac{\lambda_{\max}(\boldsymbol{Q})}{2}\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*\|^2. \end{aligned}$$

Similarly,

$$V(\boldsymbol{x}^{(k)}) \geq \frac{\lambda_{\min}(\boldsymbol{Q})}{2}\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|^2.$$

Combining the inequalities above with Lemma 8.1, we obtain

$$\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*\| \geq \sqrt{(1-\gamma_k)\frac{\lambda_{\min}(\boldsymbol{Q})}{\lambda_{\max}(\boldsymbol{Q})}}\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|.$$

Therefore, it suffices to choose $\boldsymbol{x}^{(0)}$ such that $\gamma_k \leq d$ for some $d < 1$.

Recall that for the steepest descent algorithm, assuming that $\boldsymbol{g}^{(k)} \neq \mathbf{0}$ for all k, γ_k depends on $\boldsymbol{g}^{(k)}$ according to

$$\gamma_k = \frac{(\boldsymbol{g}^{(k)\top}\boldsymbol{g}^{(k)})^2}{(\boldsymbol{g}^{(k)\top}\boldsymbol{Q}\boldsymbol{g}^{(k)})(\boldsymbol{g}^{(k)\top}\boldsymbol{Q}^{-1}\boldsymbol{g}^{(k)})}.$$

First consider the case where $n = 2$. Suppose that $\boldsymbol{x}^{(0)} \neq \boldsymbol{x}^*$ is chosen such that $\boldsymbol{x}^{(0)} - \boldsymbol{x}^*$ is not an eigenvector of $\boldsymbol{Q}$. Then, $\boldsymbol{g}^{(0)} = \boldsymbol{Q}(\boldsymbol{x}^{(0)} - \boldsymbol{x}^*) \neq \boldsymbol{0}$ is also not an eigenvector of $\boldsymbol{Q}$. By Proposition 8.1, $\boldsymbol{g}^{(k)} = (\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)})/\alpha_k$ is not an eigenvector of $\boldsymbol{Q}$ for any k [because any two eigenvectors corresponding to $\lambda_{\max}(\boldsymbol{Q})$ and $\lambda_{\min}(\boldsymbol{Q})$ are mutually orthogonal]. Also, $\boldsymbol{g}^{(k)}$ lies in one of two mutually orthogonal directions. Therefore, by Lemma 8.3, for each k, the value of γ_k is one of two numbers, both of which are strictly less than 1. This proves the $n = 2$ case.

For the general n case, let $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ be mutually orthogonal eigenvectors corresponding to $\lambda_{\max}(\boldsymbol{Q})$ and $\lambda_{\min}(\boldsymbol{Q})$. Choose $\boldsymbol{x}^{(0)}$ such that $\boldsymbol{x}^{(0)} - \boldsymbol{x}^* \neq \boldsymbol{0}$ lies in the span of $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ but is not equal to either. Note that $\boldsymbol{g}^{(0)} = \boldsymbol{Q}(\boldsymbol{x}^{(0)} - \boldsymbol{x}^*)$ also lies in the span of $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$, but is not equal to either. By manipulating $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha_k \boldsymbol{g}^{(k)}$ as before, we can write $\boldsymbol{g}^{(k+1)} = (\boldsymbol{I} - \alpha_k \boldsymbol{Q})\boldsymbol{g}^{(k)}$. Any eigenvector of $\boldsymbol{Q}$ is also an eigenvector of $\boldsymbol{I} - \alpha_k \boldsymbol{Q}$. Therefore, $\boldsymbol{g}^{(k)}$ lies in the span of $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ for all k; that is, the sequence $\{\boldsymbol{g}^{(k)}\}$ is confined within the two-dimensional subspace spanned by $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$. We can now proceed as in the $n = 2$ case. ∎

In the next chapter we discuss Newton's method, which has order of convergence at least 2 if the initial guess is near the solution.

EXERCISES

8.1 Perform two iterations leading to the minimization of

$$f(x_1, x_2) = x_1 + \frac{1}{2}x_2 + \frac{1}{2}x_1^2 + x_2^2 + 3$$

using the steepest descent method with the starting point $\boldsymbol{x}^{(0)} = \boldsymbol{0}$. Also determine an optimal solution analytically.

8.2 Let $\{\boldsymbol{x}^{(k)}\}$ be a sequence that converges to $\boldsymbol{x}^*$. Show that if there exists $c > 0$ such that

$$\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*\| \geq c\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|^p$$

for sufficiently large k, then the order of convergence (if it exists) is at most p.

8.3 Let $\{\boldsymbol{x}^{(k)}\}$ be a sequence that converges to $\boldsymbol{x}^*$. Show that there does not exist $p < 1$ such that

$$\lim_{k\to\infty} \frac{\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*\|}{\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|^p} > 0.$$

8.4 Consider the sequence $\{x^{(k)}\}$ given by $x^{(k)} = 2^{-2^k}$.

a. Write down the value of the limit of $\{x^{(k)}\}$.

b. Find the order of convergence of $\{x^{(k)}\}$.

8.5 Consider the two sequences $\{x^{(k)}\}$ and $\{y^{(k)}\}$ defined iteratively as follows:

$$x^{(k+1)} = ax^{(k)},$$
$$y^{(k+1)} = (y^{(k)})^b,$$

where $a \in \mathbb{R}$, $b \in \mathbb{R}$, $0 < a < 1$, $b > 1$, $x^{(0)} \neq 0$, $y^{(0)} \neq 0$, and $|y^{(0)}| < 1$.

a. Derive a formula for $x^{(k)}$ in terms of $x^{(0)}$ and a. Use this to deduce that $x^{(k)} \to 0$.

b. Derive a formula for $y^{(k)}$ in terms of $y^{(0)}$ and b. Use this to deduce that $y^{(k)} \to 0$.

c. Find the order of convergence of $\{x^{(k)}\}$ and the order of convergence of $\{y^{(k)}\}$.

d. Calculate the smallest number of iterations k such that $|x^{(k)}| \leq c|x^{(0)}|$, where $0 < c < 1$.
Hint: The answer is in terms of a and c. You may use the notation $\lceil z \rceil$ to represent the smallest integer not smaller than z.

e. Calculate the smallest number of iterations k such that $|y^{(k)}| \leq c|y^{(0)}|$, where $0 < c < 1$.

f. Compare the answer of part e with that of part d, focusing on the case where c is very small.

8.6 Suppose that we use the golden section algorithm to find the minimizer of a function. Let u_k be the uncertainty range at the kth iteration. Find the order of convergence of $\{u_k\}$.

8.7 Suppose that we wish to minimize a function $f : \mathbb{R} \to \mathbb{R}$ that has a derivative f'. A simple line search method, called *derivative descent search* (DDS), is described as follows: given that we are at a point $x^{(k)}$, we move in the direction of the negative derivative with step size α; that is, $x^{(k+1)} = x^{(k)} - \alpha f'(x^{(k)})$, where $\alpha > 0$ is a constant.

In the following parts, assume that f is quadratic: $f(x) = \frac{1}{2}ax^2 - bx + c$ (where a, b, and c are constants, and $a > 0$).

a. Write down the value of x^* (in terms of a, b, and c) that minimizes f.

b. Write down the recursive equation for the DDS algorithm explicitly for this quadratic f.

c. Assuming that the DDS algorithm converges, show that it converges to the optimal value x^* (found in part a).

d. Find the order of convergence of the algorithm, assuming that it does converge.

e. Find the range of values of α for which the algorithm converges (for this particular f) for all starting points $x^{(0)}$.

8.8 Consider the function

$$f(\boldsymbol{x}) = 3(x_1^2 + x_2^2) + 4x_1x_2 + 5x_1 + 6x_2 + 7,$$

where $\boldsymbol{x} = [x_1, x_2]^\top \in \mathbb{R}^2$. Suppose that we use a fixed-step-size gradient algorithm to find the minimizer of f:

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha \nabla f(\boldsymbol{x}^{(k)}).$$

Find the largest range of values of α for which the algorithm is globally convergent.

8.9 This exercise explores a zero-finding algorithm.

Suppose that we wish to solve the equation $\boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{0}$, where

$$\boldsymbol{h}(\boldsymbol{x}) = \begin{bmatrix} 4 + 3x_1 + 2x_2 \\ 1 + 2x_1 + 3x_2 \end{bmatrix}.$$

Consider using an algorithm of the form $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha \boldsymbol{h}(\boldsymbol{x}^{(k)})$, where α is scalar constant that does not depend on k.

a. Find the solution of $\boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{0}$.

b. Find the largest range of values of α such that the algorithm is globally convergent to the solution of $\boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{0}$.

c. Assuming that α is outside the range of values in part b, give an example of an initial condition $\boldsymbol{x}^{(0)}$ of the form $[x_1, 0]^\top$ such that the algorithm is guaranteed not to satisfy the descent property.

8.10 Consider the function $f : \mathbb{R}^2 \to \mathbb{R}$ given by

$$f(\boldsymbol{x}) = \frac{3}{2}(x_1^2 + x_2^2) + (1 + a)x_1x_2 - (x_1 + x_2) + b,$$

where a and b are some unknown real-valued parameters.

a. Write the function f in the usual multivariable quadratic form.

b. Find the largest set of values of a and b such that the unique global minimizer of f exists, and write down the minimizer (in terms of the parameters a and b).

c. Consider the following algorithm:

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \frac{2}{5}\nabla f(\boldsymbol{x}^{(k)}).$$

Find the largest set of values of a and b for which this algorithm converges to the global minimizer of f for any initial point $\boldsymbol{x}^{(0)}$.

8.11 Consider the function $f : \mathbb{R} \to \mathbb{R}$ given by $f(x) = \frac{1}{2}(x - c)^2$, $c \in \mathbb{R}$. We are interested in computing the minimizer of f using the iterative algorithm

$$x^{(k+1)} = x^{(k)} - \alpha_k f'(x^{(k)}),$$

where f' is the derivative of f and α_k is a step size satisfying $0 < \alpha_k < 1$.

a. Derive a formula relating $f(x^{(k+1)})$ with $f(x^{(k)})$, involving α_k.

b. Show that the algorithm is globally convergent if and only if

$$\sum_{k=0}^{\infty} \alpha_k = \infty.$$

Hint: Use part a and the fact that for any sequence $\{\alpha_k\} \subset (0, 1)$, we have

$$\prod_{k=0}^{\infty}(1 - \alpha_k) = 0 \Leftrightarrow \sum_{k=0}^{\infty} \alpha_k = \infty.$$

8.12 Consider the function $f : \mathbb{R} \to \mathbb{R}$ given by $f(x) = x^3 - x$. Suppose that we use a fixed-step-size algorithm $x^{(k+1)} = x^{(k)} - \alpha f'(x^{(k)})$ to find a local minimizer of f. Find the largest range of values of α such that the algorithm is locally convergent (i.e., for all x_0 sufficiently close to a local minimizer x^*, we have $x^{(k)} \to x^*$).

8.13 Consider the function f given by $f(x) = (x - 1)^2$, $x \in \mathbb{R}$. We are interested in computing the minimizer of f using the iterative algorithm $x^{(k+1)} = x^{(k)} - \alpha 2^{-k} f'(x^{(k)})$, where f' is the derivative of f and $0 < \alpha < 1$. Does the algorithm have the descent property? Is the algorithm globally convergent?

8.14 Let $f : \mathbb{R} \to \mathbb{R}$, $f \in \mathcal{C}^3$, with first derivative f', second derivative f'', and unique minimizer x^*. Consider a fixed-step-size gradient algorithm

$$x^{(k+1)} = x^{(k)} - \alpha f'(x^{(k)}).$$

Suppose that $f''(x^*) \neq 0$ and $\alpha = 1/f''(x^*)$. Assuming that the algorithm converges to x^*, show that the order of convergence is at least 2.

8.15 Consider the problem of minimizing $f(x) = \|\boldsymbol{a}x - \boldsymbol{b}\|^2$, where $\boldsymbol{a}$ and $\boldsymbol{b}$ are vectors in $\mathbb{R}^n$, and $\boldsymbol{a} \neq \boldsymbol{0}$.

a. Derive an expression (in terms of $\boldsymbol{a}$ and $\boldsymbol{b}$) for the solution to this problem.

b. To solve the problem, suppose that we use an iterative algorithm of the form

$$x^{(k+1)} = x^{(k)} - \alpha f'(x^{(k)}),$$

where f' is the derivative of f. Find the largest range of values of α (in terms of $\boldsymbol{a}$ and $\boldsymbol{b}$) for which the algorithm converges to the solution for all starting points $x^{(0)}$.

8.16 Consider the optimization problem

$$\text{minimize} \quad \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|^2,$$

where $\boldsymbol{A} \in \mathbb{R}^{m \times n}$, $m \geq n$, and $\boldsymbol{b} \in \mathbb{R}^m$.

a. Show that the objective function for this problem is a quadratic function, and write down the gradient and Hessian of this quadratic.

b. Write down the fixed-step-size gradient algorithm for solving this optimization problem.

c. Suppose that

$$\boldsymbol{A} = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}.$$

Find the largest range of values for α such that the algorithm in part b converges to the solution of the problem.

8.17 Consider a function $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^n$ given by $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}$, where $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ and $\boldsymbol{b} \in \mathbb{R}^n$. Suppose that $\boldsymbol{A}$ is invertible and $\boldsymbol{x}^*$ is the zero of $\boldsymbol{f}$ [i.e., $\boldsymbol{f}(\boldsymbol{x}^*) = \boldsymbol{0}$]. We wish to compute $\boldsymbol{x}^*$ using the iterative algorithm

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha \boldsymbol{f}(\boldsymbol{x}^{(k)}),$$

where $\alpha \in \mathbb{R}$, $\alpha > 0$. We say that the algorithm is *globally monotone* if for any $\boldsymbol{x}^{(0)}$, $\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*\| \leq \|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|$ for all k.

a. Assume that all the eigenvalues of $\boldsymbol{A}$ are real. Show that a necessary condition for the algorithm above to be *globally monotone* is that all the eigenvalues of $\boldsymbol{A}$ are nonnegative.
Hint: Use contraposition.

b. Suppose that

$$\boldsymbol{A} = \begin{bmatrix} 3 & 2 \\ 2 & 3 \end{bmatrix}, \qquad \boldsymbol{b} = \begin{bmatrix} 3 \\ -1 \end{bmatrix}.$$

Find the largest range of values of α for which the algorithm is *globally convergent* (i.e., $\boldsymbol{x}^{(k)} \to \boldsymbol{x}^*$ for all $\boldsymbol{x}^{(0)}$).

8.18 Let $f : \mathbb{R}^n \to \mathbb{R}$ be given by $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{Q}\boldsymbol{x} - \boldsymbol{x}^\top \boldsymbol{b}$, where $\boldsymbol{b} \in \mathbb{R}^n$ and $\boldsymbol{Q}$ is a real symmetric positive definite $n \times n$ matrix. Suppose that we apply the steepest descent method to this function, with $\boldsymbol{x}^{(0)} \neq \boldsymbol{Q}^{-1}\boldsymbol{b}$. Show that the method converges in one step, that is, $\boldsymbol{x}^{(1)} = \boldsymbol{Q}^{-1}\boldsymbol{b}$, if and only if $\boldsymbol{x}^{(0)}$ is chosen such that $\boldsymbol{g}^{(0)} = \boldsymbol{Q}\boldsymbol{x}^{(0)} - \boldsymbol{b}$ is an eigenvector of $\boldsymbol{Q}$.

8.19 Suppose that we apply the steepest descent algorithm $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha_k \boldsymbol{g}^{(k)}$ to a quadratic function f with Hessian $\boldsymbol{Q} > 0$. Let $\lambda_{\max}$ and $\lambda_{\min}$ be the largest and smallest eigenvalue of $\boldsymbol{Q}$, respectively. Which of the following two inequalities are possibly true? (When we say here that an inequality is "possibly" true, we mean that there exists a choice of f and $\boldsymbol{x}^{(0)}$ such that the inequality holds.)

a. $\alpha_0 \geq 2/\lambda_{\max}$.

b. $\alpha_0 > 1/\lambda_{\min}$.

8.20 Suppose that we apply a fixed-step-size gradient algorithm to minimize

$$f(\boldsymbol{x}) = \boldsymbol{x}^\top \begin{bmatrix} 3/2 & 2 \\ 0 & 3/2 \end{bmatrix} \boldsymbol{x} + \boldsymbol{x}^\top \begin{bmatrix} 3 \\ -1 \end{bmatrix} - 22.$$

a. Find the range of values of the step size for which the algorithm converges to the minimizer.

b. Suppose that we use a step size of 1000 (which is too large). Find an initial condition that will cause the algorithm to diverge (not converge).

8.21 Consider a fixed-step-size gradient algorithm applied to each of the functions $f : \mathbb{R}^2 \to \mathbb{R}$ in parts a and b below. In each case, find the largest range of values of the step size α for which the algorithm is globally convergent.

a. $f(\boldsymbol{x}) = 1 + 2x_1 + 3(x_1^2 + x_2^2) + 4x_1x_2$.

b. $f(\boldsymbol{x}) = \boldsymbol{x}^\top \begin{bmatrix} 3 & 3 \\ 1 & 3 \end{bmatrix} \boldsymbol{x} + [16, 23]\boldsymbol{x} + \pi^2$.

8.22 Let $f : \mathbb{R}^n \to \mathbb{R}$ be given by $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{Q}\boldsymbol{x} - \boldsymbol{x}^\top \boldsymbol{b}$, where $\boldsymbol{b} \in \mathbb{R}^n$ and $\boldsymbol{Q}$ is a real symmetric positive definite $n \times n$ matrix. Consider the algorithm

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \beta\alpha_k \boldsymbol{g}^{(k)},$$

where $\boldsymbol{g}^{(k)} = \boldsymbol{Q}\boldsymbol{x}^{(k)} - \boldsymbol{b}$, $\alpha_k = \boldsymbol{g}^{(k)\top}\boldsymbol{g}^{(k)}/\boldsymbol{g}^{(k)\top}\boldsymbol{Q}\boldsymbol{g}^{(k)}$, and $\beta \in \mathbb{R}$ is a given constant. (Note that the above reduces to the steepest descent algorithm if $\beta = 1$.) Show that $\{\boldsymbol{x}^{(k)}\}$ converges to $\boldsymbol{x}^* = \boldsymbol{Q}^{-1}\boldsymbol{b}$ for any initial condition $\boldsymbol{x}^{(0)}$ if and only if $0 < \beta < 2$.

8.23 Let $f : \mathbb{R}^n \to \mathbb{R}$ be given by $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{Q}\boldsymbol{x} - \boldsymbol{x}^\top \boldsymbol{b}$, where $\boldsymbol{b} \in \mathbb{R}^n$ and $\boldsymbol{Q}$ is a real symmetric positive definite $n \times n$ matrix. Consider a gradient algorithm

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha_k \boldsymbol{g}^{(k)},$$

where $\boldsymbol{g}^{(k)} = \boldsymbol{Q}\boldsymbol{x}^{(k)} - \boldsymbol{b}$ is the gradient of f at $\boldsymbol{x}^{(k)}$ and α_k is some step size. Show that the algorithm has the descent property [i.e., $f(\boldsymbol{x}^{(k+1)}) < f(\boldsymbol{x}^{(k)})$ whenever $\boldsymbol{g}^{(k)} \neq \boldsymbol{0}$] if and only if $\gamma_k > 0$ for all k.

8.24 Given $f : \mathbb{R}^n \to \mathbb{R}$, consider the general iterative algorithm

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \alpha_k \boldsymbol{d}^{(k)},$$

where $\boldsymbol{d}^{(1)}, \boldsymbol{d}^{(2)}, \ldots$ are given vectors in $\mathbb{R}^n$ and α_k is chosen to minimize $f(\boldsymbol{x}^{(k)} + \alpha\boldsymbol{d}^{(k)})$; that is,

$$\alpha_k = \arg\min f(\boldsymbol{x}^{(k)} + \alpha\boldsymbol{d}^{(k)}).$$

Show that for each k, the vector $\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)}$ is orthogonal to $\nabla f(\boldsymbol{x}^{(k+1)})$ (assuming that the gradient exists).

8.25 Write a simple MATLAB program for implementing the steepest descent algorithm using the secant method for the line search (e.g., the MATLAB function of Exercise 7.11). For the stopping criterion, use the condition $\|\boldsymbol{g}^{(k)}\| \leq \varepsilon$, where $\varepsilon = 10^{-6}$. Test your program by comparing the output with the numbers in Example 8.1. Also test your program using an initial condition of $[-4, 5, 1]^\top$, and determine the number of iterations required to satisfy the stopping criterion. Evaluate the objective function at the final point to see how close it is to 0.

8.26 Apply the MATLAB program from Exercise 8.25 to Rosenbrock's function:

$$f(\boldsymbol{x}) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2.$$

Use an initial condition of $\boldsymbol{x}^{(0)} = [-2, 2]^\top$. Terminate the algorithm when the norm of the gradient of f is less than 10^{-4}.

CHAPTER 9

NEWTON'S METHOD

9.1 Introduction

Recall that the method of steepest descent uses only first derivatives (gradients) in selecting a suitable search direction. This strategy is not always the most effective. If higher derivatives are used, the resulting iterative algorithm may perform better than the steepest descent method. *Newton's method* (sometimes called the *Newton-Raphson method*) uses first and second derivatives and indeed does perform better than the steepest descent method if the initial point is close to the minimizer. The idea behind this method is as follows. Given a starting point, we construct a quadratic approximation to the objective function that matches the first and second derivative values at that point. We then minimize the approximate (quadratic) function instead of the original objective function. We use the minimizer of the approximate function as the starting point in the next step and repeat the procedure iteratively. If the objective function is quadratic, then the approximation is exact, and the method yields the true minimizer in one step. If, on the other hand, the objective function is not quadratic, then the approximation will provide

An Introduction to Optimization, Fourth Edition.
By E. K. P. Chong and S. H. Żak.

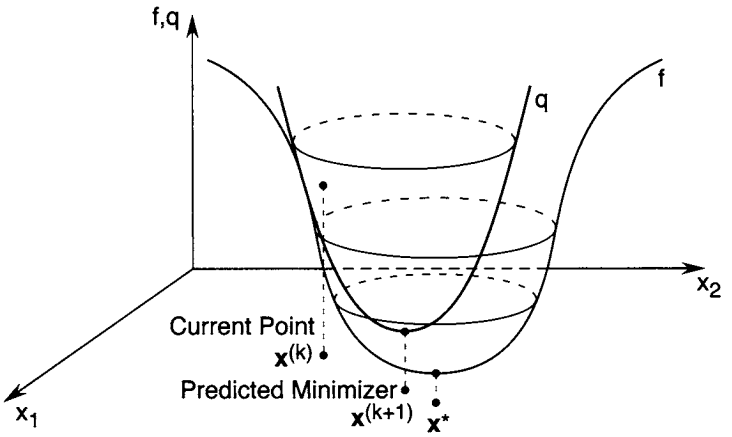

Figure 9.1 Quadratic approximation to the objective function using first and second derivatives.

only an estimate of the position of the true minimizer. Figure 9.1 illustrates this idea.

We can obtain a quadratic approximation to the twice continuously differentiable objection function $f : \mathbb{R}^n \to \mathbb{R}$ using the Taylor series expansion of f about the current point $\boldsymbol{x}^{(k)}$, neglecting terms of order three and higher. We obtain

$$f(\boldsymbol{x}) \approx f(\boldsymbol{x}^{(k)}) + (\boldsymbol{x} - \boldsymbol{x}^{(k)})^\top \boldsymbol{g}^{(k)} + \frac{1}{2}(\boldsymbol{x} - \boldsymbol{x}^{(k)})^\top \boldsymbol{F}(\boldsymbol{x}^{(k)})(\boldsymbol{x} - \boldsymbol{x}^{(k)}) \triangleq q(\boldsymbol{x}),$$

where, for simplicity, we use the notation $\boldsymbol{g}^{(k)} = \nabla f(\boldsymbol{x}^{(k)})$. Applying the FONC to q yields

$$\mathbf{0} = \nabla q(\boldsymbol{x}) = \boldsymbol{g}^{(k)} + \boldsymbol{F}(\boldsymbol{x}^{(k)})(\boldsymbol{x} - \boldsymbol{x}^{(k)}).$$

If $\boldsymbol{F}(\boldsymbol{x}^{(k)}) > 0$, then q achieves a minimum at

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \boldsymbol{F}(\boldsymbol{x}^{(k)})^{-1}\boldsymbol{g}^{(k)}.$$

This recursive formula represents Newton's method.

Example 9.1 Use Newton's method to minimize the Powell function:

$$f(x_1, x_2, x_3, x_4) = (x_1 + 10x_2)^2 + 5(x_3 - x_4)^2 + (x_2 - 2x_3)^4 + 10(x_1 - x_4)^4.$$

Use as the starting point $\boldsymbol{x}^{(0)} = [3, -1, 0, 1]^\top$. Perform three iterations.

Note that $f(\boldsymbol{x}^{(0)}) = 215$. We have

$$\nabla f(\boldsymbol{x}) = \begin{bmatrix} 2(x_1 + 10x_2) + 40(x_1 - x_4)^3 \\ 20(x_1 + 10x_2) + 4(x_2 - 2x_3)^3 \\ 10(x_3 - x_4) - 8(x_2 - 2x_3)^3 \\ -10(x_3 - x_4) - 40(x_1 - x_4)^3 \end{bmatrix},$$

and $\boldsymbol{F}(\boldsymbol{x})$ is given by

$$\begin{bmatrix} 2+120(x_1-x_4)^2 & 20 & 0 & -120(x_1-x_4)^2 \\ 20 & 200+12(x_2-2x_3)^2 & -24(x_2-2x_3)^2 & 0 \\ 0 & -24(x_2-2x_3)^2 & 10+48(x_2-2x_3)^2 & -10 \\ -120(x_1-x_4)^2 & 0 & -10 & 10+120(x_1-x_4)^2 \end{bmatrix}.$$

Iteration 1

$$\boldsymbol{g}^{(0)} = [306, -144, -2, -310]^\top,$$

$$\boldsymbol{F}(\boldsymbol{x}^{(0)}) = \begin{bmatrix} 482 & 20 & 0 & -480 \\ 20 & 212 & -24 & 0 \\ 0 & -24 & 58 & -10 \\ -480 & 0 & -10 & 490 \end{bmatrix},$$

$$\boldsymbol{F}(\boldsymbol{x}^{(0)})^{-1} = \begin{bmatrix} 0.1126 & -0.0089 & 0.0154 & 0.1106 \\ -0.0089 & 0.0057 & 0.0008 & -0.0087 \\ 0.0154 & 0.0008 & 0.0203 & 0.0155 \\ 0.1106 & -0.0087 & 0.0155 & 0.1107 \end{bmatrix},$$

$$\boldsymbol{F}(\boldsymbol{x}^{(0)})^{-1}\boldsymbol{g}^{(0)} = [1.4127, -0.8413, -0.2540, 0.7460]^\top.$$

Hence,

$$\boldsymbol{x}^{(1)} = \boldsymbol{x}^{(0)} - \boldsymbol{F}(\boldsymbol{x}^{(0)})^{-1}\boldsymbol{g}^{(0)} = [1.5873, -0.1587, 0.2540, 0.2540]^\top,$$
$$f(\boldsymbol{x}^{(1)}) = 31.8.$$

Iteration 2

$$\boldsymbol{g}^{(1)} = [94.81, -1.179, 2.371, -94.81]^\top,$$

$$\boldsymbol{F}(\boldsymbol{x}^{(1)}) = \begin{bmatrix} 215.3 & 20 & 0 & -213.3 \\ 20 & 205.3 & -10.67 & 0 \\ 0 & -10.67 & 31.34 & -10 \\ -213.3 & 0 & -10 & 223.3 \end{bmatrix},$$

$$\boldsymbol{F}(\boldsymbol{x}^{(1)})^{-1}\boldsymbol{g}^{(1)} = [0.5291, -0.0529, 0.0846, 0.0846]^\top.$$

Hence,

$$\boldsymbol{x}^{(2)} = \boldsymbol{x}^{(1)} - \boldsymbol{F}(\boldsymbol{x}^{(1)})^{-1}\boldsymbol{g}^{(1)} = [1.0582, -0.1058, 0.1694, 0.1694]^\top,$$
$$f(\boldsymbol{x}^{(2)}) = 6.28.$$

Iteration 3

$$\begin{aligned}
\boldsymbol{g}^{(2)} &= [28.09, -0.3475, 0.7031, -28.08]^\top, \\
\boldsymbol{F}(\boldsymbol{x}^{(2)}) &= \begin{bmatrix} 96.80 & 20 & 0 & -94.80 \\ 20 & 202.4 & -4.744 & 0 \\ 0 & -4.744 & 19.49 & -10 \\ -94.80 & 0 & -10 & 104.80 \end{bmatrix}, \\
\boldsymbol{x}^{(3)} &= [0.7037, -0.0704, 0.1121, 0.1111]^\top, \\
f(\boldsymbol{x}^{(3)}) &= 1.24.
\end{aligned}$$

■

Observe that the kth iteration of Newton's method can be written in two steps as

1. Solve $\boldsymbol{F}(\boldsymbol{x}^{(k)})\boldsymbol{d}^{(k)} = -\boldsymbol{g}^{(k)}$ for $\boldsymbol{d}^{(k)}$.
2. Set $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \boldsymbol{d}^{(k)}$.

Step 1 requires the solution of an $n \times n$ system of linear equations. Thus, an efficient method for solving systems of linear equations is essential when using Newton's method.

As in the one-variable case, Newton's method can also be viewed as a technique for iteratively solving the equation

$$\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{0},$$

where $\boldsymbol{x} \in \mathbb{R}^n$ and $\boldsymbol{g} : \mathbb{R}^n \to \mathbb{R}^n$. In this case $\boldsymbol{F}(\boldsymbol{x})$ is the Jacobian matrix of $\boldsymbol{g}$ at $\boldsymbol{x}$; that is, $\boldsymbol{F}(\boldsymbol{x})$ is the $n \times n$ matrix whose (i, j) entry is $(\partial g_i/\partial x_j)(\boldsymbol{x})$, $i, j = 1, 2, \ldots, n$.

9.2 Analysis of Newton's Method

As in the one-variable case there is no guarantee that Newton's algorithm heads in the direction of decreasing values of the objective function if $\boldsymbol{F}(\boldsymbol{x}^{(k)})$ is not positive definite (recall Figure 7.7 illustrating Newton's method for functions of one variable when $f'' < 0$). Moreover, even if $\boldsymbol{F}(\boldsymbol{x}^{(k)}) > 0$, Newton's method may not be a descent method; that is, it is possible that $f(\boldsymbol{x}^{(k+1)}) \geq f(\boldsymbol{x}^{(k)})$. For example, this may occur if our starting point $\boldsymbol{x}^{(0)}$ is far away from the solution. See the end of this section for a possible remedy to this problem. Despite these drawbacks, Newton's method has superior convergence properties when the starting point is near the solution, as we shall see in the remainder of this section.

The convergence analysis of Newton's method when f is a quadratic function is straightforward. In fact, Newton's method reaches the point $\boldsymbol{x}^*$ such

that $\nabla f(\boldsymbol{x}^*) = \mathbf{0}$ in just one step starting from any initial point $\boldsymbol{x}^{(0)}$. To see this, suppose that $\boldsymbol{Q} = \boldsymbol{Q}^\top$ is invertible and

$$f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{Q}\boldsymbol{x} - \boldsymbol{x}^\top \boldsymbol{b}.$$

Then,

$$\boldsymbol{g}(\boldsymbol{x}) = \nabla f(\boldsymbol{x}) = \boldsymbol{Q}\boldsymbol{x} - \boldsymbol{b}$$

and

$$\boldsymbol{F}(\boldsymbol{x}) = \boldsymbol{Q}.$$

Hence, given any initial point $\boldsymbol{x}^{(0)}$, by Newton's algorithm

$$\begin{aligned}\boldsymbol{x}^{(1)} &= \boldsymbol{x}^{(0)} - \boldsymbol{F}(\boldsymbol{x}^{(0)})^{-1}\boldsymbol{g}^{(0)} \\ &= \boldsymbol{x}^{(0)} - \boldsymbol{Q}^{-1}[\boldsymbol{Q}\boldsymbol{x}^{(0)} - \boldsymbol{b}] \\ &= \boldsymbol{Q}^{-1}\boldsymbol{b} \\ &= \boldsymbol{x}^*.\end{aligned}$$

Therefore, for the quadratic case the order of convergence of Newton's algorithm is ∞ for any initial point $\boldsymbol{x}^{(0)}$ (compare this with Exercise 8.18, which deals with the steepest descent algorithm).

To analyze the convergence of Newton's method in the general case, we use results from Section 5.1. Let $\{\boldsymbol{x}^{(k)}\}$ be the Newton's method sequence for minimizing a function $f : \mathbb{R}^n \to \mathbb{R}$. We show that $\{\boldsymbol{x}^{(k)}\}$ converges to the minimizer $\boldsymbol{x}^*$ with order of convergence at least 2.

Theorem 9.1 *Suppose that $f \in \mathcal{C}^3$ and $\boldsymbol{x}^* \in \mathbb{R}^n$ is a point such that $\nabla f(\boldsymbol{x}^*) = \mathbf{0}$ and $\boldsymbol{F}(\boldsymbol{x}^*)$ is invertible. Then, for all $\boldsymbol{x}^{(0)}$ sufficiently close to $\boldsymbol{x}^*$, Newton's method is well-defined for all k and converges to $\boldsymbol{x}^*$ with an order of convergence at least 2.* □

Proof. The Taylor series expansion of ∇f about $\boldsymbol{x}^{(0)}$ yields

$$\nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{x}^{(0)}) - \boldsymbol{F}(\boldsymbol{x}^{(0)})(\boldsymbol{x} - \boldsymbol{x}^{(0)}) = O(\|\boldsymbol{x} - \boldsymbol{x}^{(0)}\|^2).$$

Because by assumption $f \in \mathcal{C}^3$ and $\boldsymbol{F}(\boldsymbol{x}^*)$ is invertible, there exist constants $\varepsilon > 0$, $c_1 > 0$, and $c_2 > 0$ such that if $\boldsymbol{x}^{(0)}, \boldsymbol{x} \in \{\boldsymbol{x} : \|\boldsymbol{x} - \boldsymbol{x}^*\| \leq \varepsilon\}$, we have

$$\|\nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{x}^{(0)}) - \boldsymbol{F}(\boldsymbol{x}^{(0)})(\boldsymbol{x} - \boldsymbol{x}^{(0)})\| \leq c_1 \|\boldsymbol{x} - \boldsymbol{x}^{(0)}\|^2$$

and by Lemma 5.3, $\boldsymbol{F}(\boldsymbol{x})^{-1}$ exists and satisfies

$$\|\boldsymbol{F}(\boldsymbol{x})^{-1}\| \leq c_2.$$

The first inequality above holds because the remainder term in the Taylor series expansion contains third derivatives of f that are continuous and hence bounded on $\{\boldsymbol{x} : \|\boldsymbol{x} - \boldsymbol{x}^*\| \leq \varepsilon\}$.

Suppose that $\boldsymbol{x}^{(0)} \in \{\boldsymbol{x} : \|\boldsymbol{x} - \boldsymbol{x}^*\| \leq \varepsilon\}$. Then, substituting $\boldsymbol{x} = \boldsymbol{x}^*$ in the inequality above and using the assumption that $\nabla f(\boldsymbol{x}^*) = \mathbf{0}$, we get

$$\|\boldsymbol{F}(\boldsymbol{x}^{(0)})(\boldsymbol{x}^{(0)} - \boldsymbol{x}^*) - \nabla f(\boldsymbol{x}^{(0)})\| \leq c_1 \|\boldsymbol{x}^{(0)} - \boldsymbol{x}^*\|^2.$$

Now, subtracting $\boldsymbol{x}^*$ from both sides of Newton's algorithm and taking norms yields

$$\begin{aligned}
\|\boldsymbol{x}^{(1)} - \boldsymbol{x}^*\| &= \|\boldsymbol{x}^{(0)} - \boldsymbol{x}^* - \boldsymbol{F}(\boldsymbol{x}^{(0)})^{-1} \nabla f(\boldsymbol{x}^{(0)})\| \\
&= \|\boldsymbol{F}(\boldsymbol{x}^{(0)})^{-1} (\boldsymbol{F}(\boldsymbol{x}^{(0)})(\boldsymbol{x}^{(0)} - \boldsymbol{x}^*) - \nabla f(\boldsymbol{x}^{(0)}))\| \\
&\leq \|\boldsymbol{F}(\boldsymbol{x}^{(0)})^{-1}\| \|\boldsymbol{F}(\boldsymbol{x}^{(0)})(\boldsymbol{x}^{(0)} - \boldsymbol{x}^*) - \nabla f(\boldsymbol{x}^{(0)})\|.
\end{aligned}$$

Applying the inequalities above involving the constants c_1 and c_2 gives

$$\|\boldsymbol{x}^{(1)} - \boldsymbol{x}^*\| \leq c_1 c_2 \|\boldsymbol{x}^{(0)} - \boldsymbol{x}^*\|^2.$$

Suppose that $\boldsymbol{x}^{(0)}$ is such that

$$\|\boldsymbol{x}^{(0)} - \boldsymbol{x}^*\| \leq \frac{\alpha}{c_1 c_2},$$

where $\alpha \in (0, 1)$. Then,

$$\|\boldsymbol{x}^{(1)} - \boldsymbol{x}^*\| \leq \alpha \|\boldsymbol{x}^{(0)} - \boldsymbol{x}^*\|.$$

By induction, we obtain

$$\begin{aligned}
\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*\| &\leq c_1 c_2 \|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|^2, \\
\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*\| &\leq \alpha \|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|.
\end{aligned}$$

Hence,

$$\lim_{k \to \infty} \|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\| = 0,$$

and therefore the sequence $\{\boldsymbol{x}^{(k)}\}$ converges to $\boldsymbol{x}^*$. The order of convergence is at least 2 because $\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*\| \leq c_1 c_2 \|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|^2$; that is, $\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*\| = O(\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|^2)$. ∎

Warning: In the Theorem 9.1, we did not state that $\boldsymbol{x}^*$ is a local minimizer. For example, if $\boldsymbol{x}^*$ is a local *maximizer*, then provided that $f \in \mathcal{C}^3$ and $\boldsymbol{F}(\boldsymbol{x}^*)$ is invertible, Newton's method would converge to $\boldsymbol{x}^*$ if we start close enough to it.

As stated in Theorem 9.1, Newton's method has superior convergence properties if the starting point is near the solution. However, the method is not guaranteed to converge to the solution if we start far away from it (in fact, it may not even be well-defined because the Hessian may be singular). In particular, the method may not be a descent method; that is, it is possible that

$f(\boldsymbol{x}^{(k+1)}) \geq f(\boldsymbol{x}^{(k)})$. Fortunately, it is possible to modify the algorithm such that the descent property holds. To see this, we need the following result.

Theorem 9.2 *Let $\{\boldsymbol{x}^{(k)}\}$ be the sequence generated by Newton's method for minimizing a given objective function $f(\boldsymbol{x})$. If the Hessian $\boldsymbol{F}(\boldsymbol{x}^{(k)}) > 0$ and $\boldsymbol{g}^{(k)} = \nabla f(\boldsymbol{x}^{(k)}) \neq \boldsymbol{0}$, then the search direction*

$$\boldsymbol{d}^{(k)} = -\boldsymbol{F}(\boldsymbol{x}^{(k)})^{-1}\boldsymbol{g}^{(k)} = \boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)}$$

from $\boldsymbol{x}^{(k)}$ to $\boldsymbol{x}^{(k+1)}$ is a descent direction for f in the sense that there exists an $\bar{\alpha} > 0$ such that for all $\alpha \in (0, \bar{\alpha})$,

$$f(\boldsymbol{x}^{(k)} + \alpha \boldsymbol{d}^{(k)}) < f(\boldsymbol{x}^{(k)}).$$

□

Proof. Let

$$\phi(\alpha) = f(\boldsymbol{x}^{(k)} + \alpha \boldsymbol{d}^{(k)}).$$

Then, using the chain rule, we obtain

$$\phi'(\alpha) = \nabla f(\boldsymbol{x}^{(k)} + \alpha \boldsymbol{d}^{(k)})^{\top} \boldsymbol{d}^{(k)}.$$

Hence,

$$\phi'(0) = \nabla f(\boldsymbol{x}^{(k)})^{\top} \boldsymbol{d}^{(k)} = -\boldsymbol{g}^{(k)\top} \boldsymbol{F}(\boldsymbol{x}^{(k)})^{-1} \boldsymbol{g}^{(k)} < 0,$$

because $\boldsymbol{F}(\boldsymbol{x}^{(k)})^{-1} > 0$ and $\boldsymbol{g}^{(k)} \neq \boldsymbol{0}$. Thus, there exists an $\bar{\alpha} > 0$ so that for all $\alpha \in (0, \bar{\alpha})$, $\phi(\alpha) < \phi(0)$. This implies that for all $\alpha \in (0, \bar{\alpha})$,

$$f(\boldsymbol{x}^{(k)} + \alpha \boldsymbol{d}^{(k)}) < f(\boldsymbol{x}^{(k)}),$$

which completes the proof. ∎

Theorem 9.2 motivates the following modification of Newton's method:

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha_k \boldsymbol{F}(\boldsymbol{x}^{(k)})^{-1} \boldsymbol{g}^{(k)},$$

where

$$\alpha_k = \arg\min_{\alpha \geq 0} f(\boldsymbol{x}^{(k)} - \alpha \boldsymbol{F}(\boldsymbol{x}^{(k)})^{-1} \boldsymbol{g}^{(k)});$$

that is, at each iteration, we perform a line search in the direction $-\boldsymbol{F}(\boldsymbol{x}^{(k)})^{-1}\boldsymbol{g}^{(k)}$. By Theorem 9.2 we conclude that the modified Newton's method has the descent property; that is,

$$f(\boldsymbol{x}^{(k+1)}) < f(\boldsymbol{x}^{(k)})$$

whenever $\boldsymbol{g}^{(k)} \neq \boldsymbol{0}$.

A drawback of Newton's method is that evaluation of $\boldsymbol{F}(\boldsymbol{x}^{(k)})$ for large n can be computationally expensive. Furthermore, we have to solve the set of n linear equations $\boldsymbol{F}(\boldsymbol{x}^{(k)})\boldsymbol{d}^{(k)} = -\boldsymbol{g}^{(k)}$. In Chapters 10 and 11 we discuss methods that alleviate this difficulty.

Another source of potential problems in Newton's method arises from the Hessian matrix not being positive definite. In the next section we describe a simple modification of Newton's method to overcome this problem.

9.3 Levenberg-Marquardt Modification

If the Hessian matrix $\boldsymbol{F}(\boldsymbol{x}^{(k)})$ is not positive definite, then the search direction $\boldsymbol{d}^{(k)} = -\boldsymbol{F}(\boldsymbol{x}^{(k)})^{-1}\boldsymbol{g}^{(k)}$ may not point in a descent direction. A simple technique to ensure that the search direction is a descent direction is to introduce the *Levenberg-Marquardt modification* of Newton's algorithm:

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - (\boldsymbol{F}(\boldsymbol{x}^{(k)}) + \mu_k \boldsymbol{I})^{-1}\boldsymbol{g}^{(k)},$$

where $\mu_k \geq 0$.

The idea underlying the Levenberg-Marquardt modification is as follows. Consider a symmetric matrix $\boldsymbol{F}$, which may not be positive definite. Let $\lambda_1, \ldots, \lambda_n$ be the eigenvalues of $\boldsymbol{F}$ with corresponding eigenvectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$. The eigenvalues $\lambda_1, \ldots, \lambda_n$ are real, but may not all be positive. Next, consider the matrix $\boldsymbol{G} = \boldsymbol{F} + \mu \boldsymbol{I}$, where $\mu \geq 0$. Note that the eigenvalues of $\boldsymbol{G}$ are $\lambda_1 + \mu, \ldots, \lambda_n + \mu$. Indeed,

$$\begin{aligned}
\boldsymbol{G}\boldsymbol{v}_i &= (\boldsymbol{F} + \mu \boldsymbol{I})\boldsymbol{v}_i \\
&= \boldsymbol{F}\boldsymbol{v}_i + \mu \boldsymbol{I}\boldsymbol{v}_i \\
&= \lambda_i \boldsymbol{v}_i + \mu \boldsymbol{v}_i \\
&= (\lambda_i + \mu)\boldsymbol{v}_i,
\end{aligned}$$

which shows that for all $i = 1, \ldots, n$, $\boldsymbol{v}_i$ is also an eigenvector of $\boldsymbol{G}$ with eigenvalue $\lambda_i + \mu$. Therefore, if μ is sufficiently large, then all the eigenvalues of $\boldsymbol{G}$ are positive and $\boldsymbol{G}$ is positive definite. Accordingly, if the parameter μ_k in the Levenberg-Marquardt modification of Newton's algorithm is sufficiently large, then the search direction $\boldsymbol{d}^{(k)} = -(\boldsymbol{F}(\boldsymbol{x}^{(k)}) + \mu_k \boldsymbol{I})^{-1}\boldsymbol{g}^{(k)}$ always points in a descent direction (in the sense of Theorem 9.2). In this case if we further introduce a step size α_k as described in Section 9.2,

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha_k(\boldsymbol{F}(\boldsymbol{x}^{(k)}) + \mu_k \boldsymbol{I})^{-1}\boldsymbol{g}^{(k)},$$

then we are guaranteed that the descent property holds.

The Levenberg-Marquardt modification of Newton's algorithm can be made to approach the behavior of the pure Newton's method by letting $\mu_k \to 0$. On the other hand, by letting $\mu_k \to \infty$, the algorithm approaches a pure gradient method with small step size. In practice, we may start with a small value of μ_k and increase it slowly until we find that the iteration is descent: $f(\boldsymbol{x}^{(k+1)}) < f(\boldsymbol{x}^{(k)})$.

9.4 Newton's Method for Nonlinear Least Squares

We now examine a particular class of optimization problems and the use of Newton's method for solving them. Consider the following problem:

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

where $r_i : \mathbb{R}^n \to \mathbb{R}$, $i = 1, \ldots, m$, are given functions. This particular problem is called a *nonlinear least-squares problem.* The special case where the r_i are linear is discussed in Section 12.1.

Example 9.2 Suppose that we are given m measurements of a process at m points in time, as depicted in Figure 9.2 (here, $m = 21$). Let $t_1, \ldots, t_m$ denote the measurement times and $y_1, \ldots, y_m$ the measurement values. Note that $t_1 = 0$ while $t_{21} = 10$. We wish to fit a sinusoid to the measurement data. The equation of the sinusoid is

$$y = A \sin(\omega t + \phi)$$

with appropriate choices of the parameters A, ω, and ϕ. To formulate the data-fitting problem, we construct the objective function

$$\sum_{i=1}^{m} (y_i - A \sin(\omega t_i + \phi))^2,$$

representing the sum of the squared errors between the measurement values and the function values at the corresponding points in time. Let $\boldsymbol{x} = [A, \omega, \phi]^\top$ represent the vector of decision variables. We therefore obtain a nonlinear least-squares problem with

$$r_i(\boldsymbol{x}) = y_i - A \sin(\omega t_i + \phi).$$

∎

Defining $\boldsymbol{r} = [r_1, \ldots, r_m]^\top$, we write the objective function as $f(\boldsymbol{x}) = \boldsymbol{r}(\boldsymbol{x})^\top \boldsymbol{r}(\boldsymbol{x})$. To apply Newton's method, we need to compute the gradient and the Hessian of f. The jth component of $\nabla f(\boldsymbol{x})$ is

$$(\nabla f(\boldsymbol{x}))_j = \frac{\partial f}{\partial x_j}(\boldsymbol{x}) = 2 \sum_{i=1}^{m} r_i(\boldsymbol{x}) \frac{\partial r_i}{\partial x_j}(\boldsymbol{x}).$$

Denote the Jacobian matrix of $\boldsymbol{r}$ by

$$\boldsymbol{J}(\boldsymbol{x}) = \begin{bmatrix} \frac{\partial r_1}{\partial x_1}(\boldsymbol{x}) & \cdots & \frac{\partial r_1}{\partial x_n}(\boldsymbol{x}) \\ \vdots & & \vdots \\ \frac{\partial r_m}{\partial x_1}(\boldsymbol{x}) & \cdots & \frac{\partial r_m}{\partial x_n}(\boldsymbol{x}) \end{bmatrix}.$$

Then, the gradient of f can be represented as

$$\nabla f(\boldsymbol{x}) = 2 \boldsymbol{J}(\boldsymbol{x})^\top \boldsymbol{r}(\boldsymbol{x}).$$

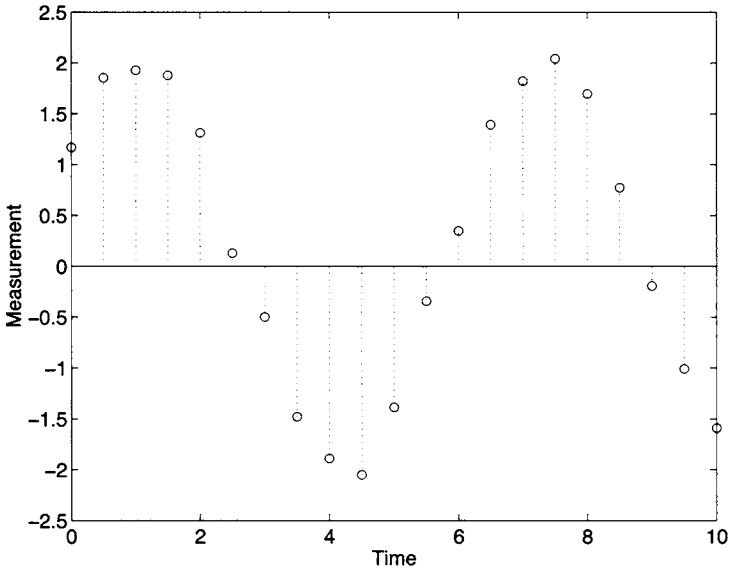

Figure 9.2 Measurement data for Example 9.2.

Next, we compute the Hessian matrix of f. The (k,j)th component of the Hessian is given by

$$\begin{aligned}\frac{\partial^2 f}{\partial x_k \partial x_j}(\boldsymbol{x}) &= \frac{\partial}{\partial x_k}\left(\frac{\partial f}{\partial x_j}(\boldsymbol{x})\right) \\ &= \frac{\partial}{\partial x_k}\left(2\sum_{i=1}^{m} r_i(\boldsymbol{x})\frac{\partial r_i}{\partial x_j}(\boldsymbol{x})\right) \\ &= 2\sum_{i=1}^{m}\left(\frac{\partial r_i}{\partial x_k}(\boldsymbol{x})\frac{\partial r_i}{\partial x_j}(\boldsymbol{x}) + r_i(\boldsymbol{x})\frac{\partial^2 r_i}{\partial x_k \partial x_j}(\boldsymbol{x})\right).\end{aligned}$$

Letting $\boldsymbol{S}(\boldsymbol{x})$ be the matrix whose (k,j)th component is

$$\sum_{i=1}^{m} r_i(\boldsymbol{x})\frac{\partial^2 r_i}{\partial x_k \partial x_j}(\boldsymbol{x}),$$

we write the Hessian matrix as

$$\boldsymbol{F}(\boldsymbol{x}) = 2(\boldsymbol{J}(\boldsymbol{x})^\top \boldsymbol{J}(\boldsymbol{x}) + \boldsymbol{S}(\boldsymbol{x})).$$

Therefore, Newton's method applied to the nonlinear least-squares problem is given by

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - (\boldsymbol{J}(\boldsymbol{x})^\top \boldsymbol{J}(\boldsymbol{x}) + \boldsymbol{S}(\boldsymbol{x}))^{-1}\boldsymbol{J}(\boldsymbol{x})^\top \boldsymbol{r}(\boldsymbol{x}).$$

In some applications, the matrix $\boldsymbol{S}(\boldsymbol{x})$ involving the second derivatives of the function $\boldsymbol{r}$ can be ignored because its components are negligibly small. In this case Newton's algorithm reduces to what is commonly called the *Gauss-Newton method*:

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - (\boldsymbol{J}(\boldsymbol{x})^\top \boldsymbol{J}(\boldsymbol{x}))^{-1} \boldsymbol{J}(\boldsymbol{x})^\top \boldsymbol{r}(\boldsymbol{x}).$$

Note that the Gauss-Newton method does not require calculation of the second derivatives of $\boldsymbol{r}$.

Example 9.3 Recall the data-fitting problem in Example 9.2, with

$$r_i(\boldsymbol{x}) = y_i - A\sin(\omega t_i + \phi), \qquad i = 1, \ldots, 21.$$

The Jacobian matrix $J(\boldsymbol{x})$ in this problem is a 21×3 matrix with elements given by

$$\begin{aligned}
(J(\boldsymbol{x}))_{(i,1)} &= -\sin(\omega t_i + \phi), \\
(J(\boldsymbol{x}))_{(i,2)} &= -t_i A\cos(\omega t_i + \phi), \\
(J(\boldsymbol{x}))_{(i,3)} &= -A\cos(\omega t_i + \phi), \qquad i = 1, \ldots, 21.
\end{aligned}$$

Using the expressions above, we apply the Gauss-Newton algorithm to find the sinusoid of best fit, given the data pairs $(t_1, y_1), \ldots, (t_m, y_m)$. Figure 9.3 shows a plot of the sinusoid of best fit obtained from the Gauss-Newton algorithm. The parameters of this sinusoid are: $A = 2.01$, $\omega = 0.992$, and $\phi = 0.541$. ∎

A potential problem with the Gauss-Newton method is that the matrix $\boldsymbol{J}(\boldsymbol{x})^\top \boldsymbol{J}(\boldsymbol{x})$ may not be positive definite. As described before, this problem can be overcome using a Levenberg-Marquardt modification:

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - (\boldsymbol{J}(\boldsymbol{x})^\top \boldsymbol{J}(\boldsymbol{x}) + \mu_k \boldsymbol{I})^{-1} \boldsymbol{J}(\boldsymbol{x})^\top \boldsymbol{r}(\boldsymbol{x}).$$

This is referred to in the literature as the *Levenberg-Marquardt algorithm*, because the original Levenberg-Marquardt modification was developed specifically for the nonlinear least-squares problem. An alternative interpretation of the Levenberg-Marquardt algorithm is to view the term $\mu_k \boldsymbol{I}$ as an approximation to $\boldsymbol{S}(\boldsymbol{x})$ in Newton's algorithm.

EXERCISES

9.1 Let $f : \mathbb{R} \to \mathbb{R}$ be given by $f(x) = (x - x_0)^4$, where $x_0 \in \mathbb{R}$ is a constant. Suppose that we apply Newton's method to the problem of minimizing f.

a. Write down the update equation for Newton's method applied to the problem.

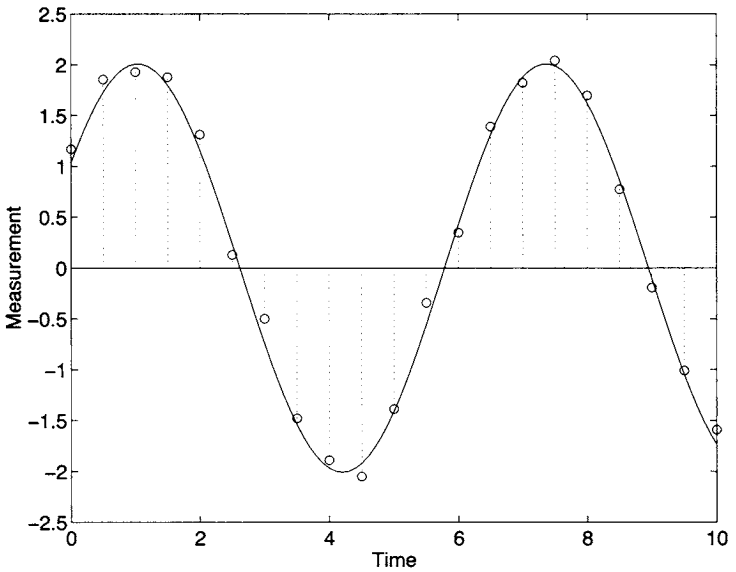

Figure 9.3 Sinusoid of best fit in Example 9.3.

b. Let $y^{(k)} = |x^{(k)} - x_0|$, where $x^{(k)}$ is the kth iterate in Newton's method. Show that the sequence $\{y^{(k)}\}$ satisfies $y^{(k+1)} = \frac{2}{3}y^{(k)}$.

c. Show that $x^{(k)} \to x_0$ for any initial guess $x^{(0)}$.

d. Show that the order of convergence of the sequence $\{x^{(k)}\}$ in part b is 1.

e. Theorem 9.1 states that under certain conditions, the order of convergence of Newton's method is at least 2. Why does that theorem not hold in this particular problem?

9.2 This question relates to the order of convergence of the secant method, using an argument similar to that of the proof of Theorem 9.1.

a. Consider a function $f : \mathbb{R} \to \mathbb{R}$, $f \in \mathcal{C}^2$, such that x^* is a local minimizer and $f''(x^*) \neq 0$. Suppose that we apply the algorithm $x^{(k+1)} = x^{(k)} - \alpha_k f'(x^{(k)})$ such that $\{\alpha_k\}$ is a positive step-size sequence that converges to $1/f''(x^*)$. Show that if $x^{(k)} \to x^*$, then the order of convergence of the algorithm is *superlinear* (i.e., strictly greater than 1).

b. Given part a, what can you say about the order of convergence of the secant algorithm?

9.3 Consider the problem of minimizing $f(x) = x^{\frac{4}{3}} = (\sqrt[3]{x})^4$, $x \in \mathbb{R}$. Note that 0 is the global minimizer of f.

a. Write down the algorithm for Newton's method applied to this problem.

b. Show that as long as the starting point is not 0, the algorithm in part a does not converge to 0 (no matter how close to 0 we start).

9.4 Consider *Rosenbrock's Function*: $f(\boldsymbol{x}) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$, where $\boldsymbol{x} = [x_1, x_2]^\top$ (known to be a "nasty" function—often used as a benchmark for testing algorithms). This function is also known as the *banana function* because of the shape of its level sets.

a. Prove that $[1, 1]^\top$ is the unique global minimizer of f over $\mathbb{R}^2$.

b. With a starting point of $[0, 0]^\top$, apply two iterations of Newton's method.

Hint: $\begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \dfrac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$.

c. Repeat part b using a gradient algorithm with a fixed step size of $\alpha_k = 0.05$ at each iteration.

9.5 Consider the modified Newton's algorithm

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha_k \boldsymbol{F}(\boldsymbol{x}^{(k)})^{-1} \boldsymbol{g}^{(k)},$$

where $\alpha_k = \arg\min_{\alpha \geq 0} f(\boldsymbol{x}^{(k)} - \alpha \boldsymbol{F}(\boldsymbol{x}^{(k)})^{-1} \boldsymbol{g}^{(k)})$. Suppose that we apply the algorithm to a quadratic function $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{Q}\boldsymbol{x} - \boldsymbol{x}^\top \boldsymbol{b}$, where $\boldsymbol{Q} = \boldsymbol{Q}^\top > 0$. Recall that the standard Newton's method reaches point $\boldsymbol{x}^*$ such that $\nabla f(\boldsymbol{x}^*) = \boldsymbol{0}$ in just one step starting from any initial point $\boldsymbol{x}^{(0)}$. Does the modified Newton's algorithm above possess the same property?

CHAPTER 10

CONJUGATE DIRECTION METHODS

10.1 Introduction

The class of *conjugate direction methods* can be viewed as being intermediate between the method of steepest descent and Newton's method. The conjugate direction methods have the following properties:

1. Solve quadratics of n variables in n steps.
2. The usual implementation, the *conjugate gradient algorithm*, requires no Hessian matrix evaluations.
3. No matrix inversion and no storage of an $n \times n$ matrix are required.

The conjugate direction methods typically perform better than the method of steepest descent, but not as well as Newton's method. As we saw from the method of steepest descent and Newton's method, the crucial factor in the efficiency of an iterative search method is the direction of search at each iteration. For a quadratic function of n variables $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{Q}\boldsymbol{x} - \boldsymbol{x}^\top \boldsymbol{b}$, $\boldsymbol{x} \in \mathbb{R}^n$, $\boldsymbol{Q} = \boldsymbol{Q}^\top > 0$, the best direction of search, as we shall see, is in the $\boldsymbol{Q}$-*conjugate* direction. Basically, two directions $\boldsymbol{d}^{(1)}$ and $\boldsymbol{d}^{(2)}$ in $\mathbb{R}^n$ are

said to be $\boldsymbol{Q}$-conjugate if $\boldsymbol{d}^{(1)\top}\boldsymbol{Q}\boldsymbol{d}^{(2)} = 0$. In general, we have the following definition.

Definition 10.1 Let $\boldsymbol{Q}$ be a real symmetric $n \times n$ matrix. The directions $\boldsymbol{d}^{(0)}, \boldsymbol{d}^{(1)}, \boldsymbol{d}^{(2)}, \ldots, \boldsymbol{d}^{(m)}$ are *$\boldsymbol{Q}$-conjugate* if for all $i \neq j$, we have $\boldsymbol{d}^{(i)\top}\boldsymbol{Q}\boldsymbol{d}^{(j)} = 0$. ∎

Lemma 10.1 *Let $\boldsymbol{Q}$ be a symmetric positive definite $n \times n$ matrix. If the directions $\boldsymbol{d}^{(0)}, \boldsymbol{d}^{(1)}, \ldots, \boldsymbol{d}^{(k)} \in \mathbb{R}^n$, $k \leq n-1$, are nonzero and $\boldsymbol{Q}$-conjugate, then they are linearly independent.* □

Proof. Let $\alpha_0, \ldots, \alpha_k$ be scalars such that

$$\alpha_0\boldsymbol{d}^{(0)} + \alpha_1\boldsymbol{d}^{(1)} + \cdots + \alpha_k\boldsymbol{d}^{(k)} = \boldsymbol{0}.$$

Premultiplying this equality by $\boldsymbol{d}^{(j)\top}\boldsymbol{Q}$, $0 \leq j \leq k$, yields

$$\alpha_j\boldsymbol{d}^{(j)\top}\boldsymbol{Q}\boldsymbol{d}^{(j)} = 0,$$

because all other terms $\boldsymbol{d}^{(j)\top}\boldsymbol{Q}\boldsymbol{d}^{(i)} = 0$, $i \neq j$, by $\boldsymbol{Q}$-conjugacy. But $\boldsymbol{Q} = \boldsymbol{Q}^\top > 0$ and $\boldsymbol{d}^{(j)} \neq \boldsymbol{0}$; hence $\alpha_j = 0$, $j = 0, 1, \ldots, k$. Therefore, $\boldsymbol{d}^{(0)}, \boldsymbol{d}^{(1)}, \ldots, \boldsymbol{d}^{(k)}$, $k \leq n-1$, are linearly independent. ∎

Example 10.1 Let

$$\boldsymbol{Q} = \begin{bmatrix} 3 & 0 & 1 \\ 0 & 4 & 2 \\ 1 & 2 & 3 \end{bmatrix}.$$

Note that $\boldsymbol{Q} = \boldsymbol{Q}^\top > 0$. The matrix $\boldsymbol{Q}$ is positive definite because all its leading principal minors are positive:

$$\Delta_1 = 3 > 0, \qquad \Delta_2 = \det\begin{bmatrix} 3 & 0 \\ 0 & 4 \end{bmatrix} = 12 > 0, \qquad \Delta_3 = \det\boldsymbol{Q} = 20 > 0.$$

Our goal is to construct a set of $\boldsymbol{Q}$-conjugate vectors $\boldsymbol{d}^{(0)}, \boldsymbol{d}^{(1)}, \boldsymbol{d}^{(2)}$.

Let $\boldsymbol{d}^{(0)} = [1, 0, 0]^\top$, $\boldsymbol{d}^{(1)} = [d_1^{(1)}, d_2^{(1)}, d_3^{(1)}]^\top$, $\boldsymbol{d}^{(2)} = [d_1^{(2)}, d_2^{(2)}, d_3^{(2)}]^\top$. We require that $\boldsymbol{d}^{(0)\top}\boldsymbol{Q}\boldsymbol{d}^{(1)} = 0$. We have

$$\boldsymbol{d}^{(0)\top}\boldsymbol{Q}\boldsymbol{d}^{(1)} = [1, 0, 0]\begin{bmatrix} 3 & 0 & 1 \\ 0 & 4 & 2 \\ 1 & 2 & 3 \end{bmatrix}\begin{bmatrix} d_1^{(1)} \\ d_2^{(1)} \\ d_3^{(1)} \end{bmatrix} = 3d_1^{(1)} + d_3^{(1)}.$$

Let $d_1^{(1)} = 1$, $d_2^{(1)} = 0$, $d_3^{(1)} = -3$. Then, $\boldsymbol{d}^{(1)} = [1, 0, -3]^\top$, and thus $\boldsymbol{d}^{(0)\top}\boldsymbol{Q}\boldsymbol{d}^{(1)} = 0$.

To find the third vector $\boldsymbol{d}^{(2)}$, which would be $\boldsymbol{Q}$-conjugate with $\boldsymbol{d}^{(0)}$ and $\boldsymbol{d}^{(1)}$, we require that $\boldsymbol{d}^{(0)\top}\boldsymbol{Q}\boldsymbol{d}^{(2)} = 0$ and $\boldsymbol{d}^{(1)\top}\boldsymbol{Q}\boldsymbol{d}^{(2)} = 0$. We have

$$\boldsymbol{d}^{(0)\top}\boldsymbol{Q}\boldsymbol{d}^{(2)} = 3d_1^{(2)} + d_3^{(2)} = 0,$$
$$\boldsymbol{d}^{(1)\top}\boldsymbol{Q}\boldsymbol{d}^{(2)} = -6d_2^{(2)} - 8d_3^{(2)} = 0.$$

If we take $\boldsymbol{d}^{(2)} = [1, 4, -3]^\top$, then the resulting set of vectors is mutually conjugate. ∎

This method of finding $\boldsymbol{Q}$-conjugate vectors is inefficient. A systematic procedure for finding $\boldsymbol{Q}$-conjugate vectors can be devised using the idea underlying the *Gram-Schmidt process* of transforming a given basis of $\mathbb{R}^n$ into an orthonormal basis of $\mathbb{R}^n$ (see Exercise 10.1).

10.2 The Conjugate Direction Algorithm

We now present the conjugate direction algorithm for minimizing the quadratic function of n variables

$$f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top\boldsymbol{Q}\boldsymbol{x} - \boldsymbol{x}^\top\boldsymbol{b},$$

where $\boldsymbol{Q} = \boldsymbol{Q}^\top > 0, \boldsymbol{x} \in \mathbb{R}^n$. Note that because $\boldsymbol{Q} > 0$, the function f has a global minimizer that can be found by solving $\boldsymbol{Q}\boldsymbol{x} = \boldsymbol{b}$.

Basic Conjugate Direction Algorithm. Given a starting point $\boldsymbol{x}^{(0)}$ and $\boldsymbol{Q}$-conjugate directions $\boldsymbol{d}^{(0)}, \boldsymbol{d}^{(1)}, \ldots, \boldsymbol{d}^{(n-1)}$; for $k \geq 0$,

$$\boldsymbol{g}^{(k)} = \nabla f(\boldsymbol{x}^{(k)}) = \boldsymbol{Q}\boldsymbol{x}^{(k)} - \boldsymbol{b},$$
$$\alpha_k = -\frac{\boldsymbol{g}^{(k)\top}\boldsymbol{d}^{(k)}}{\boldsymbol{d}^{(k)\top}\boldsymbol{Q}\boldsymbol{d}^{(k)}},$$
$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \alpha_k\boldsymbol{d}^{(k)}.$$

Theorem 10.1 *For any starting point $\boldsymbol{x}^{(0)}$, the basic conjugate direction algorithm converges to the unique $\boldsymbol{x}^*$ (that solves $\boldsymbol{Q}\boldsymbol{x} = \boldsymbol{b}$) in n steps; that is, $\boldsymbol{x}^{(n)} = \boldsymbol{x}^*$.* □

Proof. Consider $\boldsymbol{x}^* - \boldsymbol{x}^{(0)} \in \mathbb{R}^n$. Because the $\boldsymbol{d}^{(i)}$ are linearly independent, there exist constants β_i, $i = 0, \ldots, n-1$, such that

$$\boldsymbol{x}^* - \boldsymbol{x}^{(0)} = \beta_0\boldsymbol{d}^{(0)} + \cdots + \beta_{n-1}\boldsymbol{d}^{(n-1)}.$$

Now premultiply both sides of this equation by $\boldsymbol{d}^{(k)\top}\boldsymbol{Q}$, $0 \leq k < n$, to obtain

$$\boldsymbol{d}^{(k)\top}\boldsymbol{Q}(\boldsymbol{x}^* - \boldsymbol{x}^{(0)}) = \beta_k\boldsymbol{d}^{(k)\top}\boldsymbol{Q}\boldsymbol{d}^{(k)},$$

where the terms $\boldsymbol{d}^{(k)\top}\boldsymbol{Q}\boldsymbol{d}^{(i)} = 0$, $k \neq i$, by the $\boldsymbol{Q}$-conjugate property. Hence,

$$\beta_k = \frac{\boldsymbol{d}^{(k)\top}\boldsymbol{Q}(\boldsymbol{x}^* - \boldsymbol{x}^{(0)})}{\boldsymbol{d}^{(k)\top}\boldsymbol{Q}\boldsymbol{d}^{(k)}}.$$

Now, we can write

$$\boldsymbol{x}^{(k)} = \boldsymbol{x}^{(0)} + \alpha_0\boldsymbol{d}^{(0)} + \cdots + \alpha_{k-1}\boldsymbol{d}^{(k-1)}.$$

Therefore,

$$\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(0)} = \alpha_0\boldsymbol{d}^{(0)} + \cdots + \alpha_{k-1}\boldsymbol{d}^{(k-1)}.$$

So writing

$$\boldsymbol{x}^* - \boldsymbol{x}^{(0)} = (\boldsymbol{x}^* - \boldsymbol{x}^{(k)}) + (\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(0)})$$

and premultiplying the above by $\boldsymbol{d}^{(k)\top}\boldsymbol{Q}$, we obtain

$$\boldsymbol{d}^{(k)\top}\boldsymbol{Q}(\boldsymbol{x}^* - \boldsymbol{x}^{(0)}) = \boldsymbol{d}^{(k)\top}\boldsymbol{Q}(\boldsymbol{x}^* - \boldsymbol{x}^{(k)}) = -\boldsymbol{d}^{(k)\top}\boldsymbol{g}^{(k)},$$

because $\boldsymbol{g}^{(k)} = \boldsymbol{Q}\boldsymbol{x}^{(k)} - \boldsymbol{b}$ and $\boldsymbol{Q}\boldsymbol{x}^* = \boldsymbol{b}$. Thus,

$$\beta_k = -\frac{\boldsymbol{d}^{(k)\top}\boldsymbol{g}^{(k)}}{\boldsymbol{d}^{(k)\top}\boldsymbol{Q}\boldsymbol{d}^{(k)}} = \alpha_k$$

and $\boldsymbol{x}^* = \boldsymbol{x}^{(n)}$, which completes the proof. ∎

Example 10.2 Find the minimizer of

$$f(x_1, x_2) = \frac{1}{2}\boldsymbol{x}^\top \begin{bmatrix} 4 & 2 \\ 2 & 2 \end{bmatrix} \boldsymbol{x} - \boldsymbol{x}^\top \begin{bmatrix} -1 \\ 1 \end{bmatrix}, \boldsymbol{x} \in \mathbb{R}^2,$$

using the conjugate direction method with the initial point $\boldsymbol{x}^{(0)} = [0, 0]^\top$, and $\boldsymbol{Q}$-conjugate directions $\boldsymbol{d}^{(0)} = [1, 0]^\top$ and $\boldsymbol{d}^{(1)} = \left[-\frac{3}{8}, \frac{3}{4}\right]^\top$.

We have

$$\boldsymbol{g}^{(0)} = -\boldsymbol{b} = [1, -1]^\top,$$

and hence

$$\alpha_0 = -\frac{\boldsymbol{g}^{(0)\top}\boldsymbol{d}^{(0)}}{\boldsymbol{d}^{(0)\top}\boldsymbol{Q}\boldsymbol{d}^{(0)}} = -\frac{[1, -1]\begin{bmatrix} 1 \\ 0 \end{bmatrix}}{[1, 0]\begin{bmatrix} 4 & 2 \\ 2 & 2 \end{bmatrix}\begin{bmatrix} 1 \\ 0 \end{bmatrix}} = -\frac{1}{4}.$$

Thus,

$$\boldsymbol{x}^{(1)} = \boldsymbol{x}^{(0)} + \alpha_0\boldsymbol{d}^{(0)} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} - \frac{1}{4}\begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} -\frac{1}{4} \\ 0 \end{bmatrix}.$$

To find $\boldsymbol{x}^{(2)}$, we compute

$$\boldsymbol{g}^{(1)} = \boldsymbol{Q}\boldsymbol{x}^{(1)} - \boldsymbol{b} = \begin{bmatrix} 4 & 2 \\ 2 & 2 \end{bmatrix} \begin{bmatrix} -\frac{1}{4} \\ 0 \end{bmatrix} - \begin{bmatrix} -1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ -\frac{3}{2} \end{bmatrix}$$

and

$$\alpha_1 = -\frac{\boldsymbol{g}^{(1)\top}\boldsymbol{d}^{(1)}}{\boldsymbol{d}^{(1)\top}\boldsymbol{Q}\boldsymbol{d}^{(1)}} = -\frac{\left[0, -\frac{3}{2}\right] \begin{bmatrix} -\frac{3}{8} \\ \frac{3}{4} \end{bmatrix}}{\left[-\frac{3}{8}, \frac{3}{4}\right] \begin{bmatrix} 4 & 2 \\ 2 & 2 \end{bmatrix} \begin{bmatrix} -\frac{3}{8} \\ \frac{3}{4} \end{bmatrix}} = 2.$$

Therefore,

$$\boldsymbol{x}^{(2)} = \boldsymbol{x}^{(1)} + \alpha_1 \boldsymbol{d}^{(1)} = \begin{bmatrix} -\frac{1}{4} \\ 0 \end{bmatrix} + 2 \begin{bmatrix} -\frac{3}{8} \\ \frac{3}{4} \end{bmatrix} = \begin{bmatrix} -1 \\ \frac{3}{2} \end{bmatrix}.$$

Because f is a quadratic function in two variables, $\boldsymbol{x}^{(2)} = \boldsymbol{x}^*$. ∎

For a quadratic function of n variables, the conjugate direction method reaches the solution after n steps. As we shall see below, the method also possesses a certain desirable property in the intermediate steps. To see this, suppose that we start at $\boldsymbol{x}^{(0)}$ and search in the direction $\boldsymbol{d}^{(0)}$ to obtain

$$\boldsymbol{x}^{(1)} = \boldsymbol{x}^{(0)} - \left(\frac{\boldsymbol{g}^{(0)\top}\boldsymbol{d}^{(0)}}{\boldsymbol{d}^{(0)\top}\boldsymbol{Q}\boldsymbol{d}^{(0)}} \right) \boldsymbol{d}^{(0)}.$$

We claim that

$$\boldsymbol{g}^{(1)\top}\boldsymbol{d}^{(0)} = 0.$$

To see this,

$$\begin{aligned} \boldsymbol{g}^{(1)\top}\boldsymbol{d}^{(0)} &= (\boldsymbol{Q}\boldsymbol{x}^{(1)} - \boldsymbol{b})^\top \boldsymbol{d}^{(0)} \\ &= \boldsymbol{x}^{(0)\top}\boldsymbol{Q}\boldsymbol{d}^{(0)} - \left(\frac{\boldsymbol{g}^{(0)\top}\boldsymbol{d}^{(0)}}{\boldsymbol{d}^{(0)\top}\boldsymbol{Q}\boldsymbol{d}^{(0)}} \right) \boldsymbol{d}^{(0)\top}\boldsymbol{Q}\boldsymbol{d}^{(0)} - \boldsymbol{b}^\top \boldsymbol{d}^{(0)} \\ &= \boldsymbol{g}^{(0)\top}\boldsymbol{d}^{(0)} - \boldsymbol{g}^{(0)\top}\boldsymbol{d}^{(0)} = 0. \end{aligned}$$

The equation $\boldsymbol{g}^{(1)\top}\boldsymbol{d}^{(0)} = 0$ implies that α_0 has the property that $\alpha_0 = \arg\min \phi_0(\alpha)$, where $\phi_0(\alpha) = f(\boldsymbol{x}^{(0)} + \alpha \boldsymbol{d}^{(0)})$. To see this, apply the chain rule to get

$$\frac{d\phi_0}{d\alpha}(\alpha) = \nabla f(\boldsymbol{x}^{(0)} + \alpha \boldsymbol{d}^{(0)})^\top \boldsymbol{d}^{(0)}.$$

Evaluating the above at $\alpha = \alpha_0$, we get

$$\frac{d\phi_0}{d\alpha}(\alpha_0) = \boldsymbol{g}^{(1)\top}\boldsymbol{d}^{(0)} = 0.$$

Because ϕ_0 is a quadratic function of α, and the coefficient of the α^2 term in ϕ_0 is $\boldsymbol{d}^{(0)\top}\boldsymbol{Q}\boldsymbol{d}^{(0)} > 0$, the above implies that $\alpha_0 = \arg\min_{\alpha\in\mathbb{R}} \phi_0(\alpha)$.

Using a similar argument, we can show that for all k,

$$\boldsymbol{g}^{(k+1)\top}\boldsymbol{d}^{(k)} = 0$$

and hence

$$\alpha_k = \arg\min f(\boldsymbol{x}^{(k)} + \alpha\boldsymbol{d}^{(k)}).$$

In fact, an even stronger condition holds, as given by the following lemma.

Lemma 10.2 *In the conjugate direction algorithm,*

$$\boldsymbol{g}^{(k+1)\top}\boldsymbol{d}^{(i)} = 0$$

for all k, $0 \le k \le n-1$, *and* $0 \le i \le k$. □

Proof. Note that

$$\boldsymbol{Q}(\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)}) = \boldsymbol{Q}\boldsymbol{x}^{(k+1)} - \boldsymbol{b} - (\boldsymbol{Q}\boldsymbol{x}^{(k)} - \boldsymbol{b}) = \boldsymbol{g}^{(k+1)} - \boldsymbol{g}^{(k)},$$

because $\boldsymbol{g}^{(k)} = \boldsymbol{Q}\boldsymbol{x}^{(k)} - \boldsymbol{b}$. Thus,

$$\boldsymbol{g}^{(k+1)} = \boldsymbol{g}^{(k)} + \alpha_k\boldsymbol{Q}\boldsymbol{d}^{(k)}.$$

We prove the lemma by induction. The result is true for $k = 0$ because $\boldsymbol{g}^{(1)\top}\boldsymbol{d}^{(0)} = 0$, as shown before. We now show that if the result is true for $k-1$ (i.e., $\boldsymbol{g}^{(k)\top}\boldsymbol{d}^{(i)} = 0$, $i \le k-1$), then it is true for k (i.e., $\boldsymbol{g}^{(k+1)\top}\boldsymbol{d}^{(i)} = 0$, $i \le k$). Fix $k > 0$ and $0 \le i < k$. By the induction hypothesis, $\boldsymbol{g}^{(k)\top}\boldsymbol{d}^{(i)} = 0$. Because

$$\boldsymbol{g}^{(k+1)} = \boldsymbol{g}^{(k)} + \alpha_k\boldsymbol{Q}\boldsymbol{d}^{(k)},$$

and $\boldsymbol{d}^{(k)\top}\boldsymbol{Q}\boldsymbol{d}^{(i)} = 0$ by $\boldsymbol{Q}$-conjugacy, we have

$$\boldsymbol{g}^{(k+1)\top}\boldsymbol{d}^{(i)} = \boldsymbol{g}^{(k)\top}\boldsymbol{d}^{(i)} + \alpha_k\boldsymbol{d}^{(k)\top}\boldsymbol{Q}\boldsymbol{d}^{(i)} = 0.$$

It remains to be shown that

$$\boldsymbol{g}^{(k+1)\top}\boldsymbol{d}^{(k)} = 0.$$

Indeed,

$$\begin{aligned}
\boldsymbol{g}^{(k+1)\top}\boldsymbol{d}^{(k)} &= (\boldsymbol{Q}\boldsymbol{x}^{(k+1)} - \boldsymbol{b})^\top\boldsymbol{d}^{(k)} \\
&= \left(\boldsymbol{x}^{(k)} - \frac{\boldsymbol{g}^{(k)\top}\boldsymbol{d}^{(k)}}{\boldsymbol{d}^{(k)\top}\boldsymbol{Q}\boldsymbol{d}^{(k)}}\boldsymbol{d}^{(k)}\right)^\top \boldsymbol{Q}\boldsymbol{d}^{(k)} - \boldsymbol{b}^\top\boldsymbol{d}^{(k)} \\
&= \left(\boldsymbol{Q}\boldsymbol{x}^{(k)} - \boldsymbol{b}\right)^\top \boldsymbol{d}^{(k)} - \boldsymbol{g}^{(k)\top}\boldsymbol{d}^{(k)} \\
&= 0,
\end{aligned}$$

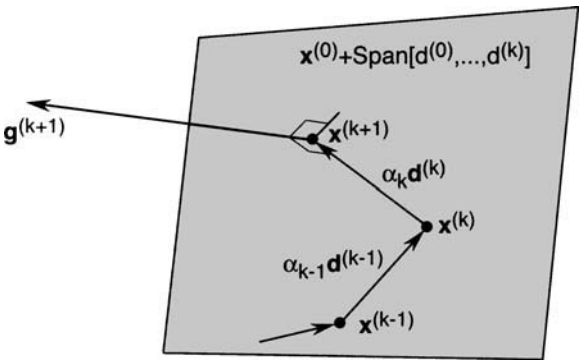

Figure 10.1 Illustration of Lemma 10.2.

because $\boldsymbol{Q}\boldsymbol{x}^{(k)} - \boldsymbol{b} = \boldsymbol{g}^{(k)}$.

Therefore, by induction, for all $0 \leq k \leq n-1$ and $0 \leq i \leq k$,

$$\boldsymbol{g}^{(k+1)\top}\boldsymbol{d}^{(i)} = 0.$$

∎

By Lemma 10.2 we see that $\boldsymbol{g}^{(k+1)}$ is orthogonal to any vector from the subspace spanned by $\boldsymbol{d}^{(0)}, \boldsymbol{d}^{(1)}, \ldots, \boldsymbol{d}^{(k)}$. Figure 10.1 illustrates this statement.

The lemma can be used to show an interesting optimal property of the conjugate direction algorithm. Specifically, we now show that not only does $f(\boldsymbol{x}^{(k+1)})$ satisfy $f(\boldsymbol{x}^{(k+1)}) = \min_\alpha f(\boldsymbol{x}^{(k)} + \alpha \boldsymbol{d}^{(k)})$, as indicated before, but also

$$f(\boldsymbol{x}^{(k+1)}) = \min_{a_0,\ldots,a_k} f\left(\boldsymbol{x}^{(0)} + \sum_{i=0}^{k} a_i \boldsymbol{d}^{(i)}\right).$$

In other words, if we write

$$\mathcal{V}_k = \boldsymbol{x}^{(0)} + \operatorname{span}[\boldsymbol{d}^{(0)}, \boldsymbol{d}^{(1)}, \ldots, \boldsymbol{d}^{(k)}],$$

then we can express $f(\boldsymbol{x}^{(k+1)}) = \min_{\boldsymbol{x}\in\mathcal{V}_k} f(\boldsymbol{x})$. As k increases, the subspace $\operatorname{span}[\boldsymbol{d}^{(0)}, \boldsymbol{d}^{(1)}, \ldots, \boldsymbol{d}^{(k)}]$ "expands," and will eventually fill the whole of $\mathbb{R}^n$ (provided that the vectors $\boldsymbol{d}^{(0)}, \boldsymbol{d}^{(1)}, \ldots,$ are linearly independent). Therefore, for some sufficiently large k, $\boldsymbol{x}^*$ will lie in $\mathcal{V}_k$. For this reason, the above result is sometimes called the *expanding subspace theorem* (see, e.g., [88, p. 266]).

To prove the expanding subspace theorem, define the matrix $\boldsymbol{D}^{(k)}$ by

$$\boldsymbol{D}^{(k)} = [\boldsymbol{d}^{(0)}, \ldots, \boldsymbol{d}^{(k)}];$$

that is, $\boldsymbol{d}^{(i)}$ is the ith column of $\boldsymbol{D}^{(k)}$. Note that $\boldsymbol{x}^{(0)} + \mathcal{R}(\boldsymbol{D}^{(k)}) = \mathcal{V}_k$. Also,

$$\begin{aligned}\boldsymbol{x}^{(k+1)} &= \boldsymbol{x}^{(0)} + \sum_{i=0}^{k} \alpha_i \boldsymbol{d}^{(i)} \\ &= \boldsymbol{x}^{(0)} + \boldsymbol{D}^{(k)}\boldsymbol{\alpha},\end{aligned}$$

where $\boldsymbol{\alpha} = [\alpha_0, \ldots, \alpha_k]^\top$. Hence,

$$\boldsymbol{x}^{(k+1)} \in \boldsymbol{x}^{(0)} + \mathcal{R}(\boldsymbol{D}^{(k)}) = \mathcal{V}_k.$$

Now, consider any vector $\boldsymbol{x} \in \mathcal{V}_k$. There exists a vector $\boldsymbol{a}$ such that $\boldsymbol{x} = \boldsymbol{x}^{(0)} + \boldsymbol{D}^{(k)}\boldsymbol{a}$. Let $\phi_k(\boldsymbol{a}) = f(\boldsymbol{x}^{(0)} + \boldsymbol{D}^{(k)}\boldsymbol{a})$. Note that ϕ_k is a quadratic function and has a unique minimizer that satisfies the FONC (see Exercises 6.33 and 10.7). By the chain rule,

$$D\phi_k(\boldsymbol{a}) = \nabla f(\boldsymbol{x}^{(0)} + \boldsymbol{D}^{(k)}\boldsymbol{a})^\top \boldsymbol{D}^{(k)}.$$

Therefore,

$$\begin{aligned}D\phi_k(\boldsymbol{\alpha}) &= \nabla f(\boldsymbol{x}^{(0)} + \boldsymbol{D}^{(k)}\boldsymbol{\alpha})^\top \boldsymbol{D}^{(k)} \\ &= \nabla f(\boldsymbol{x}^{(k+1)})^\top \boldsymbol{D}^{(k)} \\ &= \boldsymbol{g}^{(k+1)\top} \boldsymbol{D}^{(k)}.\end{aligned}$$

By Lemma 10.2, $\boldsymbol{g}^{(k+1)\top}\boldsymbol{D}^{(k)} = \boldsymbol{0}^\top$. Therefore, $\boldsymbol{\alpha}$ satisfies the FONC for the quadratic function ϕ_k, and hence $\boldsymbol{\alpha}$ is the minimizer of ϕ_k; that is,

$$f(\boldsymbol{x}^{(k+1)}) = \min_{\boldsymbol{a}} f(\boldsymbol{x}^{(0)} + \boldsymbol{D}^{(k)}\boldsymbol{a}) = \min_{\boldsymbol{x} \in \mathcal{V}_k} f(\boldsymbol{x}),$$

which completes the proof of our result.

The conjugate direction algorithm is very effective. However, to use the algorithm, we need to specify the $\boldsymbol{Q}$-conjugate directions. Fortunately, there is a way to generate $\boldsymbol{Q}$-conjugate directions as we perform iterations. In the next section we discuss an algorithm that incorporates the generation of $\boldsymbol{Q}$-conjugate directions.

10.3 The Conjugate Gradient Algorithm

The conjugate gradient algorithm does not use prespecified conjugate directions, but instead computes the directions as the algorithm progresses. At each stage of the algorithm, the direction is calculated as a linear combination of the previous direction and the current gradient, in such a way that all the directions are mutually $\boldsymbol{Q}$-conjugate—hence the name *conjugate gradient algorithm.* This calculation exploits the fact that for a quadratic function of

n variables, we can locate the function minimizer by performing n searches along mutually conjugate directions.

As before, we consider the quadratic function

$$f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{Q}\boldsymbol{x} - \boldsymbol{x}^\top \boldsymbol{b}, \quad \boldsymbol{x} \in \mathbb{R}^n,$$

where $\boldsymbol{Q} = \boldsymbol{Q}^\top > 0$. Our first search direction from an initial point $\boldsymbol{x}^{(0)}$ is in the direction of steepest descent; that is,

$$\boldsymbol{d}^{(0)} = -\boldsymbol{g}^{(0)}.$$

Thus,

$$\boldsymbol{x}^{(1)} = \boldsymbol{x}^{(0)} + \alpha_0 \boldsymbol{d}^{(0)},$$

where

$$\alpha_0 = \arg\min_{\alpha \geq 0} f(\boldsymbol{x}^{(0)} + \alpha \boldsymbol{d}^{(0)}) = -\frac{\boldsymbol{g}^{(0)\top}\boldsymbol{d}^{(0)}}{\boldsymbol{d}^{(0)\top}\boldsymbol{Q}\boldsymbol{d}^{(0)}}.$$

In the next stage, we search in a direction $\boldsymbol{d}^{(1)}$ that is $\boldsymbol{Q}$-conjugate to $\boldsymbol{d}^{(0)}$. We choose $\boldsymbol{d}^{(1)}$ as a linear combination of $\boldsymbol{g}^{(1)}$ and $\boldsymbol{d}^{(0)}$. In general, at the $(k+1)$th step, we choose $\boldsymbol{d}^{(k+1)}$ to be a linear combination of $\boldsymbol{g}^{(k+1)}$ and $\boldsymbol{d}^{(k)}$. Specifically, we choose

$$\boldsymbol{d}^{(k+1)} = -\boldsymbol{g}^{(k+1)} + \beta_k \boldsymbol{d}^{(k)}, \quad k = 0, 1, 2, \ldots.$$

The coefficients β_k, $k = 1, 2, \ldots$, are chosen in such a way that $\boldsymbol{d}^{(k+1)}$ is $\boldsymbol{Q}$-conjugate to $\boldsymbol{d}^{(0)}, \boldsymbol{d}^{(1)}, \ldots, \boldsymbol{d}^{(k)}$. This is accomplished by choosing β_k to be

$$\beta_k = \frac{\boldsymbol{g}^{(k+1)\top}\boldsymbol{Q}\boldsymbol{d}^{(k)}}{\boldsymbol{d}^{(k)\top}\boldsymbol{Q}\boldsymbol{d}^{(k)}}.$$

The conjugate gradient algorithm is summarized below.

1. Set $k := 0$; select the initial point $\boldsymbol{x}^{(0)}$.
2. $\boldsymbol{g}^{(0)} = \nabla f(\boldsymbol{x}^{(0)})$. If $\boldsymbol{g}^{(0)} = \boldsymbol{0}$, stop; else, set $\boldsymbol{d}^{(0)} = -\boldsymbol{g}^{(0)}$.
3. $\alpha_k = -\frac{\boldsymbol{g}^{(k)\top}\boldsymbol{d}^{(k)}}{\boldsymbol{d}^{(k)\top}\boldsymbol{Q}\boldsymbol{d}^{(k)}}$.
4. $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \alpha_k \boldsymbol{d}^{(k)}$.
5. $\boldsymbol{g}^{(k+1)} = \nabla f(\boldsymbol{x}^{(k+1)})$. If $\boldsymbol{g}^{(k+1)} = \boldsymbol{0}$, stop.
6. $\beta_k = \frac{\boldsymbol{g}^{(k+1)\top}\boldsymbol{Q}\boldsymbol{d}^{(k)}}{\boldsymbol{d}^{(k)\top}\boldsymbol{Q}\boldsymbol{d}^{(k)}}$.
7. $\boldsymbol{d}^{(k+1)} = -\boldsymbol{g}^{(k+1)} + \beta_k \boldsymbol{d}^{(k)}$.
8. Set $k := k + 1$; go to step 3.

Proposition 10.1 *In the conjugate gradient algorithm, the directions* $\boldsymbol{d}^{(0)}, \boldsymbol{d}^{(1)}, \ldots, \boldsymbol{d}^{(n-1)}$ *are* $\boldsymbol{Q}$*-conjugate.* □

Proof. We use induction. We first show that $\boldsymbol{d}^{(0)\top}\boldsymbol{Q}\boldsymbol{d}^{(1)} = 0$. To this end we write

$$\boldsymbol{d}^{(0)\top}\boldsymbol{Q}\boldsymbol{d}^{(1)} = \boldsymbol{d}^{(0)\top}\boldsymbol{Q}(-\boldsymbol{g}^{(1)} + \beta_0\boldsymbol{d}^{(0)}).$$

Substituting for

$$\beta_0 = \frac{\boldsymbol{g}^{(1)\top}\boldsymbol{Q}\boldsymbol{d}^{(0)}}{\boldsymbol{d}^{(0)\top}\boldsymbol{Q}\boldsymbol{d}^{(0)}}$$

in the equation above, we see that $\boldsymbol{d}^{(0)\top}\boldsymbol{Q}\boldsymbol{d}^{(1)} = 0$.

We now assume that $\boldsymbol{d}^{(0)}, \boldsymbol{d}^{(1)}, \ldots, \boldsymbol{d}^{(k)}$, $k < n-1$, are $\boldsymbol{Q}$-conjugate directions. From Lemma 10.2 we have $\boldsymbol{g}^{(k+1)\top}\boldsymbol{d}^{(j)} = 0$, $j = 0, 1, \ldots, k$. Thus, $\boldsymbol{g}^{(k+1)}$ is orthogonal to each of the directions $\boldsymbol{d}^{(0)}, \boldsymbol{d}^{(1)}, \ldots, \boldsymbol{d}^{(k)}$. We now show that

$$\boldsymbol{g}^{(k+1)\top}\boldsymbol{g}^{(j)} = 0, \quad j = 0, 1, \ldots, k.$$

Fix $j \in \{0, \ldots, k\}$. We have

$$\boldsymbol{d}^{(j)} = -\boldsymbol{g}^{(j)} + \beta_{j-1}\boldsymbol{d}^{(j-1)}.$$

Substituting this equation into the previous one yields

$$\boldsymbol{g}^{(k+1)\top}\boldsymbol{d}^{(j)} = 0 = -\boldsymbol{g}^{(k+1)\top}\boldsymbol{g}^{(j)} + \beta_{j-1}\boldsymbol{g}^{(k+1)\top}\boldsymbol{d}^{(j-1)}.$$

Because $\boldsymbol{g}^{(k+1)\top}\boldsymbol{d}^{(j-1)} = 0$, it follows that $\boldsymbol{g}^{(k+1)\top}\boldsymbol{g}^{(j)} = 0$.

We are now ready to show that $\boldsymbol{d}^{(k+1)\top}\boldsymbol{Q}\boldsymbol{d}^{(j)} = 0$, $j = 0, \ldots, k$. We have

$$\boldsymbol{d}^{(k+1)\top}\boldsymbol{Q}\boldsymbol{d}^{(j)} = (-\boldsymbol{g}^{(k+1)} + \beta_k\boldsymbol{d}^{(k)})^\top\boldsymbol{Q}\boldsymbol{d}^{(j)}.$$

If $j < k$, then $\boldsymbol{d}^{(k)\top}\boldsymbol{Q}\boldsymbol{d}^{(j)} = 0$, by virtue of the induction hypothesis. Hence, we have

$$\boldsymbol{d}^{(k+1)\top}\boldsymbol{Q}\boldsymbol{d}^{(j)} = -\boldsymbol{g}^{(k+1)\top}\boldsymbol{Q}\boldsymbol{d}^{(j)}.$$

But $\boldsymbol{g}^{(j+1)} = \boldsymbol{g}^{(j)} + \alpha_j\boldsymbol{Q}\boldsymbol{d}^{(j)}$. Because $\boldsymbol{g}^{(k+1)\top}\boldsymbol{g}^{(i)} = 0$, $i = 0, \ldots, k$,

$$\boldsymbol{d}^{(k+1)\top}\boldsymbol{Q}\boldsymbol{d}^{(j)} = -\boldsymbol{g}^{(k+1)\top}\frac{(\boldsymbol{g}^{(j+1)} - \boldsymbol{g}^{(j)})}{\alpha_j} = 0.$$

Thus,

$$\boldsymbol{d}^{(k+1)\top}\boldsymbol{Q}\boldsymbol{d}^{(j)} = 0, \quad j = 0, \ldots, k-1.$$

It remains to be shown that $\boldsymbol{d}^{(k+1)\top}\boldsymbol{Q}\boldsymbol{d}^{(k)} = 0$. We have

$$\boldsymbol{d}^{(k+1)\top}\boldsymbol{Q}\boldsymbol{d}^{(k)} = (-\boldsymbol{g}^{(k+1)} + \beta_k\boldsymbol{d}^{(k)})^\top\boldsymbol{Q}\boldsymbol{d}^{(k)}.$$

Using the expression for β_k, we get $\boldsymbol{d}^{(k+1)\top}\boldsymbol{Q}\boldsymbol{d}^{(k)} = 0$, which completes the proof. ∎

Example 10.3 Consider the quadratic function

$$f(x_1, x_2, x_3) = \frac{3}{2}x_1^2 + 2x_2^2 + \frac{3}{2}x_3^2 + x_1x_3 + 2x_2x_3 - 3x_1 - x_3.$$

We find the minimizer using the conjugate gradient algorithm, using the starting point $\boldsymbol{x}^{(0)} = [0, 0, 0]^\top$.

We can represent f as

$$f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{Q}\boldsymbol{x} - \boldsymbol{x}^\top \boldsymbol{b},$$

where

$$\boldsymbol{Q} = \begin{bmatrix} 3 & 0 & 1 \\ 0 & 4 & 2 \\ 1 & 2 & 3 \end{bmatrix}, \qquad \boldsymbol{b} = \begin{bmatrix} 3 \\ 0 \\ 1 \end{bmatrix}.$$

We have

$$\boldsymbol{g}(\boldsymbol{x}) = \nabla f(\boldsymbol{x}) = \boldsymbol{Q}\boldsymbol{x} - \boldsymbol{b} = [3x_1 + x_3 - 3, 4x_2 + 2x_3, x_1 + 2x_2 + 3x_3 - 1]^\top.$$

Hence,

$$\begin{aligned} \boldsymbol{g}^{(0)} &= [-3, 0, -1]^\top, \\ \boldsymbol{d}^{(0)} &= -\boldsymbol{g}^{(0)}, \\ \alpha_0 &= -\frac{\boldsymbol{g}^{(0)\top}\boldsymbol{d}^{(0)}}{\boldsymbol{d}^{(0)\top}\boldsymbol{Q}\boldsymbol{d}^{(0)}} = \frac{10}{36} = 0.2778 \end{aligned}$$

and

$$\boldsymbol{x}^{(1)} = \boldsymbol{x}^{(0)} + \alpha_0 \boldsymbol{d}^{(0)} = [0.8333, 0, 0.2778]^\top.$$

The next stage yields

$$\begin{aligned} \boldsymbol{g}^{(1)} &= \nabla f(\boldsymbol{x}^{(1)}) = [-0.2222, 0.5556, 0.6667]^\top, \\ \beta_0 &= \frac{\boldsymbol{g}^{(1)\top}\boldsymbol{Q}\boldsymbol{d}^{(0)}}{\boldsymbol{d}^{(0)\top}\boldsymbol{Q}\boldsymbol{d}^{(0)}} = 0.08025. \end{aligned}$$

We can now compute

$$\boldsymbol{d}^{(1)} = -\boldsymbol{g}^{(1)} + \beta_0 \boldsymbol{d}^{(0)} = [0.4630, -0.5556, -0.5864]^\top.$$

Hence,

$$\alpha_1 = -\frac{\boldsymbol{g}^{(1)\top}\boldsymbol{d}^{(1)}}{\boldsymbol{d}^{(1)\top}\boldsymbol{Q}\boldsymbol{d}^{(1)}} = 0.2187$$

and

$$\boldsymbol{x}^{(2)} = \boldsymbol{x}^{(1)} + \alpha_1 \boldsymbol{d}^{(1)} = [0.9346, -0.1215, 0.1495]^\top.$$

To perform the third iteration, we compute

$$\boldsymbol{g}^{(2)} = \nabla f(\boldsymbol{x}^{(2)}) = [-0.04673, -0.1869, 0.1402]^\top,$$
$$\beta_1 = \frac{\boldsymbol{g}^{(2)\top}\boldsymbol{Q}\boldsymbol{d}^{(1)}}{\boldsymbol{d}^{(1)\top}\boldsymbol{Q}\boldsymbol{d}^{(1)}} = 0.07075,$$
$$\boldsymbol{d}^{(2)} = -\boldsymbol{g}^{(2)} + \beta_1 \boldsymbol{d}^{(1)} = [0.07948, 0.1476, -0.1817]^\top.$$

Hence,

$$\alpha_2 = -\frac{\boldsymbol{g}^{(2)\top}\boldsymbol{d}^{(2)}}{\boldsymbol{d}^{(2)\top}\boldsymbol{Q}\boldsymbol{d}^{(2)}} = 0.8231$$

and

$$\boldsymbol{x}^{(3)} = \boldsymbol{x}^{(2)} + \alpha_2 \boldsymbol{d}^{(2)} = [1.000, 0.000, 0.000]^\top.$$

Note that

$$\boldsymbol{g}^{(3)} = \nabla f(\boldsymbol{x}^{(3)}) = \boldsymbol{0},$$

as expected, because f is a quadratic function of three variables. Hence, $\boldsymbol{x}^* = \boldsymbol{x}^{(3)}$. ∎

10.4 The Conjugate Gradient Algorithm for Nonquadratic Problems

In Section 10.3, we showed that the conjugate gradient algorithm is a conjugate direction method, and therefore minimizes a positive definite quadratic function of n variables in n steps. The algorithm can be extended to general nonlinear functions by interpreting $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top\boldsymbol{Q}\boldsymbol{x} - \boldsymbol{x}^\top\boldsymbol{b}$ as a second-order Taylor series approximation of the objective function. Near the solution such functions behave approximately as quadratics, as suggested by the Taylor series expansion. For a quadratic, the matrix $\boldsymbol{Q}$, the Hessian of the quadratic, is constant. However, for a general nonlinear function the Hessian is a matrix that has to be reevaluated at each iteration of the algorithm. This can be computationally very expensive. Thus, an efficient implementation of the conjugate gradient algorithm that eliminates the Hessian evaluation at each step is desirable.

Observe that $\boldsymbol{Q}$ appears only in the computation of the scalars α_k and β_k. Because

$$\alpha_k = \arg\min_{\alpha \geq 0} f(\boldsymbol{x}^{(k)} + \alpha \boldsymbol{d}^{(k)}),$$

the closed-form formula for α_k in the algorithm can be replaced by a numerical line search procedure. Therefore, we need only concern ourselves with the formula for β_k. Fortunately, elimination of $\boldsymbol{Q}$ from the formula is possible and results in algorithms that depend only on the function and gradient values at

each iteration. We now discuss modifications of the conjugate gradient algorithm for a quadratic function for the case in which the Hessian is unknown but in which objective function values and gradients are available. The modifications are all based on algebraically manipulating the formula β_k in such a way that $\boldsymbol{Q}$ is eliminated. We discuss three well-known modifications.

Hestenes-Stiefel Formula. Recall that

$$\beta_k = \frac{\boldsymbol{g}^{(k+1)\top}\boldsymbol{Q}\boldsymbol{d}^{(k)}}{\boldsymbol{d}^{(k)\top}\boldsymbol{Q}\boldsymbol{d}^{(k)}}.$$

The Hestenes-Stiefel formula is based on replacing the term $\boldsymbol{Q}\boldsymbol{d}^{(k)}$ by the term $(\boldsymbol{g}^{(k+1)} - \boldsymbol{g}^{(k)})/\alpha_k$. The two terms are equal in the quadratic case, as we now show. Now, $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \alpha_k \boldsymbol{d}^{(k)}$. Premultiplying both sides by $\boldsymbol{Q}$, subtracting $\boldsymbol{b}$ from both sides, and recognizing that $\boldsymbol{g}^{(k)} = \boldsymbol{Q}\boldsymbol{x}^{(k)} - \boldsymbol{b}$, we get $\boldsymbol{g}^{(k+1)} = \boldsymbol{g}^{(k)} + \alpha_k \boldsymbol{Q}\boldsymbol{d}^{(k)}$, which we can rewrite as $\boldsymbol{Q}\boldsymbol{d}^{(k)} = (\boldsymbol{g}^{(k+1)} - \boldsymbol{g}^{(k)})/\alpha_k$. Substituting this into the original equation for β_k gives the *Hestenes-Stiefel formula*

$$\beta_k = \frac{\boldsymbol{g}^{(k+1)\top}[\boldsymbol{g}^{(k+1)} - \boldsymbol{g}^{(k)}]}{\boldsymbol{d}^{(k)\top}[\boldsymbol{g}^{(k+1)} - \boldsymbol{g}^{(k)}]}.$$

Polak-Ribière Formula. Starting from the Hestenes-Stiefel formula, we multiply out the denominator to get

$$\beta_k = \frac{\boldsymbol{g}^{(k+1)\top}[\boldsymbol{g}^{(k+1)} - \boldsymbol{g}^{(k)}]}{\boldsymbol{d}^{(k)\top}\boldsymbol{g}^{(k+1)} - \boldsymbol{d}^{(k)\top}\boldsymbol{g}^{(k)}}.$$

By Lemma 10.2, $\boldsymbol{d}^{(k)\top}\boldsymbol{g}^{(k+1)} = 0$. Also, since $\boldsymbol{d}^{(k)} = -\boldsymbol{g}^{(k)} + \beta_{k-1}\boldsymbol{d}^{(k-1)}$, and premultiplying this by $\boldsymbol{g}^{(k)\top}$, we get

$$\boldsymbol{g}^{(k)\top}\boldsymbol{d}^{(k)} = -\boldsymbol{g}^{(k)\top}\boldsymbol{g}^{(k)} + \beta_{k-1}\boldsymbol{g}^{(k)\top}\boldsymbol{d}^{(k-1)} = -\boldsymbol{g}^{(k)\top}\boldsymbol{g}^{(k)},$$

where once again we used Lemma 10.2. Hence, we get the Polak-Ribière formula

$$\beta_k = \frac{\boldsymbol{g}^{(k+1)\top}[\boldsymbol{g}^{(k+1)} - \boldsymbol{g}^{(k)}]}{\boldsymbol{g}^{(k)\top}\boldsymbol{g}^{(k)}}.$$

Fletcher-Reeves Formula. Starting with the Polak-Ribière formula, we multiply out the numerator to get

$$\beta_k = \frac{\boldsymbol{g}^{(k+1)\top}\boldsymbol{g}^{(k+1)} - \boldsymbol{g}^{(k+1)\top}\boldsymbol{g}^{(k)}}{\boldsymbol{g}^{(k)\top}\boldsymbol{g}^{(k)}}.$$

We now use the fact that $\boldsymbol{g}^{(k+1)\top}\boldsymbol{g}^{(k)} = 0$, which we get by using the equation

$$\boldsymbol{g}^{(k+1)\top}\boldsymbol{d}^{(k)} = -\boldsymbol{g}^{(k+1)\top}\boldsymbol{g}^{(k)} + \beta_{k-1}\boldsymbol{g}^{(k+1)\top}\boldsymbol{d}^{(k-1)}$$

and applying Lemma 10.2. This leads to the Fletcher-Reeves formula

$$\beta_k = \frac{\boldsymbol{g}^{(k+1)\top}\boldsymbol{g}^{(k+1)}}{\boldsymbol{g}^{(k)\top}\boldsymbol{g}^{(k)}}.$$

The formulas above give us conjugate gradient algorithms that do not require explicit knowledge of the Hessian matrix $\boldsymbol{Q}$. All we need are the objective function and gradient values at each iteration. For the quadratic case the three expressions for β_k are exactly equal. However, this is not the case for a general nonlinear objective function.

We need a few more slight modifications to apply the algorithm to general nonlinear functions in practice. First, as mentioned in our discussion of the steepest descent algorithm (Section 8.2), the stopping criterion $\nabla f(\boldsymbol{x}^{(k+1)}) = \mathbf{0}$ is not practical. A suitable practical stopping criterion, such as those discussed in Section 8.2, needs to be used.

For nonquadratic problems, the algorithm will not usually converge in n steps, and as the algorithm progresses, the "$\boldsymbol{Q}$-conjugacy" of the direction vectors will tend to deteriorate. Thus, a common practice is to reinitialize the direction vector to the negative gradient after every few iterations (e.g., n or $n+1$) and continue until the algorithm satisfies the stopping criterion.

A very important issue in minimization problems of nonquadratic functions is the line search. The purpose of the line search is to minimize $\phi_k(\alpha) = f(\boldsymbol{x}^{(k)} + \alpha \boldsymbol{d}^{(k)})$ with respect to $\alpha \geq 0$. A typical approach is to bracket or box in the minimizer and then estimate it. The accuracy of the line search is a critical factor in the performance of the conjugate gradient algorithm. If the line search is known to be inaccurate, the Hestenes-Stiefel formula for β_k is recommended [69].

In general, the choice of which formula for β_k to use depends on the objective function. For example, the Polak-Ribière formula is known to perform far better than the Fletcher-Reeves formula in some cases but not in others. In fact, there are cases in which the $\boldsymbol{g}^{(k)}$, $k = 1, 2, \ldots$, are bounded away from zero when the Polak-Ribière formula is used (see [107]). In the study by Powell in [107], a global convergence analysis suggests that the Fletcher-Reeves formula for β_k is superior. Powell further suggests another formula for β_k:

$$\beta_k = \max\left\{0, \frac{\boldsymbol{g}^{(k+1)\top}[\boldsymbol{g}^{(k+1)} - \boldsymbol{g}^{(k)}]}{\boldsymbol{g}^{(k)\top}\boldsymbol{g}^{(k)}}\right\}.$$

For general results on the convergence of conjugate gradient methods, we refer the reader to [135]. For an application of conjugate gradient algorithms to Wiener filtering, see [116], [117], and [118].

Conjugate gradient algorithms are related to *Krylov subspace methods* (see Exercise 10.6). Krylov-subspace-iteration methods, initiated by Magnus Hestenes, Eduard Stiefel, and Cornelius Lanczos, have been declared one of the 10 algorithms with the greatest influence on the development and practice of science and engineering in the twentieth century [40].

For control perspective on the conjugate gradient algorithm, derived from a proportional-plus-derivative (PD) controller architecture, see [4]. In addition, these authors offer a control perspective on Krylov-subspace-iteration methods as discrete feedback control systems.

EXERCISES

10.1 (Adopted from [88, Exercise 9.8(1)]) Let $\boldsymbol{Q}$ be a real symmetric positive definite $n \times n$ matrix. Given an arbitrary set of linearly independent vectors $\{\boldsymbol{p}^{(0)}, \ldots, \boldsymbol{p}^{(n-1)}\}$ in $\mathbb{R}^n$, the *Gram-Schmidt procedure* generates a set of vectors $\{\boldsymbol{d}^{(0)}, \ldots, \boldsymbol{d}^{(n-1)}\}$ as follows:

$$\boldsymbol{d}^{(0)} = \boldsymbol{p}^{(0)},$$

$$\boldsymbol{d}^{(k+1)} = \boldsymbol{p}^{(k+1)} - \sum_{i=0}^{k} \frac{\boldsymbol{p}^{(k+1)\top} \boldsymbol{Q} \boldsymbol{d}^{(i)}}{\boldsymbol{d}^{(i)\top} \boldsymbol{Q} \boldsymbol{d}^{(i)}} \boldsymbol{d}^{(i)}.$$

Show that the vectors $\boldsymbol{d}^{(0)}, \ldots, \boldsymbol{d}^{(n-1)}$ are $\boldsymbol{Q}$-conjugate.

10.2 Let $f : \mathbb{R}^n \to \mathbb{R}$ be the quadratic function

$$f(\boldsymbol{x}) = \frac{1}{2} \boldsymbol{x}^\top \boldsymbol{Q} \boldsymbol{x} - \boldsymbol{x}^\top \boldsymbol{b},$$

where $\boldsymbol{Q} = \boldsymbol{Q}^\top > 0$. Given a set of directions $\{\boldsymbol{d}^{(0)}, \boldsymbol{d}^{(1)}, \ldots\} \subset \mathbb{R}^n$, consider the algorithm

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \alpha_k \boldsymbol{d}^{(k)},$$

where α_k is the step size. Suppose that $\boldsymbol{g}^{(k+1)\top} \boldsymbol{d}^{(i)} = 0$ for all $k = 0, \ldots, n-1$ and $i = 0, \ldots, k$, where $\boldsymbol{g}^{(k+1)} = \nabla f(\boldsymbol{x}^{(k+1)})$. Show that if $\boldsymbol{g}^{(k)\top} \boldsymbol{d}^{(k)} \neq 0$ for all $k = 0, \ldots, n-1$, then $\boldsymbol{d}^{(0)}, \ldots, \boldsymbol{d}^{(n-1)}$ are $\boldsymbol{Q}$-conjugate.

10.3 Let $f : \mathbb{R}^n \to \mathbb{R}$ be given by $f(\boldsymbol{x}) = \frac{1}{2} \boldsymbol{x}^\top \boldsymbol{Q} \boldsymbol{x} - \boldsymbol{x}^\top \boldsymbol{b}$, where $\boldsymbol{b} \in \mathbb{R}^n$ and $\boldsymbol{Q}$ is a real symmetric positive definite $n \times n$ matrix. Show that in the conjugate gradient method for this f, $\boldsymbol{d}^{(k)\top} \boldsymbol{Q} \boldsymbol{d}^{(k)} = -\boldsymbol{d}^{(k)\top} \boldsymbol{Q} \boldsymbol{g}^{(k)}$.

10.4 Let $\boldsymbol{Q}$ be a real $n \times n$ symmetric matrix.

a. Show that there exists a $\boldsymbol{Q}$-conjugate set $\{\boldsymbol{d}^{(1)}, \ldots, \boldsymbol{d}^{(n)}\}$ such that each $\boldsymbol{d}^{(i)}$ $(i = 1, \ldots, n)$ is an eigenvector of $\boldsymbol{Q}$.
Hint: Use the fact that for any real symmetric $n \times n$ matrix, there exists a set $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n\}$ of its eigenvectors such that $\boldsymbol{v}_i^\top \boldsymbol{v}_j = 0$ for all $i, j = 1, \ldots, n$, $i \neq j$.

b. Suppose that $\boldsymbol{Q}$ is positive definite. Show that if $\{\boldsymbol{d}^{(1)}, \ldots, \boldsymbol{d}^{(n)}\}$ is a $\boldsymbol{Q}$-conjugate set that is also orthogonal (i.e., $\boldsymbol{d}^{(i)\top} \boldsymbol{d}^{(j)} = 0$ for all $i, j = 1, \ldots, n$, $i \neq j$), and $\boldsymbol{d}^{(i)} \neq \boldsymbol{0}$, $i = 1, \ldots, n$, then each $\boldsymbol{d}^{(i)}$, $i = 1, \ldots, n$, is an eigenvector of $\boldsymbol{Q}$.

10.5 Consider the following algorithm for minimizing a function f:

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \alpha_k \boldsymbol{d}^{(k)},$$

where $\alpha_k = \arg\min_\alpha f(\boldsymbol{x}^{(k)} + \alpha \boldsymbol{d}^{(k)})$. Let $\boldsymbol{g}^{(k)} = \nabla f(\boldsymbol{x}^{(k)})$ (as usual).

Suppose that f is quadratic with Hessian $\boldsymbol{Q}$. We choose $\boldsymbol{d}^{(k+1)} = \gamma_k \boldsymbol{g}^{(k+1)} + \boldsymbol{d}^{(k)}$, and we wish the directions $\boldsymbol{d}^{(k)}$ and $\boldsymbol{d}^{(k+1)}$ to be $\boldsymbol{Q}$-conjugate. Find a formula for γ_k in terms of $\boldsymbol{d}^{(k)}$, $\boldsymbol{g}^{(k+1)}$, and $\boldsymbol{Q}$.

10.6 Consider the algorithm

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \alpha_k \boldsymbol{d}^{(k)},$$

with $\alpha_k \in \mathbb{R}$ scalar and $\boldsymbol{x}^{(0)} = \boldsymbol{0}$, applied to the quadratic function $f : \mathbb{R}^n \to \mathbb{R}$ given by

$$f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{Q}\boldsymbol{x} - \boldsymbol{b}^\top \boldsymbol{x},$$

where $\boldsymbol{Q} > 0$. As usual, write $\boldsymbol{g}^{(k)} = \nabla f(\boldsymbol{x}^{(k)})$. Suppose that the search directions are generated according to

$$\boldsymbol{d}^{(k+1)} = a_k \boldsymbol{g}^{(k+1)} + b_k \boldsymbol{d}^{(k)},$$

where a_k and b_k are real constants, and by convention we take $\boldsymbol{d}^{(-1)} = \boldsymbol{0}$.

a. Define the subspace $\mathcal{V}_k = \text{span}[\boldsymbol{b}, \boldsymbol{Q}\boldsymbol{b}, \ldots, \boldsymbol{Q}^{k-1}\boldsymbol{b}]$ (called the *Krylov subspace of order k*). Show that $\boldsymbol{d}^{(k)} \in \mathcal{V}_{k+1}$ and $\boldsymbol{x}^{(k)} \in \mathcal{V}_k$.
Hint: Use induction. Note that $\mathcal{V}_0 = \{\boldsymbol{0}\}$ and $\mathcal{V}_1 = \text{span}[\boldsymbol{b}]$.

b. In light of part a, what can you say about the "optimality" of the conjugate gradient algorithm with respect to the Krylov subspace?

10.7 Consider the quadratic function $f : \mathbb{R}^n \to \mathbb{R}$ given by

$$f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{Q}\boldsymbol{x} - \boldsymbol{x}^\top \boldsymbol{b},$$

where $\boldsymbol{Q} = \boldsymbol{Q}^\top > 0$. Let $\boldsymbol{D} \in \mathbb{R}^{n \times r}$ be of rank r and $\boldsymbol{x}_0 \in \mathbb{R}^n$. Define the function $\phi : \mathbb{R}^r \to \mathbb{R}$ by

$$\phi(\boldsymbol{a}) = f(\boldsymbol{x}_0 + \boldsymbol{D}\boldsymbol{a}).$$

Show that ϕ is a quadratic function with a positive definite quadratic term.

10.8 Consider a conjugate gradient algorithm applied to a quadratic function.

a. Show that the gradients associated with the algorithm are mutually orthogonal. Specifically, show that $\boldsymbol{g}^{(k+1)\top} \boldsymbol{g}^{(i)} = 0$ for all $0 \le k \le n-1$ and $0 \le i \le k$.
Hint: Write $\boldsymbol{g}^{(i)}$ in terms of $\boldsymbol{d}^{(i)}$ and $\boldsymbol{d}^{(i-1)}$.

b. Show that the gradients associated with the algorithm are $\boldsymbol{Q}$-conjugate if separated by at least two iterations. Specifically, show that $\boldsymbol{g}^{(k+1)\top} \boldsymbol{Q} \boldsymbol{g}^{(i)} = 0$ for all $0 \le k \le n-1$ and $0 \le i \le k-1$.

10.9 Represent the function

$$f(x_1, x_2) = \frac{5}{2}x_1^2 + x_2^2 - 3x_1x_2 - x_2 - 7$$

in the form $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{Q}\boldsymbol{x} - \boldsymbol{x}^\top \boldsymbol{b} + c$. Then use the *conjugate gradient algorithm* to construct a vector $\boldsymbol{d}^{(1)}$ that is $\boldsymbol{Q}$-conjugate with $\boldsymbol{d}^{(0)} = \nabla f(\boldsymbol{x}^{(0)})$, where $\boldsymbol{x}^{(0)} = \boldsymbol{0}$.

10.10 Let $f(\boldsymbol{x})$, $\boldsymbol{x} = [x_1, x_2]^\top \in \mathbb{R}^2$, be given by

$$f(\boldsymbol{x}) = \frac{5}{2}x_1^2 + \frac{1}{2}x_2^2 + 2x_1x_2 - 3x_1 - x_2.$$

a. Express $f(\boldsymbol{x})$ in the form of $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{Q}\boldsymbol{x} - \boldsymbol{x}^\top \boldsymbol{b}$.

b. Find the minimizer of f using the conjugate gradient algorithm. Use a starting point of $\boldsymbol{x}^{(0)} = [0, 0]^\top$.

c. Calculate the minimizer of f analytically from $\boldsymbol{Q}$ and $\boldsymbol{b}$, and check it with your answer in part b.

10.11 Write a MATLAB program to implement the conjugate gradient algorithm for general functions. Use the secant method for the line search (e.g., the MATLAB function of Exercise 7.11). Test the different formulas for β_k on Rosenbrock's function (see Exercise 9.4) with an initial condition $\boldsymbol{x}^{(0)} = [-2, 2]^\top$. For this exercise, reinitialize the update direction to the negative gradient every six iterations.

CHAPTER 11

QUASI-NEWTON METHODS

11.1 Introduction

Newton's method is one of the more successful algorithms for optimization. If it converges, it has a quadratic order of convergence. However, as pointed out before, for a general nonlinear objective function, convergence to a solution cannot be guaranteed from an arbitrary initial point $\boldsymbol{x}^{(0)}$. In general, if the initial point is not sufficiently close to the solution, then the algorithm may not possess the descent property [i.e., $f(\boldsymbol{x}^{(k+1)}) \not< f(\boldsymbol{x}^{(k)})$ for some k].

Recall that the idea behind Newton's method is to locally approximate the function f being minimized, at every iteration, by a quadratic function. The minimizer for the quadratic approximation is used as the starting point for the next iteration. This leads to Newton's recursive algorithm

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \boldsymbol{F}(\boldsymbol{x}^{(k)})^{-1}\boldsymbol{g}^{(k)}.$$

We may try to guarantee that the algorithm has the descent property by modifying the original algorithm as follows:

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha_k \boldsymbol{F}(\boldsymbol{x}^{(k)})^{-1}\boldsymbol{g}^{(k)},$$

An Introduction to Optimization, Fourth Edition.
By E. K. P. Chong and S. H. Żak.

where α_k is chosen to ensure that

$$f(\boldsymbol{x}^{(k+1)}) < f(\boldsymbol{x}^{(k)}).$$

For example, we may choose $\alpha_k = \arg\min_{\alpha \geq 0} f(\boldsymbol{x}^{(k)} - \alpha \boldsymbol{F}(\boldsymbol{x}^{(k)})^{-1}\boldsymbol{g}^{(k)})$ (see Theorem 9.2). We can then determine an appropriate value of α_k by performing a line search in the direction $-\boldsymbol{F}(\boldsymbol{x}^{(k)})^{-1}\boldsymbol{g}^{(k)}$. Note that although the line search is simply the minimization of the real variable function $\phi_k(\alpha) = f(\boldsymbol{x}^{(k)} - \alpha \boldsymbol{F}(\boldsymbol{x}^{(k)})^{-1}\boldsymbol{g}^{(k)})$, it is not a trivial problem to solve.

A computational drawback of Newton's method is the need to evaluate $\boldsymbol{F}(\boldsymbol{x}^{(k)})$ and solve the equation $\boldsymbol{F}(\boldsymbol{x}^{(k)})\boldsymbol{d}^{(k)} = -\boldsymbol{g}^{(k)}$ [i.e., compute $\boldsymbol{d}^{(k)} = -\boldsymbol{F}(\boldsymbol{x}^{(k)})^{-1}\boldsymbol{g}^{(k)}$]. To avoid the computation of $\boldsymbol{F}(\boldsymbol{x}^{(k)})^{-1}$, the quasi-Newton methods use an approximation to $\boldsymbol{F}(\boldsymbol{x}^{(k)})^{-1}$ in place of the true inverse. This approximation is updated at every stage so that it exhibits at least some properties of $\boldsymbol{F}(\boldsymbol{x}^{(k)})^{-1}$. To get some idea about the properties that an approximation to $\boldsymbol{F}(\boldsymbol{x}^{(k)})^{-1}$ should satisfy, consider the formula

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha \boldsymbol{H}_k \boldsymbol{g}^{(k)},$$

where $\boldsymbol{H}_k$ is an $n \times n$ real matrix and α is a positive search parameter. Expanding f about $\boldsymbol{x}^{(k)}$ yields

$$\begin{aligned} f(\boldsymbol{x}^{(k+1)}) &= f(\boldsymbol{x}^{(k)}) + \boldsymbol{g}^{(k)\top}(\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)}) + o(\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)}\|) \\ &= f(\boldsymbol{x}^{(k)}) - \alpha \boldsymbol{g}^{(k)\top}\boldsymbol{H}_k\boldsymbol{g}^{(k)} + o(\|\boldsymbol{H}_k\boldsymbol{g}^{(k)}\|\alpha). \end{aligned}$$

As α tends to zero, the second term on the right-hand side of this equation dominates the third. Thus, to guarantee a decrease in f for small α, we have to have

$$\boldsymbol{g}^{(k)\top}\boldsymbol{H}_k\boldsymbol{g}^{(k)} > 0.$$

A simple way to ensure this is to require that $\boldsymbol{H}_k$ be positive definite. We have proved the following result.

Proposition 11.1 *Let $f \in \mathcal{C}^1$, $\boldsymbol{x}^{(k)} \in \mathbb{R}^n$, $\boldsymbol{g}^{(k)} = \nabla f(\boldsymbol{x}^{(k)}) \neq \boldsymbol{0}$, and $\boldsymbol{H}_k$ an $n \times n$ real symmetric positive definite matrix. If we set $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha_k \boldsymbol{H}_k \boldsymbol{g}^{(k)}$, where $\alpha_k = \arg\min_{\alpha \geq 0} f(\boldsymbol{x}^{(k)} - \alpha \boldsymbol{H}_k \boldsymbol{g}^{(k)})$, then $\alpha_k > 0$ and $f(\boldsymbol{x}^{(k+1)}) < f(\boldsymbol{x}^{(k)})$.* □

In constructing an approximation to the inverse of the Hessian matrix, we should use only the objective function and gradient values. Thus, if we can find a suitable method of choosing $\boldsymbol{H}_k$, the iteration may be carried out without any evaluation of the Hessian and without the solution of any set of linear equations.

11.2 Approximating the Inverse Hessian

Let $\boldsymbol{H}_0, \boldsymbol{H}_1, \boldsymbol{H}_2, \ldots$ be successive approximations of the inverse $\boldsymbol{F}(\boldsymbol{x}^{(k)})^{-1}$ of the Hessian. We now derive a condition that the approximations should

satisfy, which forms the starting point for our subsequent discussion of quasi-Newton algorithms. To begin, suppose first that the Hessian matrix $\boldsymbol{F}(\boldsymbol{x})$ of the objective function f is constant and independent of $\boldsymbol{x}$. In other words, the objective function is quadratic, with Hessian $\boldsymbol{F}(\boldsymbol{x}) = \boldsymbol{Q}$ for all $\boldsymbol{x}$, where $\boldsymbol{Q} = \boldsymbol{Q}^\top$. Then,

$$\boldsymbol{g}^{(k+1)} - \boldsymbol{g}^{(k)} = \boldsymbol{Q}(\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)}).$$

Let

$$\Delta \boldsymbol{g}^{(k)} \triangleq \boldsymbol{g}^{(k+1)} - \boldsymbol{g}^{(k)}$$

and

$$\Delta \boldsymbol{x}^{(k)} \triangleq \boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)}.$$

Then, we may write

$$\Delta \boldsymbol{g}^{(k)} = \boldsymbol{Q} \Delta \boldsymbol{x}^{(k)}.$$

We start with a real symmetric positive definite matrix $\boldsymbol{H}_0$. Note that given k, the matrix $\boldsymbol{Q}^{-1}$ satisfies

$$\boldsymbol{Q}^{-1} \Delta \boldsymbol{g}^{(i)} = \Delta \boldsymbol{x}^{(i)}, \qquad 0 \leq i \leq k.$$

Therefore, we also impose the requirement that the approximation $\boldsymbol{H}_{k+1}$ of the Hessian satisfy

$$\boldsymbol{H}_{k+1} \Delta \boldsymbol{g}^{(i)} = \Delta \boldsymbol{x}^{(i)}, \qquad 0 \leq i \leq k.$$

If n steps are involved, then moving in n directions $\Delta \boldsymbol{x}^{(0)}, \Delta \boldsymbol{x}^{(1)}, \ldots, \Delta \boldsymbol{x}^{(n-1)}$ yields

$$\begin{aligned}
\boldsymbol{H}_n \Delta \boldsymbol{g}^{(0)} &= \Delta \boldsymbol{x}^{(0)}, \\
\boldsymbol{H}_n \Delta \boldsymbol{g}^{(1)} &= \Delta \boldsymbol{x}^{(1)}, \\
&\vdots \\
\boldsymbol{H}_n \Delta \boldsymbol{g}^{(n-1)} &= \Delta \boldsymbol{x}^{(n-1)}.
\end{aligned}$$

This set of equations can be represented as

$$\boldsymbol{H}_n [\Delta \boldsymbol{g}^{(0)}, \Delta \boldsymbol{g}^{(1)}, \ldots, \Delta \boldsymbol{g}^{(n-1)}] = [\Delta \boldsymbol{x}^{(0)}, \Delta \boldsymbol{x}^{(1)}, \ldots, \Delta \boldsymbol{x}^{(n-1)}].$$

Note that $\boldsymbol{Q}$ satisfies

$$\boldsymbol{Q} [\Delta \boldsymbol{x}^{(0)}, \Delta \boldsymbol{x}^{(1)}, \ldots, \Delta \boldsymbol{x}^{(n-1)}] = [\Delta \boldsymbol{g}^{(0)}, \Delta \boldsymbol{g}^{(1)}, \ldots, \Delta \boldsymbol{g}^{(n-1)}]$$

and

$$\boldsymbol{Q}^{-1} [\Delta \boldsymbol{g}^{(0)}, \Delta \boldsymbol{g}^{(1)}, \ldots, \Delta \boldsymbol{g}^{(n-1)}] = [\Delta \boldsymbol{x}^{(0)}, \Delta \boldsymbol{x}^{(1)}, \ldots, \Delta \boldsymbol{x}^{(n-1)}].$$

Therefore, if $[\Delta \boldsymbol{g}^{(0)}, \Delta \boldsymbol{g}^{(1)}, \ldots, \Delta \boldsymbol{g}^{(n-1)}]$ is nonsingular, then $\boldsymbol{Q}^{-1}$ is determined uniquely after n steps, via

$$\boldsymbol{Q}^{-1} = \boldsymbol{H}_n = [\Delta \boldsymbol{x}^{(0)}, \Delta \boldsymbol{x}^{(1)}, \ldots, \Delta \boldsymbol{x}^{(n-1)}][\Delta \boldsymbol{g}^{(0)}, \Delta \boldsymbol{g}^{(1)}, \ldots, \Delta \boldsymbol{g}^{(n-1)}]^{-1}.$$

As a consequence, we conclude that if $\boldsymbol{H}_n$ satisfies the equations $\boldsymbol{H}_n \Delta \boldsymbol{g}^{(i)} = \Delta \boldsymbol{x}^{(i)}$, $0 \leq i \leq n-1$, then the algorithm $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha_k \boldsymbol{H}_k \boldsymbol{g}^{(k)}$, $\alpha_k = \arg\min_{\alpha \geq 0} f(\boldsymbol{x}^{(k)} - \alpha \boldsymbol{H}_k \boldsymbol{g}^{(k)})$, is guaranteed to solve problems with quadratic objective functions in $n+1$ steps, because the update $\boldsymbol{x}^{(n+1)} = \boldsymbol{x}^{(n)} - \alpha_n \boldsymbol{H}_n \boldsymbol{g}^{(n)}$ is equivalent to Newton's algorithm. In fact, as we shall see below (Theorem 11.1), such algorithms solve quadratic problems of n variables in at most n steps.

The considerations above illustrate the basic idea behind the quasi-Newton methods. Specifically, quasi-Newton algorithms have the form

$$\begin{aligned} \boldsymbol{d}^{(k)} &= -\boldsymbol{H}_k \boldsymbol{g}^{(k)}, \\ \alpha_k &= \arg\min_{\alpha \geq 0} f(\boldsymbol{x}^{(k)} + \alpha \boldsymbol{d}^{(k)}), \\ \boldsymbol{x}^{(k+1)} &= \boldsymbol{x}^{(k)} + \alpha_k \boldsymbol{d}^{(k)}, \end{aligned}$$

where the matrices $\boldsymbol{H}_0, \boldsymbol{H}_1, \ldots$ are symmetric. In the quadratic case these matrices are required to satisfy

$$\boldsymbol{H}_{k+1} \Delta \boldsymbol{g}^{(i)} = \Delta \boldsymbol{x}^{(i)}, \qquad 0 \leq i \leq k,$$

where $\Delta \boldsymbol{x}^{(i)} = \boldsymbol{x}^{(i+1)} - \boldsymbol{x}^{(i)} = \alpha_i \boldsymbol{d}^{(i)}$ and $\Delta \boldsymbol{g}^{(i)} = \boldsymbol{g}^{(i+1)} - \boldsymbol{g}^{(i)} = \boldsymbol{Q} \Delta \boldsymbol{x}^{(i)}$. It turns out that quasi-Newton methods are also conjugate direction methods, as stated in the following.

Theorem 11.1 *Consider a quasi-Newton algorithm applied to a quadratic function with Hessian $\boldsymbol{Q} = \boldsymbol{Q}^\top$ such that for $0 \leq k < n-1$,*

$$\boldsymbol{H}_{k+1} \Delta \boldsymbol{g}^{(i)} = \Delta \boldsymbol{x}^{(i)}, \qquad 0 \leq i \leq k,$$

where $\boldsymbol{H}_{k+1} = \boldsymbol{H}_{k+1}^\top$. If $\alpha_i \neq 0$, $0 \leq i \leq k$, then $\boldsymbol{d}^{(0)}, \ldots, \boldsymbol{d}^{(k+1)}$ are $\boldsymbol{Q}$-conjugate. □

Proof. We proceed by induction. We begin with the $k = 0$ case: that $\boldsymbol{d}^{(0)}$ and $\boldsymbol{d}^{(1)}$ are $\boldsymbol{Q}$-conjugate. Because $\alpha_0 \neq 0$, we can write $\boldsymbol{d}^{(0)} = \Delta \boldsymbol{x}^{(0)} / \alpha_0$. Hence,

$$\begin{aligned} \boldsymbol{d}^{(1)\top} \boldsymbol{Q} \boldsymbol{d}^{(0)} &= -\boldsymbol{g}^{(1)\top} \boldsymbol{H}_1 \boldsymbol{Q} \boldsymbol{d}^{(0)} \\ &= -\boldsymbol{g}^{(1)\top} \boldsymbol{H}_1 \frac{\boldsymbol{Q} \Delta \boldsymbol{x}^{(0)}}{\alpha_0} \\ &= -\boldsymbol{g}^{(1)\top} \frac{\boldsymbol{H}_1 \Delta \boldsymbol{g}^{(0)}}{\alpha_0} \\ &= -\boldsymbol{g}^{(1)\top} \frac{\Delta \boldsymbol{x}^{(0)}}{\alpha_0} \\ &= -\boldsymbol{g}^{(1)\top} \boldsymbol{d}^{(0)}. \end{aligned}$$

But $\boldsymbol{g}^{(1)\top}\boldsymbol{d}^{(0)} = 0$ as a consequence of $\alpha_0 > 0$ being the minimizer of $\phi(\alpha) = f(\boldsymbol{x}^{(0)} + \alpha\boldsymbol{d}^{(0)})$ (see Exercise 11.1). Hence, $\boldsymbol{d}^{(1)\top}\boldsymbol{Q}\boldsymbol{d}^{(0)} = 0$.

Assume that the result is true for $k-1$ (where $k < n-1$). We now prove the result for k, that is, that $\boldsymbol{d}^{(0)}, \ldots, \boldsymbol{d}^{(k+1)}$ are $\boldsymbol{Q}$-conjugate. It suffices to show that $\boldsymbol{d}^{(k+1)\top}\boldsymbol{Q}\boldsymbol{d}^{(i)} = 0$, $0 \le i \le k$. Given i, $0 \le i \le k$, using the same algebraic steps as in the $k = 0$ case, and using the assumption that $\alpha_i \neq 0$, we obtain

$$\begin{aligned}\boldsymbol{d}^{(k+1)\top}\boldsymbol{Q}\boldsymbol{d}^{(i)} &= -\boldsymbol{g}^{(k+1)\top}\boldsymbol{H}_{k+1}\boldsymbol{Q}\boldsymbol{d}^{(i)} \\ &\vdots \\ &= -\boldsymbol{g}^{(k+1)\top}\boldsymbol{d}^{(i)}.\end{aligned}$$

Because $\boldsymbol{d}^{(0)}, \ldots, \boldsymbol{d}^{(k)}$ are $\boldsymbol{Q}$-conjugate by assumption, we conclude from Lemma 10.2 that $\boldsymbol{g}^{(k+1)\top}\boldsymbol{d}^{(i)} = 0$. Hence, $\boldsymbol{d}^{(k+1)\top}\boldsymbol{Q}\boldsymbol{d}^{(i)} = 0$, which completes the proof. ∎

By Theorem 11.1 we conclude that a quasi-Newton algorithm solves a quadratic of n variables in at most n steps.

Note that the equations that the matrices $\boldsymbol{H}_k$ are required to satisfy do not determine those matrices uniquely. Thus, we have some freedom in the way we compute the $\boldsymbol{H}_k$. In the methods we describe, we compute $\boldsymbol{H}_{k+1}$ by adding a correction to $\boldsymbol{H}_k$. In the following sections we consider three specific updating formulas.

11.3 The Rank One Correction Formula

In the *rank one correction formula*, the correction term is symmetric and has the form $a_k\boldsymbol{z}^{(k)}\boldsymbol{z}^{(k)\top}$, where $a_k \in \mathbb{R}$ and $\boldsymbol{z}^{(k)} \in \mathbb{R}^n$. Therefore, the update equation is

$$\boldsymbol{H}_{k+1} = \boldsymbol{H}_k + a_k\boldsymbol{z}^{(k)}\boldsymbol{z}^{(k)\top}.$$

Note that

$$\operatorname{rank}\boldsymbol{z}^{(k)}\boldsymbol{z}^{(k)\top} = \operatorname{rank}\left(\begin{bmatrix} z_1^{(k)} \\ \vdots \\ z_n^{(k)} \end{bmatrix}\begin{bmatrix} z_1^{(k)} & \cdots & z_n^{(k)} \end{bmatrix}\right) = 1$$

and hence the name *rank one correction* [it is also called the *single-rank symmetric (SRS) algorithm*]. The product $\boldsymbol{z}^{(k)}\boldsymbol{z}^{(k)\top}$ is sometimes referred to as the *dyadic product* or *outer product*. Observe that if $\boldsymbol{H}_k$ is symmetric, then so is $\boldsymbol{H}_{k+1}$.

Our goal now is to determine a_k and $\boldsymbol{z}^{(k)}$, given $\boldsymbol{H}_k$, $\Delta\boldsymbol{g}^{(k)}$, $\Delta\boldsymbol{x}^{(k)}$, so that the required relationship discussed in Section 11.2 is satisfied; namely,

$\boldsymbol{H}_{k+1}\Delta\boldsymbol{g}^{(i)} = \Delta\boldsymbol{x}^{(i)}$, $i = 1, \ldots, k$. To begin, let us first consider the condition $\boldsymbol{H}_{k+1}\Delta\boldsymbol{g}^{(k)} = \Delta\boldsymbol{x}^{(k)}$. In other words, given $\boldsymbol{H}_k$, $\Delta\boldsymbol{g}^{(k)}$, and $\Delta\boldsymbol{x}^{(k)}$, we wish to find a_k and $\boldsymbol{z}^{(k)}$ to ensure that

$$\boldsymbol{H}_{k+1}\Delta\boldsymbol{g}^{(k)} = (\boldsymbol{H}_k + a_k\boldsymbol{z}^{(k)}\boldsymbol{z}^{(k)\top})\Delta\boldsymbol{g}^{(k)} = \Delta\boldsymbol{x}^{(k)}.$$

First note that $\boldsymbol{z}^{(k)\top}\Delta\boldsymbol{g}^{(k)}$ is a scalar. Thus,

$$\Delta\boldsymbol{x}^{(k)} - \boldsymbol{H}_k\Delta\boldsymbol{g}^{(k)} = (a_k\boldsymbol{z}^{(k)\top}\Delta\boldsymbol{g}^{(k)})\boldsymbol{z}^{(k)},$$

and hence

$$\boldsymbol{z}^{(k)} = \frac{\Delta\boldsymbol{x}^{(k)} - \boldsymbol{H}_k\Delta\boldsymbol{g}^{(k)}}{a_k(\boldsymbol{z}^{(k)\top}\Delta\boldsymbol{g}^{(k)})}.$$

We can now determine

$$a_k\boldsymbol{z}^{(k)}\boldsymbol{z}^{(k)\top} = \frac{(\Delta\boldsymbol{x}^{(k)} - \boldsymbol{H}_k\Delta\boldsymbol{g}^{(k)})(\Delta\boldsymbol{x}^{(k)} - \boldsymbol{H}_k\Delta\boldsymbol{g}^{(k)})^\top}{a_k(\boldsymbol{z}^{(k)\top}\Delta\boldsymbol{g}^{(k)})^2}.$$

Hence,

$$\boldsymbol{H}_{k+1} = \boldsymbol{H}_k + \frac{(\Delta\boldsymbol{x}^{(k)} - \boldsymbol{H}_k\Delta\boldsymbol{g}^{(k)})(\Delta\boldsymbol{x}^{(k)} - \boldsymbol{H}_k\Delta\boldsymbol{g}^{(k)})^\top}{a_k(\boldsymbol{z}^{(k)\top}\Delta\boldsymbol{g}^{(k)})^2}.$$

The next step is to express the denominator of the second term on the right-hand side of the equation above as a function of the given quantities $\boldsymbol{H}_k$, $\Delta\boldsymbol{g}^{(k)}$, and $\Delta\boldsymbol{x}^{(k)}$. To accomplish this, premultiply $\Delta\boldsymbol{x}^{(k)} - \boldsymbol{H}_k\Delta\boldsymbol{g}^{(k)} = (a_k\boldsymbol{z}^{(k)\top}\Delta\boldsymbol{g}^{(k)})\boldsymbol{z}^{(k)}$ by $\Delta\boldsymbol{g}^{(k)\top}$ to obtain

$$\Delta\boldsymbol{g}^{(k)\top}\Delta\boldsymbol{x}^{(k)} - \Delta\boldsymbol{g}^{(k)\top}\boldsymbol{H}_k\Delta\boldsymbol{g}^{(k)} = \Delta\boldsymbol{g}^{(k)\top}a_k\boldsymbol{z}^{(k)}\boldsymbol{z}^{(k)\top}\Delta\boldsymbol{g}^{(k)}.$$

Observe that a_k is a scalar and so is $\Delta\boldsymbol{g}^{(k)\top}\boldsymbol{z}^{(k)} = \boldsymbol{z}^{(k)\top}\Delta\boldsymbol{g}^{(k)}$. Thus,

$$\Delta\boldsymbol{g}^{(k)\top}\Delta\boldsymbol{x}^{(k)} - \Delta\boldsymbol{g}^{(k)\top}\boldsymbol{H}_k\Delta\boldsymbol{g}^{(k)} = a_k(\boldsymbol{z}^{(k)\top}\Delta\boldsymbol{g}^{(k)})^2.$$

Taking this relation into account yields

$$\boldsymbol{H}_{k+1} = \boldsymbol{H}_k + \frac{(\Delta\boldsymbol{x}^{(k)} - \boldsymbol{H}_k\Delta\boldsymbol{g}^{(k)})(\Delta\boldsymbol{x}^{(k)} - \boldsymbol{H}_k\Delta\boldsymbol{g}^{(k)})^\top}{\Delta\boldsymbol{g}^{(k)\top}(\Delta\boldsymbol{x}^{(k)} - \boldsymbol{H}_k\Delta\boldsymbol{g}^{(k)})}.$$

We summarize the above development in the following algorithm.

Rank One Algorithm

1. Set $k := 0$; select $\boldsymbol{x}^{(0)}$ and a real symmetric positive definite $\boldsymbol{H}_0$.
2. If $\boldsymbol{g}^{(k)} = \boldsymbol{0}$, stop; else, $\boldsymbol{d}^{(k)} = -\boldsymbol{H}_k\boldsymbol{g}^{(k)}$.