6

UNCONSTRAINED MULTIVARIABLE OPTIMIZATION

THE NUMERICAL OPTIMIZATION of general nonlinear multivariable objective functions requires efficient and robust techniques. Efficiency is important because these problems require an iterative solution procedure, and trial and error becomes impractical for more than three or four variables. Robustness (the ability to achieve a solution) is desirable because a general nonlinear function is unpredictable in its behavior; there may be relative maxima or minima, saddle points, regions of convexity, concavity, and so on. In some regions the optimization algorithm may progress very slowly toward the optimum, requiring excessive computer time. Fortunately, we can draw on extensive experience in testing nonlinear programming algorithms for unconstrained functions to evaluate various approaches proposed for the optimization of such functions.

In this chapter we discuss the solution of the unconstrained optimization problem:

$$\text{Find:} \quad \mathbf{x}^* = [x_1^* \quad x_2^* \quad \cdots \quad x_n^*]^T \quad \text{that minimizes} \quad f(x_1, x_2, \ldots, x_n) \equiv f(\mathbf{x})$$

Most effective iterative procedures alternate between two phases in the optimization. At iteration k, where the current $\mathbf{x}$ is $\mathbf{x}^k$, they do the following:

1. Choose a search direction $\mathbf{s}^k$
2. Minimize along that direction (usually inexactly) to find a new point

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha^k \mathbf{s}^k$$

where α^k is a positive scalar called the step size. The step size is determined by an optimization process called a line search as described in Chapter 5.

In addition to 1 and 2, an algorithm must specify

3. The initial starting vector $\mathbf{x}^0 = [x_1^0 \quad x_2^0 \quad \cdots \quad x_n^0]^T$ and
4. The convergence criteria for termination.

From a given starting point, a search direction is determined, and $f(\mathbf{x})$ is minimized in that direction. The search stops based on some criteria, and then a new search direction is determined, followed by another line search. The line search can be carried out to various degrees of precision. For example, we could use a simple successive doubling of the step size as a screening method until we detect the optimum has been bracketed. At this point the screening search can be terminated and a more sophisticated method employed to yield a higher degree of accuracy. In any event, refer to the techniques discussed in Chapter 5 for ways to carry out the line search.

The NLP (nonlinear programming) methods to be discussed in this chapter differ mainly in how they generate the search directions. Some nonlinear programming methods require information about derivative values, whereas others do not use derivatives and rely solely on function evaluations. Furthermore, finite difference substitutes can be used in lieu of derivatives as explained in Section 8.10. For differentiable functions, methods that use analytical derivatives almost always use less computation time and are more accurate, even if finite difference approxima-

tions are used. Symbolic codes can be employed to obtain analytical derivatives but this may require more computer time than finite differencing to get derivatives. For nonsmooth functions, a function-values-only method may be more successful than using a derivative-based method. We first describe some simple nonderivative methods and then present a series of methods that use derivative information. We also show how the nature of the objective function influences the effectiveness of the particular optimization algorithm.

6.1 METHODS USING FUNCTION VALUES ONLY

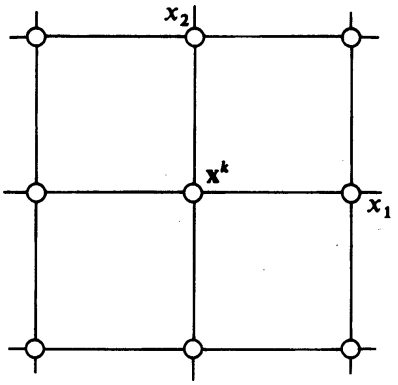
Some methods do not require the use of derivatives in determining the search direction. Under some circumstances the methods described in this section can be used effectively, but they may be inefficient compared with methods discussed in subsequent sections. They have the advantage of being simple to understand and execute.

6.1.1 Random Search

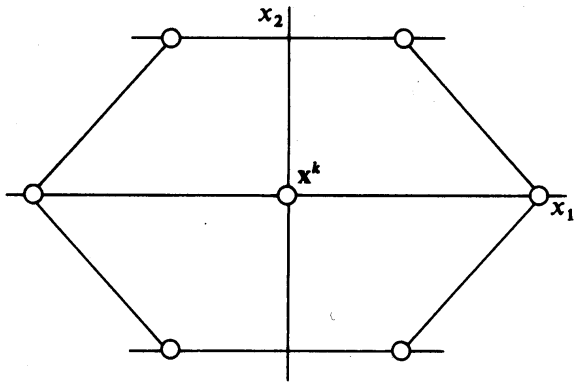
A random search method simply selects a starting vector $\mathbf{x}^0$, evaluates $f(\mathbf{x})$ at $\mathbf{x}^0$, and then randomly selects another vector $\mathbf{x}^1$ and evaluates $f(\mathbf{x})$ at $\mathbf{x}^1$. In effect, both a search direction and step length are chosen simultaneously. After one or more stages, the value of $f(\mathbf{x}^k)$ is compared with the best previous value of $f(\mathbf{x})$ from among the previous stages, and the decision is made to continue or terminate the procedure. Variations of this form of random search involve randomly selecting a search direction and then minimizing (possibly by random steps) in that search direction as a series of cycles. Clearly, *the* optimal solution can be obtained with a probability of 1 only as $k \rightarrow \infty$ but as a practical matter, if the objective function is quite flat, a suboptimal solution may be quite acceptable. Even though the method is inefficient insofar as function evaluations are concerned, it may provide a good starting point for another method. You might view random search as an extension of the case study method. Refer to Dixon and James (1980) for some practical algorithms.

6.1.2 Grid Search

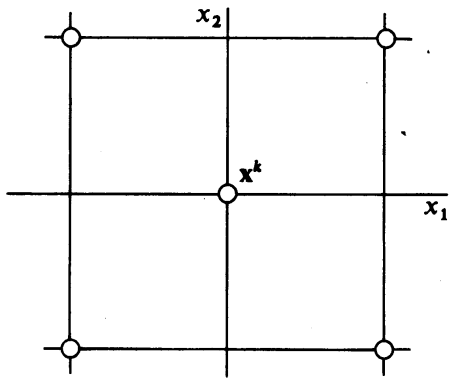
Methods of experimental design discussed in most basic statistics books can be applied equally well to minimizing $f(\mathbf{x})$ (see Chapter 2). You evaluate a series of points about a reference point selected according to some type of design such as the ones shown in Figure 6.1 (for an objective function of two variables). Next you move to the point that improves the objective function the most, and repeat.



(a) Three-level factorial design ($3^2 - 1 = 8$ points plus center)



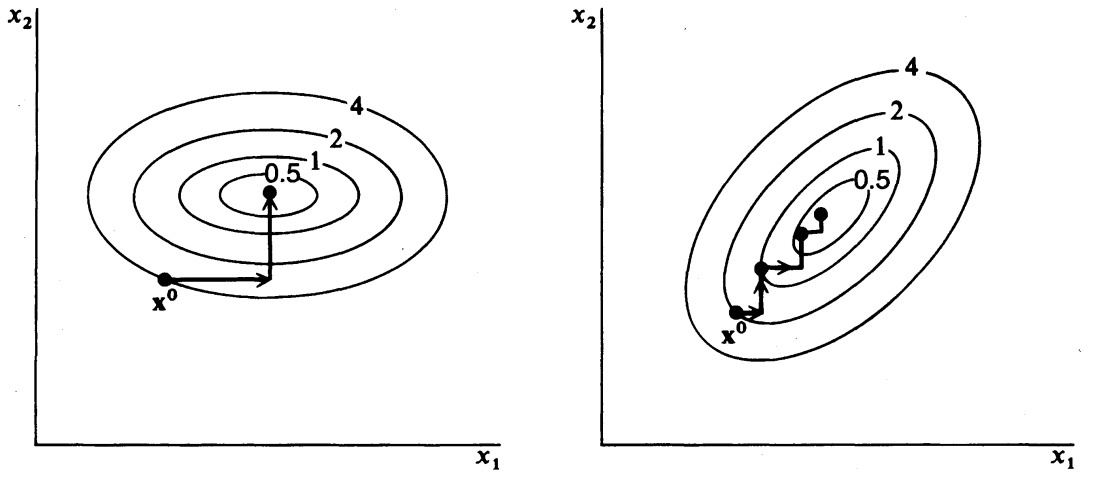
(b) Hexagon design (6 points + center)



(c) Two-level factorial design ($2^2 = 4$ points plus center)

FIGURE 6.1
Various grid search designs to select vectors $\mathbf{x}$ to evaluate $f(\mathbf{x})$.

For $n = 30$, we must examine $3^{30} - 1 = 2.0588 \times 10^{14}$ values of $f(\mathbf{x})$ if a three-level factorial design is to be used, obviously a prohibitive number of function evaluations.



(a) $f_1(\mathbf{x})$ (b) $f_2(\mathbf{x})$

FIGURE 6.2
Execution of a univariate search on two different quadratic functions.

6.1.3 Univariate Search

Another simple optimization technique is to select n fixed search directions (usually the coordinate axes) for an objective function of n variables. Then $f(\mathbf{x})$ is minimized in each search direction sequentially using a one-dimensional search. This method is effective for a quadratic function of the form

$$f_1(\mathbf{x}) = \sum_{i=1}^{n} c_i x_i^2$$

because the search directions line up with the principal axes as indicated in Figure 6.2a. However, it does not perform satisfactorily for more general quadratic objective functions of the form

$$f_2(\mathbf{x}) = \sum_{i=1}^{n} \sum_{j=1}^{n} d_{ij} x_i x_j$$

as illustrated in Figure 6.2b. For the latter case, the changes in $\mathbf{x}$ decrease as the optimum is neared, so many iterations will be required to attain high accuracy.

6.1.4 Simplex Search Method

The method of the "Sequential Simplex" formulated by Spendley, Hext, and Himsworth (1962) selects points at the vertices of the simplex at which to evaluate $f(\mathbf{x})$. In two dimensions the figure is an equilateral triangle. Examine Figure 6.3. In three dimensions this figure becomes a regular tetrahedron, and so on. Each search direction points away from the vertex having the highest value of $f(\mathbf{x})$ to the other vertices in the simplex. Thus, the direction of search changes, but the step size is

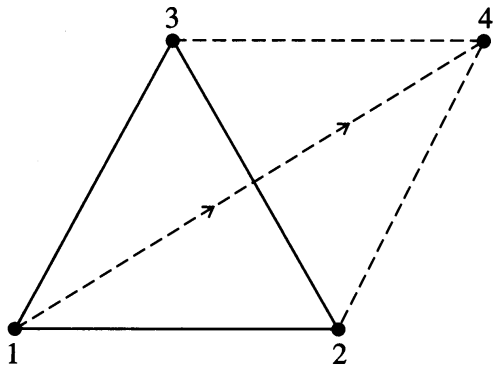


FIGURE 6.3
Reflection to a new point in the simplex method.
At point 1, $f(\mathbf{x})$ is greater than f at points 2 or 3.

fixed for a given size simplex. Let us use a function of two variables to illustrate the procedure.

At each iteration, to minimize $f(\mathbf{x})$, $f(\mathbf{x})$ is evaluated at each of three vertices of the triangle. The direction of search is oriented away from the point with the highest value for the function through the centroid of the simplex. By making the search direction bisect the line between the other two points of the triangle, the direction goes through the centroid. A new point is selected in this reflected direction (as shown in Figure 6.3), preserving the geometric shape. The objective function is then evaluated at the new point, and a new search direction is determined. The method proceeds, rejecting one vertex at a time until the simplex straddles the optimum. Various rules are used to prevent excessive repetition of the same cycle or simplexes.

As the optimum is approached, the last equilateral triangle straddles the optimum point or is within a distance of the order of its own size from the optimum (examine Figure 6.4). The procedure cannot therefore get closer to the optimum and repeats itself so that the simplex size must be reduced, such as halving the length of all the sides of the simplex containing the vertex where the oscillation started. A new simplex composed of the midpoints of the ending simplex is constructed. When the simplex size is smaller than a prescribed tolerance, the routine is stopped. Thus, the optimum position is determined to within a tolerance influenced by the size of the simplex.

Nelder and Mead (1965) described a more efficient (but more complex) version of the simplex method that permitted the geometric figures to expand and contract continuously during the search. Their method minimized a function of n variables using $(n + 1)$ vertices of a flexible polyhedron. Details of the method together with a computer code to execute the algorithm can be found in Avriel (1976).

6.1.5 Conjugate Search Directions

Experience has shown that conjugate directions are much more effective as search directions than arbitrarily chosen search directions, such as in univariate search, or

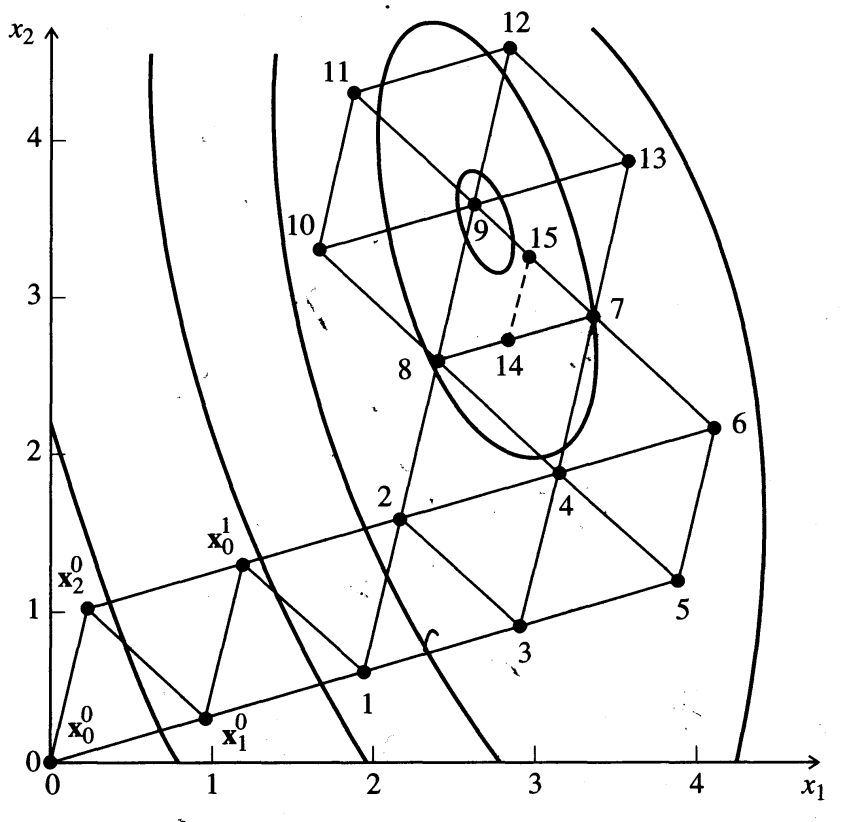


FIGURE 6.4
Progression to the vicinity of the optimum and oscillation around the optimum using the simplex method of search. The original vertices are $\mathbf{x}_0^0$, $\mathbf{x}_1^0$, and $\mathbf{x}_2^0$. The next point (vertex) is $\mathbf{x}_0^1$. Succeeding new vertices are numbered starting with 1 and continuing to 13 at which point a cycle starts to repeat. The size of the simplex is reduced to the triangle determined by points 7, 14, and 15, and then the procedure is continued (not shown).

even orthogonal search directions. Two directions $\mathbf{s}^i$ and $\mathbf{s}^j$ are said to be *conjugate* with respect to a positive-definite matrix $\mathbf{Q}$ if

$$(\mathbf{s}^i)^T\mathbf{Q}(\mathbf{s}^j) = 0 \tag{6.1}$$

In general, a set of n linearly independent directions of search $\mathbf{s}^0, \mathbf{s}^1 \ldots, \mathbf{s}^{n-1}$ are said to be conjugate with respect to a positive-definite square matrix $\mathbf{Q}$ if

$$(\mathbf{s}^i)^T\mathbf{Q}\mathbf{s}^j = 0 \qquad 0 \leq i \neq j \leq n-1 \tag{6.2}$$

In optimization the matrix $\mathbf{Q}$ is the Hessian matrix of the objective function, $\mathbf{H}$. For a *quadratic function* $f(\mathbf{x})$ of n variables, in which $\mathbf{H}$ is a constant matrix, you are guaranteed to reach the minimum of $f(\mathbf{x})$ in n *stages* if you minimize *exactly* on each stage (Dennis and Schnabel, 1996). In n dimensions, many different sets of conjugate directions exist for a given matrix $\mathbf{Q}$. In two dimensions, however, if you choose an initial direction $\mathbf{s}^1$ and $\mathbf{Q}$, $\mathbf{s}^2$ is fully specified as illustrated in Example 6.1.

Orthogonality is a special case of conjugacy because when $\mathbf{Q} = \mathbf{I}$, $(\mathbf{s}^i)^T\mathbf{s}^j = 0$ in Equation (6.2). If the coordinates of $\mathbf{x}$ are translated and rotated by suitable transformations so as to align the new principal axes of $\mathbf{H}(\mathbf{x})$ with the eigenvectors of $\mathbf{H}(\mathbf{x})$ and to place the center of the coordinate system at the stationary point of $f(\mathbf{x})$ (refer to Figures 4.12 through 4.15), then conjugacy can be interpreted as orthogonality in the space of the transformed coordinates.

Although authors and practitioners refer to a class of unconstrained optimization methods as "methods that use conjugate directions," for a general nonlinear function, the conjugate directions exist only for a quadratic approximation of the function at a single stage k. Once the objective function is modeled by a new approximation at stage $(k + 1)$, the directions on stage k are unlikely to be conjugate to any of the directions selected in stage $(k + 1)$.

EXAMPLE 6.1 CALCULATION OF CONJUGATE DIRECTIONS

Suppose we want to minimize $f(\mathbf{x}) = 2x_1^2 + x_2^2 - 3$ starting at $(\mathbf{x}^0)^T = [1\ \ 1]$ with the initial direction being $\mathbf{s}^0 = [-4\ \ -2]^T$. Find a conjugate direction to the initial direction $\mathbf{s}^0$.

Solution

$$\mathbf{s}^0 = -\begin{bmatrix} 4 \\ 2 \end{bmatrix} \qquad \mathbf{H}(\mathbf{x}) = \begin{bmatrix} 4 & 0 \\ 0 & 2 \end{bmatrix}$$

We need to solve Equation (6.2) for $\mathbf{s}^1 = [s_1^1\ \ s_2^1]^T$ with $\mathbf{Q} = \mathbf{H}$ and $\mathbf{s}^0 = [-4\ \ -2]^T$.

$$(-1)\begin{bmatrix} 4 & 2 \end{bmatrix}\begin{bmatrix} 4 & 0 \\ 0 & 2 \end{bmatrix}\begin{bmatrix} s_1^1 \\ s_2^1 \end{bmatrix} = 0$$

Because s_1^1 is not unique, we can pick $s_1^1 = 1$ and determine s_2^1

$$\begin{bmatrix} -16 & -4 \end{bmatrix}\begin{bmatrix} 1 \\ s_2^1 \end{bmatrix} = 0$$

Thus $\mathbf{s}^1 = [1\ -4]^T$ is a direction conjugate to $\mathbf{s}^0 = [-4\ -2]^T$.

We can reach the minimum of $f(\mathbf{x})$ in two stages using first $\mathbf{s}^0$ and then $\mathbf{s}^1$. Can we use the search directions in reverse order? From $\mathbf{x}^0 = [1\ \ 1]^T$ we can carry out a numerical search in the direction $\mathbf{s}^0 = [-4\ \ -2]^T$ to reach the point $\mathbf{x}^1$. Quadratic interpolation can obtain the exact optimal step length because f is quadratic, yielding $\alpha = 0.27778$. Then

$$\mathbf{x}^1 = \mathbf{x}^0 - \alpha^0\mathbf{s}^0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix} + 0.27778\begin{bmatrix} -4 \\ -2 \end{bmatrix} = \begin{bmatrix} -0.1111 \\ 0.4444 \end{bmatrix}$$

For the next stage, the search direction is $\mathbf{s}^1 = [1\ -4]^T$, and the optimal step length calculated by quadratic interpolation is $\alpha^1 = 0.1111$. Hence

$$\mathbf{x}^2 = \mathbf{x}^1 + \alpha^1\mathbf{s}^1 = \begin{bmatrix} -0.1111 \\ 0.4444 \end{bmatrix} + 0.1111\begin{bmatrix} 1 \\ -4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

as expected.

6.1.6 Summary

As mentioned earlier, nonlinear objective functions are sometimes nonsmooth due to the presence of functions like abs, min, max, or if-then-else statements, which can cause derivatives, or the function itself, to be discontinuous at some points. Unconstrained optimization methods that do not use derivatives are often able to solve nonsmooth NLP problems, whereas methods that use derivatives can fail. Methods employing derivatives can get "stuck" at a point of discontinuity, but the function-value-only methods are less affected. For smooth functions, however, methods that use derivatives are both more accurate and faster, and their advantage grows as the number of decision variables increases. Hence, we now turn our attention to unconstrained optimization methods that use only first partial derivatives of the objective function.

6.2 METHODS THAT USE FIRST DERIVATIVES

A good search direction should reduce (for minimization) the objective function so that if $\mathbf{x}^0$ is the original point and $\mathbf{x}^1$ is the new point

$$f(\mathbf{x}^1) < f(\mathbf{x}^0)$$

Such a direction $\mathbf{s}$ is called a descent direction and satisfies the following requirement at any point

$$\nabla^T f(\mathbf{x})\mathbf{s} < 0$$

To see why, examine the two vectors $\nabla f(\mathbf{x}^k)$ and $\mathbf{s}^k$ in Figure 6.5. The angle between them is θ, hence

$$\nabla^T f(\mathbf{x})\mathbf{s}^k = |\nabla f(\mathbf{x}^k)||\mathbf{s}^k| \cos\theta$$

If $\theta = 90°$ as in Figure 6.5, then steps along $\mathbf{s}^k$ do not reduce (improve) the value of $f(\mathbf{x})$. If $0 \leq \theta < 90°$, no improvement is possible and $f(\mathbf{x})$ increases. Only if $\theta > 90°$ does the search direction yield smaller values of $f(\mathbf{x})$, hence $\nabla^T f(\mathbf{x}^k)\mathbf{s}^k < 0$.

We first examine the classic steepest descent method of using the gradient and then examine a conjugate gradient method.

6.2.1 Steepest Descent

The gradient is the vector at a point $\mathbf{x}$ that gives the (local) direction of the greatest rate of increase in $f(\mathbf{x})$. It is orthogonal to the contour of $f(\mathbf{x})$ at $\mathbf{x}$. For maximization, the search direction is simply the gradient (when used the algorithm is called "steepest ascent"); for minimization, the search direction is the negative of the gradient ("steepest descent")

$$\mathbf{s}^k = -\nabla f(\mathbf{x}^k) \tag{6.3}$$

In steepest descent at the kth stage, the transition from the current point $\mathbf{x}^k$ to the new point $\mathbf{x}^{k+1}$ is given by the following expression:

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \Delta\mathbf{x}^k = \mathbf{x}^k + \alpha^k\mathbf{s}^k = \mathbf{x}^k - \alpha^k\nabla f(\mathbf{x}^k) \tag{6.4}$$

where $\Delta\mathbf{x}^k$ = vector from $\mathbf{x}^k$ to $\mathbf{x}^{k+1}$
$\mathbf{s}^k$ = search direction, the direction of steepest descent
α^k = scalar that determines the step length in direction $\mathbf{s}^k$

The negative of the gradient gives the direction for minimization but not the magnitude of the step to be taken, so that various steepest descent procedures are pos-

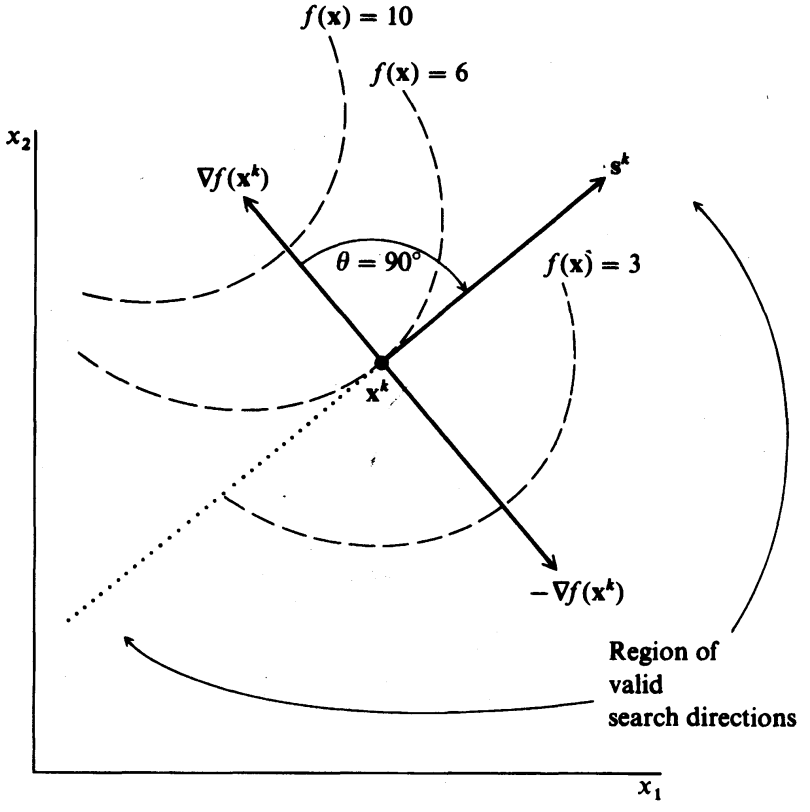


FIGURE 6.5
Identification of the region of possible search directions.

sible, depending on the choice of α^k. We assume that the value of $f(\mathbf{x})$ is continuously reduced. Because one step in the direction of steepest descent will not, in general, arrive at the minimum of $f(\mathbf{x})$, Equation (6.4) must be applied repetitively until the minimum is reached. At the minimum, the value of the elements of the gradient vector will each be equal to zero.

The step size α^k is determined by a line search, using methods like those described in Chapter 5. Although inexact line searches (not continued to the exact minimum) are always used in practice, insight is gained by examining the behavior of steepest descent when an exact line search is used.

First, let us consider the perfectly scaled quadratic objective function $f(\mathbf{x}) = x_1^2 + x_2^2$, whose contours are concentric circles as shown in Figure 6.6. Suppose we calculate the gradient at the point $\mathbf{x}^T = [2 \quad 2]$

$$\nabla f(x) = \begin{bmatrix} 2x_1 \\ 2x_2 \end{bmatrix} \qquad \nabla f(2,2) = \begin{bmatrix} 4 \\ 4 \end{bmatrix} \qquad \mathbf{H}(\mathbf{x}) = \mathbf{H} = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$$

The direction of steepest descent is

$$\mathbf{s} = -\begin{bmatrix} 4 \\ 4 \end{bmatrix}$$

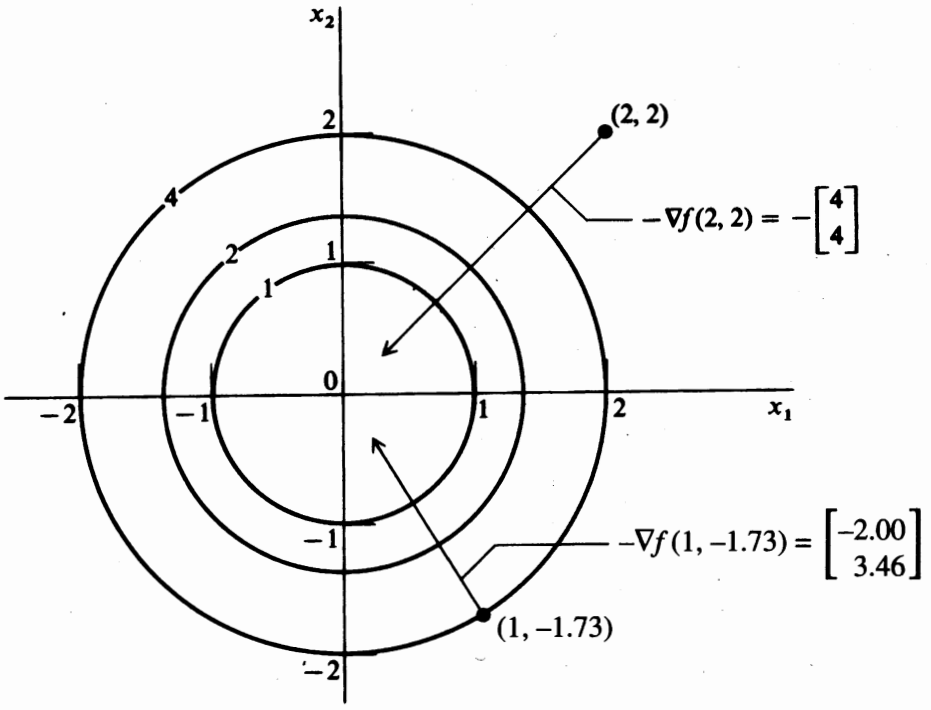


FIGURE 6.6
Gradient vector for $f(\mathbf{x}) = x_1^2 + x_2^2$.

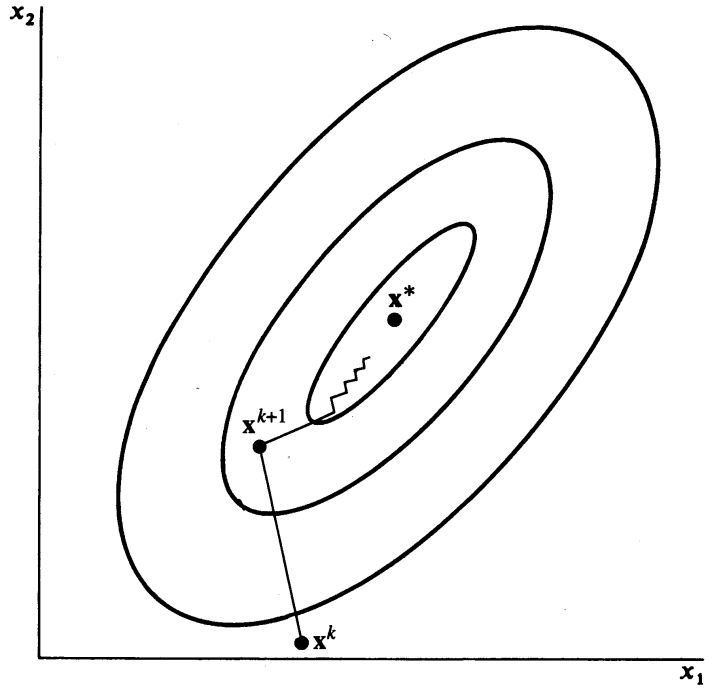


FIGURE 6.7
Steepest descent method for a general quadratic function.

Observe that $\mathbf{s}$ is a vector pointing toward the optimum at (0, 0). In fact, the gradient at any point passes through the origin (the optimum).

On the other hand, for functions not so nicely scaled and that have nonzero off-diagonal terms in the Hessian matrix (corresponding to interaction terms such as x_1x_2), then the negative gradient direction is unlikely to pass directly through the optimum. Figure 6.7 illustrates the contours of a quadratic function of two variables that includes an interaction term. Observe that contours are tilted with respect to the axes. Interaction terms plus poor scaling corresponding to narrow valleys, or ridges, cause the gradient method to exhibit slow convergence.

If α^k is chosen to minimize $f(\mathbf{x}^k + \alpha\mathbf{s}^k)$ exactly then at the minimum,

$$\frac{d}{d\alpha} f(\mathbf{x}^k + \alpha\mathbf{s}^k) = 0$$

We illustrate this in Figure 6.8 using the notation

$$g^k(\alpha) = f(\mathbf{x}^k + \alpha\mathbf{s}^k)$$

where g^k is the function value along the search direction for a given value of α. Because $\mathbf{x}^k$ and $\mathbf{s}^k$ are fixed at known values, g^k depends only on the step size α. If $\mathbf{s}^k$ is a descent direction, then we can always find a positive α that causes f to decrease.

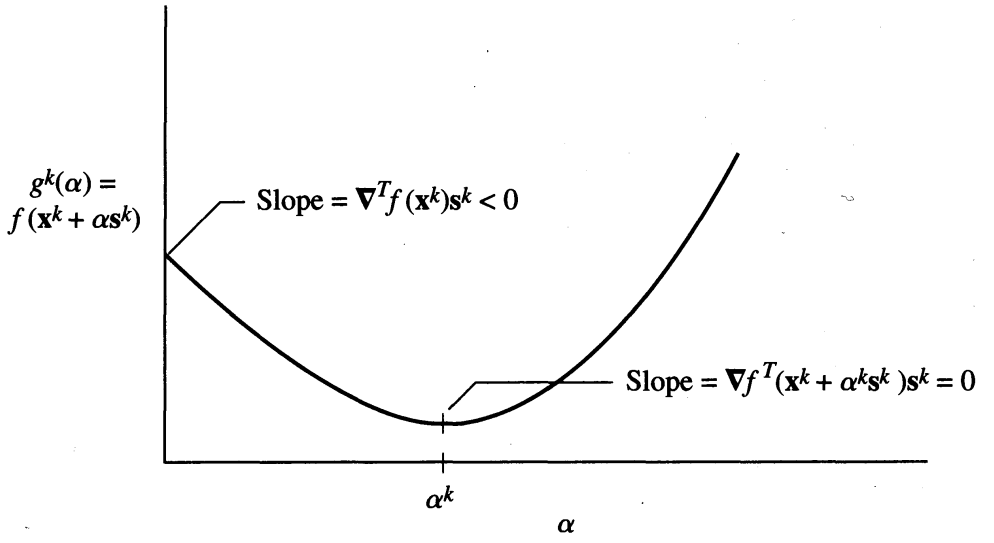


FIGURE 6.8
Exact line search along the search direction $\mathbf{s}^k$.

Using the chain rule

$$\frac{d}{d\alpha} f(\mathbf{x}^k + \alpha \mathbf{s}^k) = \sum_i \frac{\partial f(\mathbf{x}^k + \alpha \mathbf{s}^k)}{\partial \mathbf{x}_i} s_i^k$$

$$= (\mathbf{s}^k)^T \nabla f(\mathbf{x}^k + \alpha \mathbf{s}^k)$$

In an exact line search, we choose α^k as the α that minimizes $g^k(\alpha)$, so

$$\left.\frac{dg^k}{d\alpha}\right|_{\alpha^k} = (\mathbf{s}^k)^T \nabla f(\mathbf{x}^k + \alpha \mathbf{s}^k)\Big|_{\alpha^k} = 0 \tag{6.5}$$

as shown in Figure 6.8. But when the inner product of two vectors is zero, the vectors are orthogonal, so if an exact line search is used, the gradient at the new point $\mathbf{x}^{k+1}$ is orthogonal to the search direction $\mathbf{s}^k$. In steepest descent $\mathbf{s}^k = -\nabla f(\mathbf{x}^k)$, so the gradients at points $\mathbf{x}^k$ and $\mathbf{x}^{k+1}$ are orthogonal. This is illustrated in Figure 6.7, which shows that the orthogonality of successive search directions leads to a very inefficient zigzagging behavior. Although large steps are taken in early iterations, the step sizes shrink rapidly, and converging to an accurate solution of the optimization problem takes many iterations.

The steepest descent algorithm can be summarized in the following steps:

1. Choose an initial or starting point $\mathbf{x}^0$. Thereafter at the point $\mathbf{x}^k$:
2. Calculate (analytically or numerically) the partial derivatives

$$\frac{\partial f(\mathbf{x})}{\partial x_j} \quad j = 1, \ldots, n$$

3. Calculate the search vector

$$\mathbf{s}^k = -\nabla f(\mathbf{x}^k)$$

4. Use the relation

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha^k \mathbf{s}^k$$

to obtain the value of $\mathbf{x}^{k+1}$. To get α^k minimize $g^k(\alpha)$ numerically, as described in Chapter 5.

5. Compare $f(\mathbf{x}^{k+1})$ with $f(\mathbf{x}^k)$: if the change in $f(\mathbf{x})$ is smaller than some tolerance, stop. If not, return to step 2 and set $k = k + 1$. Termination can also be specified by stipulating some tolerance on the norm of $\nabla f(\mathbf{x}^k)$.

Steepest descent can terminate at any type of stationary point, that is, at any point where the elements of the gradient of $f(\mathbf{x})$ are zero. Thus you must ascertain if the presumed minimum is indeed a local minimum (i.e., a solution) or a saddle point. If it is a saddle point, it is necessary to employ a nongradient method to move away from the point, after which the minimization may continue as before. The stationary point may be tested by examining the Hessian matrix of the objective function as described in Chapter 4. If the Hessian matrix is not positive-definite, the stationary point is a saddle point. Perturbation from the stationary point followed by optimization should lead to a local minimum $\mathbf{x}^*$.

The basic difficulty with the steepest descent method is that it is too sensitive to the scaling of $f(\mathbf{x})$, so that convergence is very slow and what amounts to oscillation in the $\mathbf{x}$ space can easily occur. For these reasons steepest descent or ascent is not a very effective optimization technique. Fortunately, conjugate gradient methods are much faster and more accurate.

6.2.2 Conjugate Gradient Methods

The earliest conjugate gradient method was devised by Fletcher and Reeves (1964). If $f(\mathbf{x})$ is quadratic and is minimized exactly in each search direction, it has the desirable features of converging in at most n iterations because its search directions are conjugate. The method represents a major improvement over steepest descent with only a marginal increase in computational effort. It combines current information about the gradient vector with that of gradient vectors from previous iterations (a memory feature) to obtain the new search direction. You compute the search direction by a linear combination of the current gradient and the previous search direction. The main advantage of this method is that it requires only a small amount of information to be stored at each stage of calculation and thus can be applied to very large problems. The steps are listed here.

Step 1. At $\mathbf{x}^0$ calculate $f(\mathbf{x}^0)$. Let

$$\mathbf{s}^0 = -\nabla f(\mathbf{x}^0)$$

Step 2. Save $\nabla f(\mathbf{x}^0)$ and compute

$$\mathbf{x}^1 = \mathbf{x}^0 + \alpha^0 \mathbf{s}^0$$

by minimizing $f(\mathbf{x})$ with respect to α in the $\mathbf{s}^0$ direction (i.e., carry out a unidimensional search for α^0).

Step 3. Calculate $f(\mathbf{x}^1)$, $\nabla f(\mathbf{x}^1)$. The new search direction is a linear combination of $\mathbf{s}^0$ and $\nabla f(\mathbf{x}^1)$:

$$\mathbf{s}^1 = -\nabla f(\mathbf{x}^1) + \mathbf{s}^0 \frac{\nabla^T f(\mathbf{x}^1)\, \nabla f(\mathbf{x}^1)}{\nabla^T f(\mathbf{x}^0)\, \nabla f(\mathbf{x}^0)}$$

For the kth iteration the relation is

$$\mathbf{s}^{k+1} = -\nabla f(\mathbf{x}^{k+1}) + \mathbf{s}^k \frac{\nabla^T f(\mathbf{x}^{k+1})\, \nabla f(\mathbf{x}^{k+1})}{\nabla^T f(\mathbf{x}^k)\, \nabla f(\mathbf{x}^k)} \tag{6.6}$$

For a quadratic function it can be shown that these successive search directions are conjugate. After n iterations ($k = n$), the quadratic function is minimized. For a nonquadratic function, the procedure cycles again with $\mathbf{x}^{n+1}$ becoming $\mathbf{x}^0$.

Step 4. Test for convergence to the minimum of $f(\mathbf{x})$. If convergence is not attained, return to step 3.

Step *n*. Terminate the algorithm when $\| \nabla f(\mathbf{x}^k) \|$ is less than some prescribed tolerance.

Note that if the ratio of the inner products of the gradients from stage $k + 1$ relative to stage k is very small, the conjugate gradient method behaves much like the steepest descent method. One difficulty is the linear dependence of search directions, which can be resolved by periodically restarting the conjugate gradient method with a steepest descent search (step 1). The proof that Equation (6.6) yields conjugate directions and quadratic convergence was given by Fletcher and Reeves (1964).

In doing the line search we can minimize a quadratic approximation in a given search direction. This means that to compute the value for α for the relation $\mathbf{x}^{k-1} = \mathbf{x}^k + \alpha \mathbf{s}^k$ we must minimize

$$f(\mathbf{x}) = f(\mathbf{x}^k + \alpha \mathbf{s}^k) = f(\mathbf{x}^k) + \nabla^T f(\mathbf{x}^k)\, \alpha \mathbf{s}^k + \tfrac{1}{2} (\alpha \mathbf{s}^k)^T \mathbf{H}(\mathbf{x}^k)\, (\alpha \mathbf{s}^k) \tag{6.7}$$

where $\Delta \mathbf{x}^k = \alpha \mathbf{s}^k$. To get the minimum of $f(\mathbf{x}^k + \alpha \mathbf{s}^k)$, we differentiate Equation (6.3) with respect to α and equate the derivative to zero

$$\frac{df(\mathbf{x}^k + \alpha \mathbf{s}^k)}{d\alpha} = 0 = \nabla^T f(\mathbf{x}^k) \mathbf{s}^k + (\mathbf{s}^k)^T \mathbf{H}(\mathbf{x}^k)\, \alpha \mathbf{s}^k \tag{6.8}$$

with the result

$$\alpha^{\text{opt}} = -\frac{\nabla^T f(\mathbf{x}^k)\, \mathbf{s}^k}{(\mathbf{s}^k)^T \mathbf{H}(\mathbf{x}^k)\, \mathbf{s}^k} \tag{6.9}$$

For additional details concerning the application of conjugate gradient methods, especially to large-scale and sparse problems, refer to Fletcher (1980), Gill et al. (1981), Dembo et al. (1982), and Nash and Sofer (1996).

EXAMPLE 6.2 APPLICATION OF THE FLETCHER–REEVES CONJUGATE GRADIENT ALGORITHM

We solve the problem known as Rosenbrock's function

$$\text{Minimize: } f(\mathbf{x}) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$$

starting at $\mathbf{x}^{(0)} = [-1.2 \;\; 1.0]^T$. The first few stages of the Fletcher–Reeves procedure are listed in Table E6.2. The trajectory as it moves toward the optimum is shown in Figure E6.2.

TABLE E6.2
Results for Example 6.2 using the Fletcher–Reeves method

Iteration	Number of function calls	$f(\mathbf{x})$	x_1	x_2	$\frac{\partial f(\mathbf{x})}{\partial x_1}$	$\frac{\partial f(\mathbf{x})}{\partial x_2}$
0	1	24.2	−1.2	1.0	−215.6	−88.00
1	4	4.377945	−1.050203	1.061141	−21.65	−8.357
5	14	3.165142	−0.777190	0.612232	−1.002	−1.6415
10	28	1.247687	−0.079213	−0.025322	−3.071	−5.761
15	41	0.556612	0.254058	0.063189	−1.354	−0.271
20	57	0.147607	0.647165	0.403619	3.230	−3.040
25	69	0.024667	0.843083	0.710119	−0.0881	−0.1339
30	80	0.0000628	0.995000	0.989410	0.2348	−0.1230
35	90	1.617×10^{-15}	1.000000	1.000000	-1.60×10^{-8}	-3.12×10^{-8}

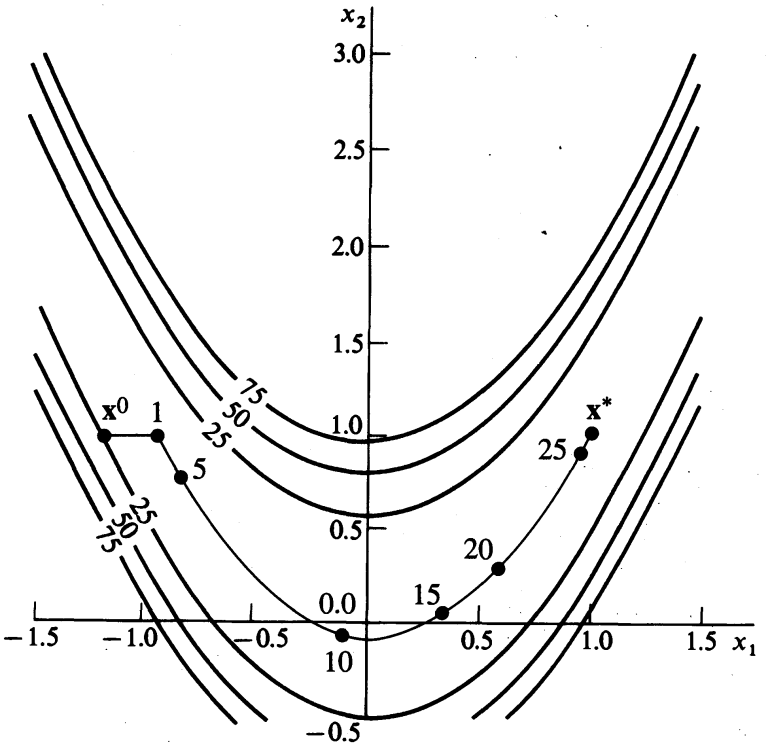


FIGURE E6.2
Search trajectory for the Fletcher–Reeves algorithm (the numbers designate the iteration).

6.3 NEWTON'S METHOD

From one viewpoint the search direction of steepest descent can be interpreted as being orthogonal to a linear approximation (tangent to) of the objective function at point $\mathbf{x}^k$; examine Figure 6.9a. Now suppose we make a quadratic approximation of $f(\mathbf{x})$ at $\mathbf{x}^k$

$$f(\mathbf{x}) \approx f(\mathbf{x}^k) + \nabla^T f(\mathbf{x}^k)\,\Delta\mathbf{x}^k + \tfrac{1}{2}(\Delta\mathbf{x}^k)^T\mathbf{H}(\mathbf{x}^k)\,\Delta\mathbf{x}^k \tag{6.10}$$

where $\mathbf{H}(\mathbf{x}^k)$ is the Hessian matrix of $f(\mathbf{x})$ defined in Chapter 4 (the matrix of second partial derivatives with respect to $\mathbf{x}$ evaluated at $\mathbf{x}^k$). Then it is possible to take into account the curvature of $f(\mathbf{x})$ at $\mathbf{x}^k$ in determining a search direction as described later on.

Newton's method makes use of the second-order (quadratic) approximation of $f(\mathbf{x})$ at $\mathbf{x}^k$ and thus employs second-order information about $f(\mathbf{x})$, that is, information obtained from the second partial derivatives of $f(\mathbf{x})$ with respect to the independent variables. Thus, it is possible to take into account the curvature of $f(\mathbf{x})$ at $\mathbf{x}^k$ and identify better search directions than can be obtained via the gradient method. Examine Figure 6.9b.

The minimum of the quadratic approximation of $f(\mathbf{x})$ in Equation (6.10) is obtained by differentiating (6.10) with respect to each of the components of $\Delta\mathbf{x}$ and equating the resulting expressions to zero to give

$$\nabla f(\mathbf{x}) = \nabla f(\mathbf{x}^k) + \mathbf{H}(\mathbf{x}^k)\,\Delta\mathbf{x}^k = 0 \tag{6.11}$$

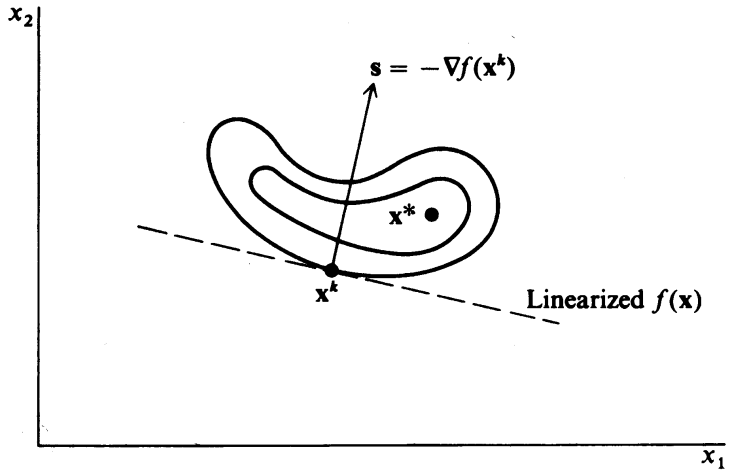
or

$$\mathbf{x}^{k+1} - \mathbf{x}^k = \Delta\mathbf{x}^k = -[\mathbf{H}(\mathbf{x}^k)]^{-1}\nabla f(\mathbf{x}^k) \tag{6.12}$$

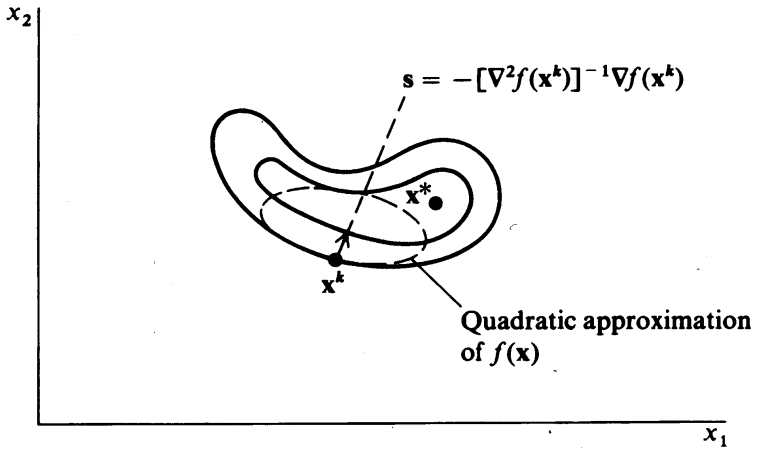
where $[\mathbf{H}(\mathbf{x}^k)]^{-1}$ is the inverse of the Hessian matrix $\mathbf{H}(\mathbf{x}^k)$. Equation (6.12) reduces to Equation (5.5) for a one-dimensional search.

Note that both the direction *and* step length are specified as a result of Equation (6.11). If $f(\mathbf{x})$ is actually quadratic, only one step is required to reach the minimum of $f(\mathbf{x})$. For a general nonlinear objective function, however, the minimum of $f(\mathbf{x})$ cannot be reached in one step, so that Equation (6.12) can be modified to conform to Equation (6.7) by introducing the parameter for the step length into (6.12).

$$\mathbf{x}^{k+1} - \mathbf{x}^k = -\alpha^k[\mathbf{H}(\mathbf{x}^k)]^{-1}\nabla f(\mathbf{x}^k) \tag{6.13}$$



(a) Steepest descent: first-order approximation (linearization) of $f(\mathbf{x})$ at $\mathbf{x}^k$



(b) Newton's method: second-order (quadratic) approximation of $f(\mathbf{x})$ at $\mathbf{x}^k$

FIGURE 6.9
Comparison of steepest descent with Newton's method from the viewpoint of objective function approximation.

Observe that the search direction $\mathbf{s}$ is now given (for minimization) by

$$\mathbf{s}^k = -[\mathbf{H}(\mathbf{x}^k)]^{-1}\nabla f(\mathbf{x}^k) \tag{6.14}$$

and that the step length is α^k. The step length α^k can be evaluated numerically as described in Chapter 5. Equation (6.13) is applied iteratively until some termination criteria are satisfied. For the "pure" version of Newton's method, $\alpha = 1$ on each step. However, this version often does not converge if the initial point is not close enough to a local minimum.

Also note that to evaluate $\Delta \mathbf{x}$ in Equation (6.12), a matrix inversion is not necessarily required. You can take its precursor, Equation (6.11), and solve the following set of linear equations for $\Delta \mathbf{x}^k$

$$\mathbf{H}(\mathbf{x}^k)\,\Delta \mathbf{x}^k = -\nabla f(\mathbf{x}^k) \tag{6.15}$$

a procedure that often leads to less round-off error than calculating **s** via the inversion of a matrix.

EXAMPLE 6.3 APPLICATION OF NEWTON'S METHOD TO A CONVEX QUADRATIC FUNCTION

We minimize the function

$$f(\mathbf{x}) = 4x_1^2 + x_2^2 - 2x_1x_2$$

starting at $\mathbf{x}^0 = [1 \quad 1]^T$

$$\nabla f(\mathbf{x}) = \begin{bmatrix} 8x_1 & -2x_2 \\ 2x_2 & -2x_1 \end{bmatrix}$$

$$\mathbf{H}(\mathbf{x}) = \begin{bmatrix} 8 & -2 \\ -2 & 2 \end{bmatrix} \qquad \mathbf{H}^{-1}(\mathbf{x}) = \begin{bmatrix} \frac{1}{6} & \frac{1}{6} \\ \frac{1}{6} & \frac{2}{3} \end{bmatrix}$$

with $\alpha = 1$,

$$\Delta \mathbf{x}^0 = -\mathbf{H}^{-1}\,\nabla f(\mathbf{x}^0) = -\begin{bmatrix} \frac{1}{6} & \frac{1}{6} \\ \frac{1}{6} & \frac{2}{3} \end{bmatrix}\begin{bmatrix} 6 \\ 0 \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \end{bmatrix}$$

hence,

$$\mathbf{x}^1 = \mathbf{x}^* = \mathbf{x}^0 + \Delta \mathbf{x}^0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \begin{bmatrix} -1 \\ -1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

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

Instead of taking the inverse of **H**, we can solve Equation (6.15)

$$\begin{bmatrix} 8 & -2 \\ -2 & 2 \end{bmatrix}\begin{bmatrix} \Delta x_1^0 \\ \Delta x_2^2 \end{bmatrix} = -\begin{bmatrix} 6 \\ 0 \end{bmatrix}$$

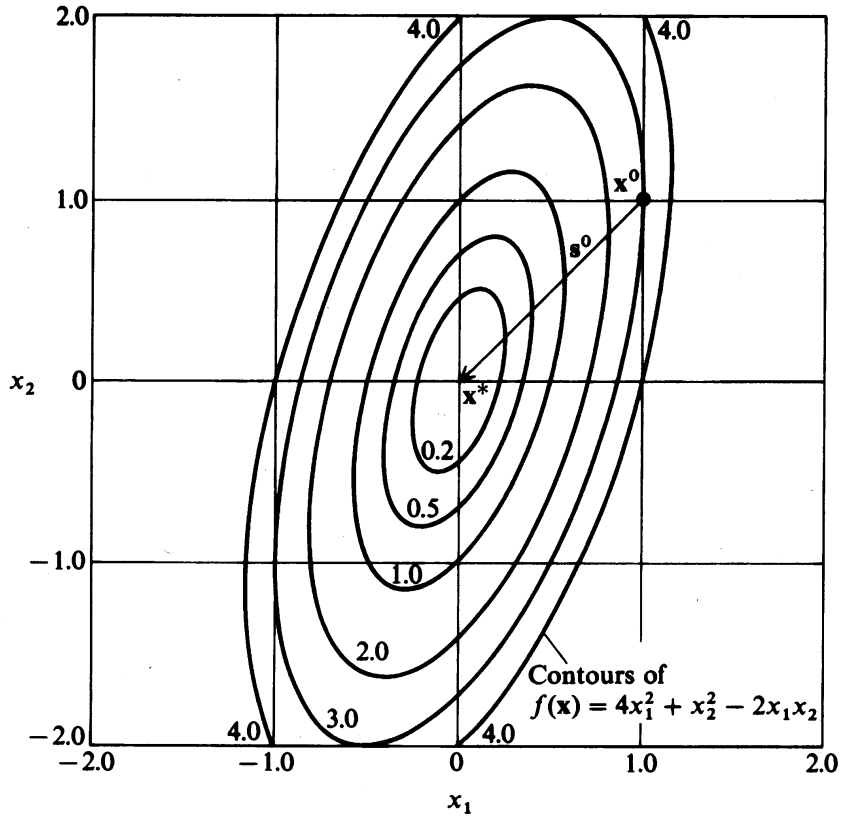


FIGURE E6.3

which gives

$$\Delta x_1^0 = -1$$

$$\Delta x_2^0 = -1$$

as before. The search direction $\mathbf{s}^0 = -\mathbf{H}^{-1}\nabla f(\mathbf{x}^0)$ is shown in Figure E6.3

EXAMPLE 6.4 APPLICATION OF NEWTON'S METHOD AND QUADRATIC CONVERGENCE

If we minimize the nonquadratic function

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

from the starting point of (1, 1), can you show that Newton's method exhibits quadratic convergence? *Hint*: Show that

$$\frac{\|\mathbf{x}^{k+1} - \mathbf{x}^*\|}{\|\mathbf{x}^k - \mathbf{x}^*\|^2} < c \quad \text{(see Section 5.3)}$$

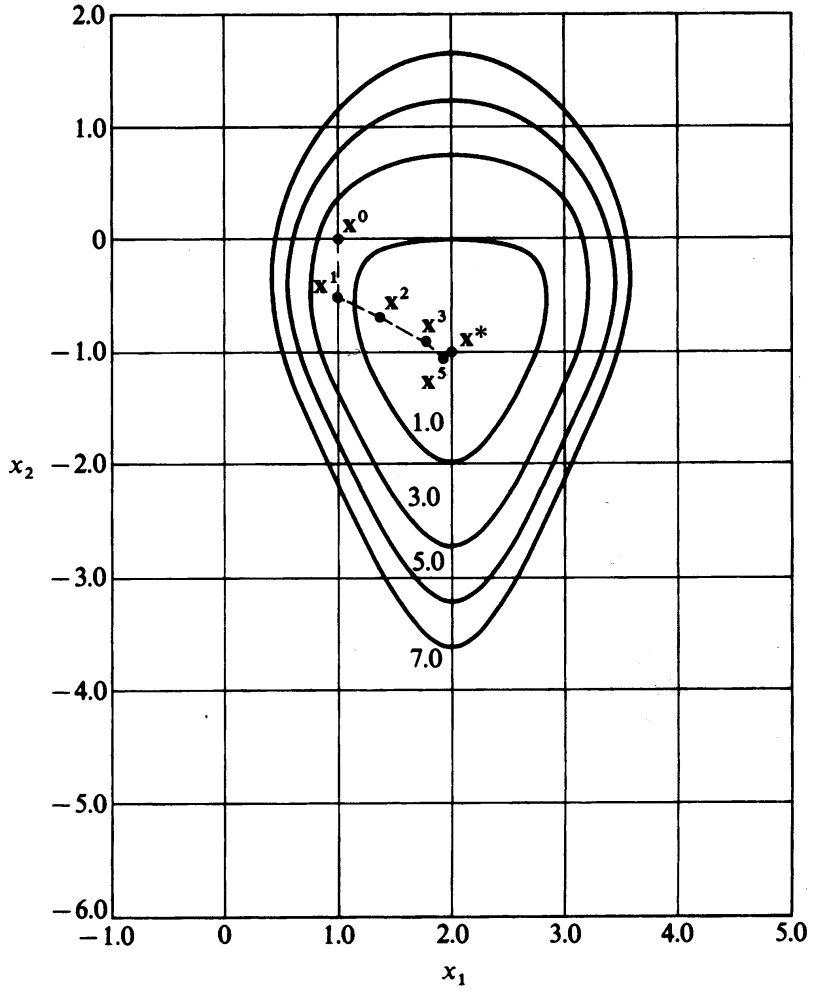


FIGURE E6.4

Solution. Newton's method produces the following sequences of values for x_1, x_2, and $[f(\mathbf{x}^{k+1}) - f(\mathbf{x}^k)]$ (you should try to verify the calculations shown in the following table; the trajectory is traced in Figure E6.4).

Iteration	x_1	x_2	$f(\mathbf{x}^{k+1}) - f(\mathbf{x}^k)$
0	1.000000	1.000000	6.000
1	1.000000	–0.500000	1.500
2	1.391304	–0.695652	4.09×10^{-1}
3	1.745944	–0.948798	6.49×10^{-2}
4	1.986278	–1.048208	2.53×10^{-3}
5	1.998734	–1.000170	1.63×10^{-6}
6	1.9999996	–1.000002	2.75×10^{-12}

You can calculate between iterations 2 and 3 that $c = 0.55$; and between 3 and 4 that $c \approx 0.74$. Hence, quadratic convergence can be demonstrated numerically.

Newton's method usually requires the fewest iterations of all the methods discussed in this chapter, but it has the following disadvantages:

1. The method does not necessarily find the global solution if multiple local solutions exist, but this is a characteristic of all the methods described in this chapter.
2. It requires the solution of a set of n symmetric linear equations.
3. It requires both first and second partial derivatives, which may not be practical to obtain.
4. Using a step size of unity, the method may not converge.

Difficulty 3 can be ameliorated by using (properly) finite difference approximation as substitutes for derivatives. To overcome difficulty 4, two classes of methods exist to modify the "pure" Newton's method so that it is guaranteed to converge to a local minimum from an arbitrary starting point. The first of these, called *trust region methods*, minimize the quadratic approximation, Equation (6.10), within an elliptical region, whose size is adjusted so that the objective improves at each iteration; see Section 6.3.2. The second class, line search methods, modifies the pure Newton's method in two ways: (1) instead of taking a step size of one, a line search is used and (2) if the Hessian matrix $\mathbf{H}(\mathbf{x}^k)$ is not positive-definite, it is replaced by a positive-definite matrix that is "close" to $\mathbf{H}(\mathbf{x}^k)$. This is motivated by the easily verified fact that, if $\mathbf{H}(\mathbf{x}^k)$ is positive-definite, the Newton direction

$$\mathbf{s}^k = -[\mathbf{H}(\mathbf{x}^k)]^{-1} \nabla f(\mathbf{x}^k)$$

is a descent direction, that is

$$\nabla f^T(\mathbf{x}^k)\mathbf{s}^k < 0$$

If $f(\mathbf{x})$ is convex, $\mathbf{H}(\mathbf{x})$ is positive-semidefinite at all points $\mathbf{x}$ and is usually positive-definite. Hence Newton's method, using a line search, converges. If $f(\mathbf{x})$ is not strictly convex (as is often the case in regions far from the optimum), $\mathbf{H}(\mathbf{x})$ may not be positive-definite everywhere, so one approach to forcing convergence is to replace $\mathbf{H}(\mathbf{x})$ by another positive-definite matrix. The Marquardt–Levenberg method is one way of doing this, as discussed in the next section.

6.3.1 Forcing the Hessian Matrix to Be Positive-Definite

Marquardt (1963), Levenberg (1944), and others have suggested that the Hessian matrix of $f(\mathbf{x})$ be modified on each stage of the search as needed to ensure that the modified $\mathbf{H}(\mathbf{x}), \tilde{\mathbf{H}}(\mathbf{x})$, is positive-definite and well conditioned. The procedure adds elements to the diagonal elements of $\mathbf{H}(\mathbf{x})$

$$\tilde{\mathbf{H}}(\mathbf{x}) = [\mathbf{H}(\mathbf{x}) + \beta \mathbf{I}] \tag{6.16}$$

where β is a positive constant large enough to make $\tilde{\mathbf{H}}(\mathbf{x})$ positive-definite when $\mathbf{H}(\mathbf{x})$ is not. Note that with a β sufficiently large, $\beta\mathbf{I}$ can overwhelm $\mathbf{H}(\mathbf{x})$ and the minimization approaches a steepest descent search.

TABLE 6.1
A modified Marquardt method

Step 1
Pick $\mathbf{x}^0$ the starting point. Let ε = convergence criterion.
Step 2
Set $k = 0$. Let $\beta^0 = 10^3$.
Step 3
Calculate $\nabla f(\mathbf{x}^k)$.
Step 4
Is $\| \nabla f(\mathbf{x}^k)) < \varepsilon$? If yes, terminate. If no, continue.
Step 5
Solve $(\mathbf{H}(\mathbf{x}^k) + \beta^k \mathbf{I})\mathbf{s}^k = -\nabla f(\mathbf{x}^k)$ for $\mathbf{s}^k$.
Step 6
If $\nabla f^T(\mathbf{x}^k)\mathbf{s}^k < 0$, go to step 8.
Step 7
Set $\beta^k = 2\beta^k$ and go to step 5.
Step 8
Choose α^k by a line search procedure so that

$$f(\mathbf{x}^k + \alpha^k \mathbf{s}^k) < f(\mathbf{x}^k)$$

Step 9
If certain conditions are met (Dennis and Schnabel, 1996), reduce β.
Go to step 3 with k replaced by $k + 1$.

A simpler procedure that may result in a suitable value of β is to apply a modified Cholesky factorization as follows:

$$\mathbf{H}(\mathbf{x}^k) + \mathbf{D} = \mathbf{L}\mathbf{L}^T \tag{6.17}$$

where $\mathbf{D}$ is a diagonal matrix with nonnegative elements [$d_{ii} = 0$ if $\mathbf{H}(\mathbf{x}^k)$ is positive-definite] and $\mathbf{L}$ is a lower triangular matrix. Upper bounds on the elements in $\mathbf{D}$ are calculated using the Gershgorin circle theorem [see Dennis and Schnabel (1996) for details].

A simple algorithm based on an arbitrary adjustment of β (a modified Marquardt's method) is listed in Table 6.1.

EXAMPLE 6.5 APPLICATION OF MARQUARDT'S METHOD

The algorithm listed in Table 6.1 is to be applied to Rosenbrock's function $f(\mathbf{x}) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$ starting at $\mathbf{x}^0 = [-1.2 \quad 1.0]^T$ with $\mathbf{H}^0 = \mathbf{H}(\mathbf{x}^0)$.

TABLE E6.5
Marquardt's method

					Elements of $[\mathbf{H}(\mathbf{x}^k) + \beta\mathbf{I}]^{-1}$			
$f(\mathbf{x})$	x_1	x_2	$\frac{\partial f(\mathbf{x})}{\partial x_1}$	$\frac{\partial f(\mathbf{x})}{\partial x_2}$	$\tilde{h}_{11}^{-1}$	$\tilde{h}_{12}^{-1}$	$\tilde{h}_{21}^{-1}$	$\tilde{h}_{22}^{-1}$
24.2000	−1.2000	1.0000	−215.6000	−88.0000	0.0005	−0.0002	−0.0002	0.0009
4.1498	−1.0315	1.0791	2.1844	3.0284	0.0005	−0.0002	−0.0002	0.0009
4.1173	−1.0289	1.0557	−5.2448	−0.5768	0.0014	−0.0013	−0.0013	0.0034
3.9642	−0.9412	0.9301	12.7861	8.8552	0.0037	−0.0059	−0.0059	0.0130
3.4776	−0.8542	0.7098	−10.5031	−3.9772	0.0195	−0.0341	−0.0341	0.0641
2.7527	−0.6028	0.3206	−13.5391	−8.5706	0.0399	−0.0669	−0.0669	0.1170
1.9132	−0.3167	0.0580	−7.9993	−8.4706	0.0464	−0.0557	−0.0557	0.0718
1.1890	−0.0313	−0.0344	−2.5059	−7.0832	0.0519	−0.0328	−0.0328	0.0258
0.6885	0.2278	0.0215	1.2242	−6.0759	0.0616	−0.0039	−0.0039	0.0052
0.3266	0.4570	0.2031	3.2402	−4.5160	0.0706	0.0322	0.0322	0.0196
0.1275	0.6846	0.4520	3.9595	3.3523	0.0906	0.0861	0.0861	0.0868
0.0237	0.8705	0.7495	2.6299	−1.6593	0.1148	0.1573	0.1573	0.2203
0.0006	0.9870	0.9721	0.7700	−0.4033	0.1880	0.3273	0.3273	0.5748
0.0000	0.9974	0.9949	−0.0589	0.0269	0.3563	0.7033	0.7033	1.3932
0.0000	0.9999	0.9999	−0.0004	0.0002	0.5138	1.0249	1.0249	2.0494
0.0000	1.0000	1.0000	0.0000	−0.0000	0.5001	1.0001	1.0001	2.0050
0.0000	1.0000	1.0000						

A quadratic interpolation subroutine was used to minimize in each search direction. Table E6.5 lists the values of $f(\mathbf{x})$, $\mathbf{x}$, $\nabla f(\mathbf{x})$, and the elements of $[\mathbf{H}(\mathbf{x}) + \beta\mathbf{I}]^{-1}$ for each stage of the minimization. A total of 96 function evaluations and 16 calls to the gradient evaluation subroutine were needed.

6.3.2 Movement in the Search Direction

Up to this point we focused on calculating $\mathbf{H}$ or $\mathbf{H}^{-1}$, from which the search direction $\mathbf{s}$ can be ascertained via Equation (6.14) or $\Delta\mathbf{x}$ from Equation (6.15) (for minimization). In this section we discuss briefly how far to proceed in the search direction, that is, select a step length, for a general function $f(\mathbf{x})$. If $\Delta\mathbf{x}$ is calculated from Equations (6.12) or (6.15), $\alpha = 1$ and the step is a Newton step. If $\alpha \neq 1$, then any procedure can be used to calculate α as discussed in Chapter 5.

Line search. The oldest and simplest method of calculating α to obtain $\Delta\mathbf{x}$ is via a *unidimensional line search.* In a given direction that reduces $f(\mathbf{x})$, take a step, or a sequence of steps yielding an overall step, that reduces $f(\mathbf{x})$ to some acceptable degree. This operation can be carried out by any of the one-dimensional search

techniques described in Chapter 5. Early investigators always minimized $f(\mathbf{x})$ as accurately as possible in a search direction $\mathbf{s}$, but subsequent experience, and to some extent theoretical results, have indicated that such a concept is invalid. Good algorithms first calculate a full Newton step ($\alpha = 1$) to get $\mathbf{x}^{k+1}$, and if $f(\mathbf{x}^k)$ is not reduced, backtrack in some systematic way toward $\mathbf{x}^k$. Failure to take the full Newton step in the first iteration leads to loss of the advantages of Newton's method near the minimum, where convergence is slow. To avoid very small decreases in $f(\mathbf{x})$, most algorithms require that the average rate of descent from $\mathbf{x}^k$ to $\mathbf{x}^{k+1}$ be at least some prescribed fraction of the initial rate of descent in the search direction. Mathematically this means (Armijo, 1966)

$$f(\mathbf{x}^k + \alpha^1 \mathbf{s}^k) \leq f(\mathbf{x}^k) + \gamma\alpha \nabla^T f(\mathbf{x}^k)\mathbf{s}^k \tag{6.18}$$

Examine Figure 6.10. In practice γ is often chosen to be very small, about 10^{-4}, so just a small decrease in the function value is required.

Backtracking can be accomplished in any of the ways outlined in Chapter 5 but with the objective of locating an $\mathbf{x}^{k+1}$ for which $f(\mathbf{x}^{k+1}) < f(\mathbf{x}^k)$ but moving as far as possible in the direction $\mathbf{s}^k$ from $\mathbf{x}^k$. The minimum of $f(\mathbf{x}^k + \alpha\mathbf{s}^k)$ does not have to be found exactly. As an example of one procedure, at $\mathbf{x}^k$, where $\alpha = 0$, you know two pieces of information about $f(\mathbf{x}^k + \alpha\mathbf{s}^k)$: the values of $f(\mathbf{x}^k)$ and $\nabla^T f(\mathbf{x}^k)\,\mathbf{s}^k$. After the Newton step ($\alpha = 1$) you know the value of $f(\mathbf{x}^k + \mathbf{s}^k)$. From these three pieces of information you can make a quadratic interpolation to get the value α where the objective function $f(\alpha)$ has a minimum:

$$\hat{\alpha} = -\frac{\nabla^T f(\mathbf{x}^k)\mathbf{s}^k}{2[f(\mathbf{x}^k + \mathbf{s}^k) - f(\mathbf{x}^k) - \nabla^T f(\mathbf{x}^k)\mathbf{s}^k]} \tag{6.19}$$

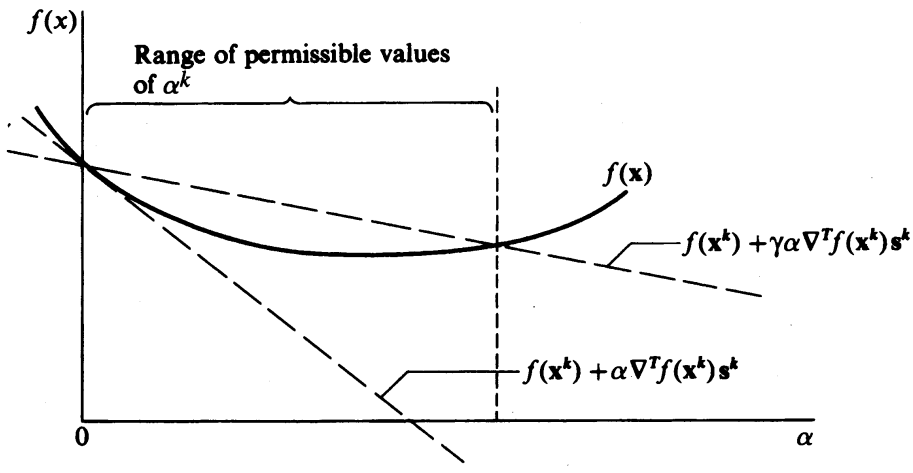


FIGURE 6.10
Range of acceptable values for choice of α^k to meet criterion (6.20) with $\gamma = 0.02$.

After $\hat{\alpha}$ is obtained, if additional backtracking is needed, cubic interpolation can be carried out. We suggest that if $\hat{\alpha}$ is too small, say $\hat{\alpha} < 0.1$, try $\hat{\alpha} = 0.1$ instead.

Trust regions. The name *trust region* refers to the region in which the quadratic model can be "trusted" to represent $f(\mathbf{x})$ reasonably well. In the unidimensional line search, the search direction is retained but the step length is reduced if the Newton step proves to be unsatisfactory. In the trust region approach, a shorter step length is selected and *then* the search direction determined. Refer to Dennis and Schnabel (1996) and Section 8.5.1 for details.

The trust region approach estimates the length of a maximal successful step from $\mathbf{x}^k$. In other words, $\|\mathbf{x}\| < \rho$, the bound on the step. Figure 6.11 shows $f(\mathbf{x})$, the quadratic model of $f(\mathbf{x})$, and the desired trust region. First, an initial estimate of ρ or the step bound has to be determined. If knowledge about the problem does

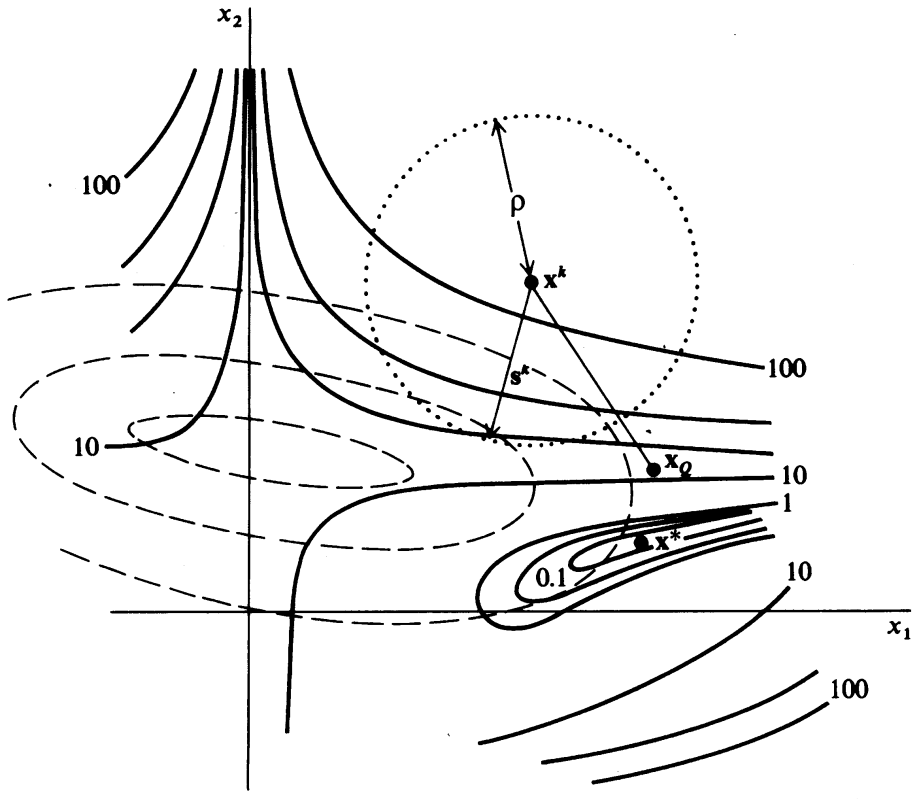


FIGURE 6.11
Representation of the trust region to select the step length. Solid lines are contours of $f(\mathbf{x})$. Dashed lines are contours of the convex quadratic approximation of $f(\mathbf{x})$ at $\mathbf{x}^k$. The dotted circle is the trust region boundary in which δ is the step length. $\mathbf{x}_0$ is the minimum of the quadratic model for which $\hat{\mathbf{H}}(\mathbf{x})$ is positive-definite.

not help, Powell (1970) suggested using the distance to the minimizer of the quadratic model of $f(\mathbf{x})$ in the direction of steepest descent from $\mathbf{x}^k$, the so-called Cauchy point. Next, some curve or piecewise linear function is determined with an initial direction of steepest descent so that the tentative point $\mathbf{x}^{k+1}$ lies on the curve and is less than ρ. Figure 6.11 shows $\mathbf{s}$ as a straight line of one segment. The trust region is updated, and the sequence is continued. Heuristic parameters are usually required, such as minimum and maximum step lengths, scaling $\mathbf{s}$, and so forth.

6.3.3 Termination

No single stopping criterion will suffice for Newton's method or any of the optimization methods described in this chapter. The following simultaneous criteria are recommended to avoid scaling problems:

$$|f(\mathbf{x}^{k+1}) - f(\mathbf{x}^k)| < \varepsilon_1(1 + |f(\mathbf{x}^k)|) \tag{6.20}$$

where the "one" on the right-hand side is present to ensure that the right-hand side is not too small when $f(\mathbf{x}^k)$ approaches zero. Also

$$\|\mathbf{x}^{k+1} - \mathbf{x}^k\| < \varepsilon_2(1 + \|\mathbf{x}^k\|) \tag{6.21}$$

and

$$\|\nabla f(\mathbf{x}^k)\| < \varepsilon_3 \tag{6.22}$$

6.3.4 Safeguarded Newton's Method

Several numerical subroutine libraries contain "safeguarded" Newton codes using the ideas previously discussed. When first and second derivatives can be computed quickly and accurately, a good safeguarded Newton code is fast, reliable, and locates a local optimum very accurately. We discuss this NLP software in Section 8.9.

6.3.5 Computation of Derivatives

From numerous tests involving optimization of nonlinear functions, methods that use derivatives have been demonstrated to be more efficient than those that do not. By replacing analytical derivatives with their finite difference substitutes, you can avoid having to code formulas for derivatives. Procedures that use second-order information are more accurate and require fewer iterations than those that use only first-order information(gradients), but keep in mind that usually the second-order information may be only approximate as it is based not on second derivatives themselves but their finite difference approximations.

6.4 QUASI-NEWTON METHODS

Procedures that compute a search direction using only first derivatives of f provide an attractive alternative to Newton's method. The most popular of these are the quasi-Newton methods that replace $\mathbf{H}(\mathbf{x}^k)$ in Equation (6.11) by a positive-definite approximation $\tilde{\mathbf{H}}^k$:

$$\tilde{\mathbf{H}}^k \mathbf{s}^k = -\nabla f(\mathbf{x}^k) \tag{6.23}$$

$\tilde{\mathbf{H}}^k$ is initialized as any positive-definite symmetric matrix (often the identity matrix or a diagonal matrix) and is updated after each line search using the changes in $\mathbf{x}$ and in $\nabla f(\mathbf{x})$ over the last two points, as measured by the vectors

$$\mathbf{d}^k = \mathbf{x}^{k+1} - \mathbf{x}^k \tag{6.24}$$

and

$$\mathbf{y}^k = \nabla f(\mathbf{x}^{k+1}) - \nabla f(\mathbf{x}^k) \tag{6.25}$$

One of the most efficient and widely used updating formula is the BFGS update. Broyden (1970), Fletcher (1970), Goldfarb (1970), and Shanno (1970) independently published this algorithm in the same year, hence the combined name BFGS. Here the approximate Hessian is given by

$$\tilde{\mathbf{H}}^{k+1} = \tilde{\mathbf{H}}^k + \frac{\mathbf{y}^k (\mathbf{y}^k)^T}{(\mathbf{d}^k)^T \mathbf{y}^k} - \frac{(\tilde{\mathbf{H}}^k \mathbf{d}^k)(\tilde{\mathbf{H}}^k \mathbf{d}^k)^T}{(\mathbf{d}^k)^T \tilde{\mathbf{H}}^k \mathbf{d}^k} \tag{6.26}$$

If $\tilde{\mathbf{H}}^k$ is positive-definite and $(\mathbf{d}^k)^T \mathbf{y}^k > 0$, it can be shown that $\tilde{\mathbf{H}}^{k+1}$ is positive-definite (Dennis and Schnabel, 1996, Chapter 9). The condition $(\mathbf{d}^k)^T \mathbf{y}^k > 0$ can be interpreted geometrically, since

$$\begin{aligned}
(\mathbf{d}^k)^T \mathbf{y}^k &= \alpha^k (\mathbf{s}^k)^T [\nabla f(\mathbf{x}^{k+1}) - \nabla f(\mathbf{x}^k)] \\
&= \alpha^k [(\mathbf{s}^k)^T \nabla f(\mathbf{x}^{k+1}) - (\mathbf{s}^k)^T \nabla f(\mathbf{x}^k)] \\
&= \alpha^k (\text{slope2} - \text{slope1})
\end{aligned}$$

The quantity slope2 is the slope of the line search objective function $g^k(\alpha)$ at $\alpha = \alpha^k$ (see Figure 6.8) and slope1 is its slope at $\alpha = 0$, so $(\mathbf{d}^k)^T \mathbf{y}^k > 0$ if and only if slope2 > slope1. This condition is always satisfied if f is strictly convex. A good line search routine attempts to meet this condition; if it is not met, then $\tilde{\mathbf{H}}^k$ is not updated.

If the BFGS algorithm is applied to a positive-definite quadratic function of n variables and the line search is exact, it will minimize the function in at most n iterations (Dennis and Schnabel, 1996, Chapter 9). This is also true for some other updating formulas. For nonquadratic functions, a good BFGS code usually requires more iterations than a comparable Newton implementation and may not be as accurate. Each BFGS iteration is generally faster, however, because second derivatives are not required and the system of linear equations (6.15) need not be solved.

EXAMPLE 6.6 APPLICATION OF THE BFGS METHOD

Apply the BFGS method to find the minimum of the function $f(\mathbf{x}) = x_1^4 - 2x_2x_1^2 + x_2^2 + x_1^2 - 2x_1 + 5$.

Use a starting point of (1,2) and terminate the search when f changes less than 0.00005 between iterations. The contour plot for the function was shown in Figure 5.7.

Solution. Using the Optimization Toolbox from MATLAB, the BFGS method requires 20 iterations before the search is terminated, as shown below.

TABLE E6.6
BFGS method

Iteration	x_1	x_2	$f(x_1, x_2)$	$\frac{\partial f}{\partial x_1}$	$\frac{\partial f}{\partial x_2}$
	1.00000	2.00000	5.00000	−4.00000	2.00000
1	1.29611	1.82473	4.10866	−0.15866	0.28966
2	1.29192	1.73556	4.08964	0.24022	0.13299
3	1.22980	1.63069	4.06680	−0.12218	0.23654
4	1.22409	1.54972	4.05285	0.19694	0.10263
5	1.17160	1.46528	4.03803	−0.09085	0.18524
6	1.16530	1.39587	4.02876	0.15372	0.07589
7	1.12318	1.33087	4.01998	−0.06513	0.13867
8	1.11718	1.27501	4.01446	0.11408	0.05383
9	1.08519	1.22728	4.00972	−0.04507	0.09927
10	1.08012	1.18504	4.00676	0.08077	0.03678
11	1.05705	1.15150	4.00442	−0.03024	0.06828
12	1.05314	1.12129	4.00297	0.05494	0.02438
13	1.03725	1.09861	4.00190	−0.01977	0.04544
14	1.03444	1.07795	4.00125	0.03623	0.01578
15	1.02386	1.06305	4.00079	−0.01269	0.02950
16	1.02195	1.04940	4.00051	0.02335	0.01005
17	1.01509	1.03981	4.00032	−0.00803	0.01882
18	1.01382	1.03100	4.00020	0.01482	0.00632
19	1.00945	1.02492	4.00012	−0.00503	0.01186
20	1.00863	1.01932	4.00008	0.00930	0.00395

For problems with hundreds or thousands of variables, storing and manipulating the matrices $\tilde{\mathbf{H}}^k$ or $\nabla^2 f(\mathbf{x}^k)$ requires much time and computer memory, making conjugate gradient methods more attractive. These compute $\mathbf{s}^k$ using formulas involving no matrices. The Fletcher–Reeves method uses

$$\mathbf{s}^0 = -\nabla f(\mathbf{x}^0)$$

$$\mathbf{s}^k = -\nabla f(\mathbf{x}^k) + \beta^k \mathbf{s}^{k-1}, \quad k = 1, 2, \ldots$$

where

$$\beta^k = \frac{\nabla f^T(\mathbf{x}^k)\,\nabla f(\mathbf{x}^k)}{\nabla f^T(\mathbf{x}^{k-1})\,\nabla f(\mathbf{x}^{k-1})}$$

The one-step BFGS formula is usually more efficient than the Fletcher–Reeves method. It uses somewhat more complex formulas:

$$\mathbf{s}^k = -\nabla f(\mathbf{x}^0)$$

$$\mathbf{s}^k = -\nabla f(\mathbf{x}^k) - \frac{(\mathbf{y}^{k-1})^T \nabla f(\mathbf{x}^k)}{(\mathbf{y}^{k-1})^T \mathbf{d}^{k-1}} (\mathbf{y}^{k-1} - \mathbf{d}^{k-1})$$

$$+ \frac{(\mathbf{y}^{k-1} - \mathbf{d}^{k-1})^T \mathbf{d}^{k-1} [(\mathbf{y}^{k-1})^T \nabla f(\mathbf{x}^k)]}{[(\mathbf{y}^{k-1})^T \mathbf{d}^{k-1}]^2} \mathbf{y}^{k-1} \quad k = 1, 2, \ldots$$

This formula follows from the BFGS formula for $(\tilde{\mathbf{H}}^k)^{-1}$ by (1) assuming $(\tilde{\mathbf{H}}^{k-1})^{-1} = \mathbf{I}$, (2) computing $(\tilde{\mathbf{H}}^k)^{-1}$ from the update formula, and (3) computing s^k as $-(\tilde{\mathbf{H}}^k)^{-1} \nabla f(\mathbf{x}^k)$. Both methods minimize a positive-definite quadratic function of n variables in at most n iterations using exact line searches but generally require significantly more iterations than the BFGS procedure for general nonlinear functions. A class of algorithms called variable memory quasi-Newton methods (Nash and Sofer, 1996) partially overcomes this difficulty and provides an effective compromise between standard quasi-Newton and conjugate gradient algorithms.

REFERENCES

Armijo, L. "Minimization of Functions Having Lipschitz Continuous First Partial Derivatives." *Pac J Math* **16:** 1–3 (1966).

Avriel, M. *Nonlinear Programming*. Prentice-Hall, Englewood Cliffs, New Jersey (1976).

Broyden, C. G. "The Convergence of a Class of Double-Rank Minimization Algorithms." *J Inst Math Appl* **6:** 76–90 (1970).

Dembo, R. S.; S. C. Eisenstat; and T. Steihang. "Inexact Newton Methods." *SIAM J Num Anal* **19:** 400–408 (1982).

Dennis, J. E.; and R. B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentice-Hall, Englewood Cliffs, New Jersey (1996).

Dixon, L. C. W.; and L. James. "On Stochastic Variable Metric Methods." In *Analysis and Optimization of Stochastic Systems*. Q. L. R. Jacobs et al. eds. Academic Press, London (1980).

Fletcher, R. "A New Approach to Variable Metric Algorithms." *Comput J* **13:** 317 (1970).

Fletcher, R. *Practical Methods of Optimization*, vol. 1. John Wiley, New York (1980).

Fletcher, R.; and C. M. Reeves. "Function Minimization by Conjugate Gradients." *Comput J* **7:** 149–154 (1964).

Gill, P. E.; W. Murray; and M. H. Wright. *Practical Optimization*. Academic Press, New York (1981).

Goldfarb, D. "A Family of Variable Metric Methods Derived by Variational Means." *Math Comput* **24:** 23–26 (1970).

Levenberg, K. "A Method for the Solution of Certain Problems in Least Squares." *Q Appl Math* **2:** 164–168 (1944).

Marquardt, D. "An Algorithm for Least-Squares Estimation of Nonlinear Parameters." *SIAM J Appl Math* **11:** 431–441 (1963).
Nash, S. G.; and A. Sofer. *Linear and Nonlinear Programming*. McGraw-Hill, New York (1996).
Nelder, J. A.; and R. Mead. "A Simplex Method for Function Minimization." *Comput J* **7:** 308–313 (1965).
Powell, M. J. D. "A New Algorithm for Unconstrained Optimization." In *Nonlinear Programming*. J. B. Rosen; O. L. Mangasarian; and K. Ritter, eds. pp. 31–65, Academic Press, New York (1970).
Shanno, D. F. "Conditioning of Quasi-Newton Methods for Function Minimization." *Math Comput* **24:** 647–657 (1970).
Spendley, W.; G. R. Hext; and F. R. Himsworth. "Sequential Application of Simplex Designs in Optimization and Evolutionary Operations." *Technometrics* **4:** 441–461 (1962).
Uchiyama, T. "Best Size for Refinery and Tankers." *Hydrocarbon Process.* **47**(12): 85–88 (1968).

SUPPLEMENTARY REFERENCES

Brent, R. P. *Algorithms for Minimization Without Derivatives*. Prentice-Hall, Englewood Cliffs, New Jersey (1973).
Broyden, C. G. "Quasi-Newton Methods and Their Application to Function Minimization." *Math Comput* **21:** 368 (1967).
Boggs, P. T.; R. H. Byrd; and R. B. Schnabel. *Numerical Optimization*. SIAM. Philadelphia (1985).
Hestenes, M. R. *Conjugate-Direction Methods in Optimization*. Springer-Verlag, New York (1980).
Kelley, C. T. *Iterative Methods for Optimization*. SIAM, Philadelphia (1999).
Li, J.; and R. R. Rhinehart. "Heuristic Random Optimization." *Comput Chem Engin* **22:** 427–444 (1998).
Powell, M. J. D. "An Efficient Method for Finding the Minimum of a Function of Several Variables Without Calculating Derivatives." *Comput J* **7:** 155–162 (1964).
Powell, M. J. D. "Convergence Properties of Algorithms to Nonlinear Optimization." *SIAM Rev* **28:** 487–496 (1986).
Reklaitis, G. V.; A. Ravindran; and K. M. Ragsdell. *Engineering Optimization—Methods and Applications*. John Wiley, New York (1983).
Schittkowski, K. *Computational Mathematical Programming*. Springer-Verlag, Berlin (1985).

PROBLEMS

6.1 If you carry out an exhaustive search (i.e., examine each grid point) for the optimum of a function of five variables, and each step is 1/20 of the interval for each variable, how many objective function calculations must be made?

6.2 Consider the following minimization problem:

$$\text{Minimize: } f(\mathbf{x}) = x_1^2 + x_1x_2 + x_2^2 + 3x_1$$

(a) Find the minimum (or minima) analytically.
(b) Are they global or relative minima?
(c) Construct four contours of $f(\mathbf{x})$ [lines of constant value of $f(\mathbf{x})$].
(d) Is univariate search a good numerical method for finding the optimum of $f(\mathbf{x})$? Why or why not?
(e) Suppose the search direction is given by $\mathbf{s} = [1 \ \ 0]^T$. Start at (0,0), find the optimum point P_1 in that search direction analytically, not numerically. Repeat the exercise for a starting point of (0,4) to find P_2.
(f) Show graphically that a line connecting P_1 and P_2 passes through the optimum.

6.3 Determine a regular simplex figure in a three-dimensional space such that the distance between vertices is 0.2 unit and one vertex is at the point $(-1, 2, -2)$.

6.4 Carry out the four stages of the simplex method to minimize the function

$$f(\mathbf{x}) = x_1^2 + 3x_2^2$$

starting at $\mathbf{x} = [1 \ \ 1.5]^T$. Use $\mathbf{x} = [1 \ \ 2]^T$ for another corner. Show each stage on a graph.

6.5 A three-dimensional simplex optimal search for a minimum provides the following intermediate results:

x vector			Value of objective function
$[\ 0$	0	$0]^T$	4
$[-\frac{4}{3}$	$-\frac{1}{3}$	$-\frac{1}{3}]^T$	7
$[-\frac{1}{3}$	$-\frac{4}{3}$	$-\frac{1}{3}]^T$	10
$[-\frac{1}{3}$	$-\frac{1}{3}$	$-\frac{4}{3}]^T$	5

What is the next point to be evaluated in the search? What point is dropped?

6.6 Find a direction orthogonal to the vector

$$\mathbf{s} = \left[\frac{1}{\sqrt{3}} \quad -\frac{1}{\sqrt{3}} \quad -\frac{1}{\sqrt{3}}\right]^T$$

at the point

$$\mathbf{x} = [0 \ \ 0 \ \ 0]^T$$

Find a direction conjugate to $\mathbf{s}$ with respect to the Hessian matrix of the objective function $f(\mathbf{x}) = x_1 + 2x_2^2 - x_1x_2$ at the same point.

6.7 Given the function $f(\mathbf{x}) = x_1^2 + x_2^2 + 2x_3^2 - x_1x_2$, generate a set of conjugate directions. Carry out two stages of the minimization in the conjugate directions minimizing $f(\mathbf{x})$ in each direction. Did you reach the minimum of $f(\mathbf{x})$? Start at (1, 1, 1).

6.8 For what values of $\mathbf{x}$ are the following directions conjugate for the function $f(\mathbf{x}) = x_1^2 + x_1x_2 + 16x_2^2 + x_3^2 - x_1x_2x_3$?

$$\mathbf{s}^{(1)} = \begin{bmatrix} -\frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{3}} \\ -\frac{1}{\sqrt{3}} \end{bmatrix} \qquad \mathbf{s}^{(2)} = \begin{bmatrix} -\frac{1}{\sqrt{3}} \\ \frac{2}{\sqrt{3}} \\ 0 \end{bmatrix}$$

6.9 In the minimization of

$$f(\mathbf{x}) = 5x_1^2 + x_2^2 + 2x_1x_2 - 12x_1 - 4x_2 + 8$$

starting at $(0, -2)$, find a search direction $\mathbf{s}$ conjugate to the x_1 axis. Find a second search vector $\mathbf{s}_2$ conjugate to $\mathbf{s}_1$.

6.10 (a) Find two directions respectively orthogonal to

$$\mathbf{x}^T = \left[\frac{2}{3}, -\frac{1}{3}, -\frac{2}{3}\right]$$

and each other.

(b) Find two directions respectively conjugate to the vector in part (a) and to each other for the given matrix

$$\begin{bmatrix} 2 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 3 \end{bmatrix}$$

6.11 The starting search direction from $\mathbf{x} = [2\ 2]^T$ to minimize

$$f(\mathbf{x}) = x_1^2 + x_1x_2 + x_2^2 - 3x_1 - 3x_2$$

is the negative gradient. Find a conjugate direction to the starting direction. Is it unique?

6.12 Evaluate the gradient of the function

$$f(\mathbf{x}) = (x_1 + x_2)^3x_3 + x_3^2x_1^2x_2^2$$

at the point $\mathbf{x} = [1\ \ 1\ \ 1]^T$.

6.13 You are asked to maximize

$$f(\mathbf{x}) = x_1 + x_2 - \tfrac{1}{2}(x_1^2 + 2x_1x_2 + 2x_2)$$

Begin at $\mathbf{x} = [1 \ \ 1]^T$, and select the gradient as the first search direction. Find a second search direction that is conjugate to the first search direction. (Do not continue after getting the second direction.)

6.14 You wish to minimize

$$f(\mathbf{x}) = 10x_1^2 + x_2^2$$

If you use steepest descent starting at (1, 1), will you reach the optimum in
(a) One iteration
(b) Two iterations
(c) More that two?

Explain.

6.15 Evaluate the gradient of the function

$$f(\mathbf{x}) = e^{x_1x_2} - 2e^{x_1} + 2e^{x_2} + (x_1x_2)^2$$

at the point (0, 0).

6.16 Consider minimizing the function $f(\mathbf{x}) = x_1^2 + x_2^2$. Use the formula $\mathbf{x}^{k+1} = \mathbf{x}^k - \alpha\nabla f(\mathbf{x}^k)$, where α is chosen to minimize $f(\mathbf{x})$. Show that $\mathbf{x}^{k+1}$ will be the optimum $\mathbf{x}$ after only one iteration. You should be able to optimize $f(\mathbf{x})$ with respect to α analytically. Start from

$$\mathbf{x}^0 = \begin{bmatrix} 3 \\ 5 \end{bmatrix}$$

6.17 Why is the steepest descent method not widely used in unconstrained optimization codes?

6.18 Use the Fletcher–Reeves search to find the minimum of the objective function

(*a*) $f(\mathbf{x}) = 3x_1^2 + x_2^2$

(*b*) $f(\mathbf{x}) = 4(x_1 - 5)^2 + (x_2 - 6)^2$

starting at $\mathbf{x}^0 = [1 \ \ 1]^T$.

6.19 Discuss the advantages and disadvantages of the following two search methods for the function shown in Figure P6.19.
(a) Steepest descent
(b) Conjugate gradient

Discuss the basic idea behind each of the two methods (don't write out the individual steps, though). Be sure to consider the significance of the starting point for the search.

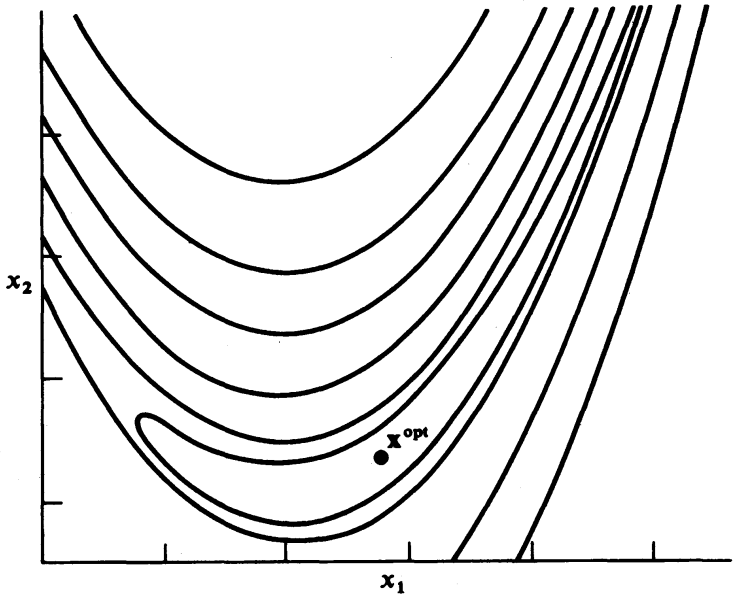


FIGURE P6.19

6.20 Repeat Problem 6.18 for the Woods function.

$$f = \sum_{i=1}^{4} F_i(\mathbf{x})$$

where $F_1(x) = -200x_1(x_4 - x_3^2) - (1 - x_1)$

$F_2(x) = 200(x_2 - x_1^2) + 20(x_2 - 1) + 19.8(x_4 - 1)$

$F_3(x) = -180x_3(x_4 - x_3^2) - (1 - x_3)$

$F_4(x) = (-3, -1, -3, -1)$

6.21 An open cylindrical vessel is to be used to store 10 ft^3 of liquid. The objective function for the sum of the operating and capital costs of the vessel is

$$f(h,r) = \frac{1}{\pi r^2 h} 2\pi rh + 10\pi r^2$$

Can Newton's method be used to minimize this function? The solution is $[r^* \ h^*]^T = [0.22 \ 2.16]^T$.

6.22 Is it necessary that the Hessian matrix of the objective function always be positive-definite in an unconstrained minimization problem?

6.23 Cite two circumstances in which the use of the simplex method of multivariate unconstrained optimization might be a better choice than a quasi-Newton method.

6.24 Given the function $f(\mathbf{x}) = 3x_1^2 + 3x_2^2 + 3x_3^2$ to minimize, would you expect that steepest descent or Newton's method (in which adjustment of the step length is used for minimization in the search direction) would be faster in solving the problem from the same starting point $\mathbf{x} = [10 \ 10 \ 10]^T$? Explain the reasons for your answer.

6.25 Consider the following objective functions:

(a) $f(\mathbf{x}) = 1 + x_1 + x_2 + \dfrac{4}{x_1} + \dfrac{9}{x_2}$

(b) $f(\mathbf{x}) = (x_1 + 5)^2 + (x_2 + 8)^2 + (x_3 + 7)^2 + 2x_1^2x_2^2 + 4x_1^2x_3^2$

Will Newton's method converge for these functions?

6.26 Consider the minimization of the objective function

$$f(\mathbf{x}) = x_1^3 + x_1x_2 - x_2^2x_1^2$$

by Newton's method starting from the point $\mathbf{x}^0 = [1 \;\; 1]^T$. A computer code carefully programmed to execute Newton's method has not been successful. Explain the probable reason(s) for the failure.

6.27 What is the initial direction of search determined by Newton's method for $f(\mathbf{x}) = x_1^2 + 2x_2^2$? What is the step length? How many steps are needed to minimize $f(\mathbf{x})$ analytically?

6.28 Will Newton's method minimize Rosenbrock's function

$$f(\mathbf{x}) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$$

starting at $\mathbf{x}^0 = [-1.2 \;\; 1.0]^T$ in one stage? How many stages will it take if you minimize $f(\mathbf{x})$ exactly on each stage? How many stages if you let the step length be unity on each stage?

6.29 Find the minimum of the following objective function by (a) Newton's method or (b) Fletcher–Reeves conjugate gradient

$$f(\mathbf{x}) = 8x_1^2 + 4x_1x_2 + 5x_2^2.$$

starting at $\mathbf{x}^T = [10 \;\; 10]$.

6.30 Solve the following problems by Newton's method:

Minimize:

(a) $f(\mathbf{x}) = 1 + x_1 + x_2 + x_3 + x_4 + x_1x_2 + x_1x_3 + x_1x_4 + x_2x_3 + x_2x_4 + x_3x_4 + x_1^2 + x_2^2 + x_3^2 + x_4^2$

starting from

$\mathbf{x}^0 = [-3 \quad -30 \quad -4 \quad -0.1]^T$ and also $\mathbf{x}^0 = [0.5 \quad 1.0 \quad 8.0 \quad -0.7]^T$

(b) $f(\mathbf{x}) = x_1x_2^2x_3^3x_4^4\,[\exp - (x_1 + x_2 + x_3 + x_4)]$

starting from

$$\mathbf{x}^0 = [3 \quad 4 \quad 0.5 \quad 1]^T$$

6.31 List the relative advantages and disadvantages (there can be more than one) of the following methods for a two-variable optimization problem such as Rosenbrock's "banana" function (see Fig. P6.19)
(a) Sequential simplex
(b) Conjugate gradient
(c) Newton's method

Would your evaluation change if there were 20 independent variables in the optimization problem?

6.32 Find the maximum of the function $f(\mathbf{x}) = 100 - (10 - x_1)^2 - (5 - x_2)^2$ by the
(a) Simplex method
(b) Newton's method
(c) BFGS method

Start at $\mathbf{x}^T = [0\ \ 0]$. Show all equations and intermediate calculations you use. For the simplex method, carry out only five stages of the minimization.

6.33 For the function $f(\mathbf{x}) = (x - 100)^2$, use
(a) Newton's method
(b) Quasi-Newton method
(c) Quadratic interpolation

to minimize the function. Show all equations and intermediate calculations you use. Start at $\mathbf{x} = 0$.

6.34 For the function $f(\mathbf{x}) = (x - 100)^3$, use
(a) Steepest descent
(b) Newton's method
(c) Quasi-Newton method
(d) Quadratic interpolation

to minimize the function. Show all equations and intermediate calculations you use. Start at $\mathbf{x} = 0$.

6.35 How can the inverse of the Hessian matrix for the function

$$f(\mathbf{x}) = 2x_1^2 - 2x_2^2 - x_1x_2$$

be approximated by a positive-definite matrix using the method of Marquardt?

6.36 You are to minimize $f(\mathbf{x}) = 2x_1^2 - 4x_1x_2 + x_2^2$.
Is $\mathbf{H}(\mathbf{x})$ positive-definite? If not, start at $\mathbf{x}^0 = [2\ \ 2]^T$, and develop an approximation of $\mathbf{H}(\mathbf{x})$ that is positive-definite by Marquardt's method.

6.37 Show how to make the Hessian matrix of the following objective function positive-definite at $\mathbf{x} = [1\ \ 1]^T$ by using Marquardt's method:

$$f(\mathbf{x}) = 2x_1^3 - 6x_1x_2 + x_2^2$$

6.38 The Hessian matrix of the following function

$$f(\mathbf{x}) = u_1^2 + u_2^2 + u_3^2$$

where $u_1 = 1.5 - x_1(1 - x_2)$
$u_2 = 2.25 - x_1(1 - x_2^2)$
$u_3 = 2.625 - x_1(1 - x_2^3)$

is not positive-definite in the vicinity of $\mathbf{x} = [0\ 1]^T$ and Newton's method will terminate at a saddle point if started there. If you start at $\mathbf{x} = [0\ 1]^T$, what procedure should you carry out to make a Newton or quasi-Newton method continue with searches to reach the optimum, which is in the vicinity of $\mathbf{x} = [3\ 0.5]^T$?

6.39 Determine whether the following statements are true or false, and explain the reasons for your answer.

(a) All search methods based on conjugate directions (e.g., Fletcher–Reeves method) always use conjugate directions.

(b) The matrix, or its inverse, used in the BFGS relation, is an approximation of the Hessian matrix, or its inverse, of the objective function $[\nabla^2 f(\mathbf{x})]$.

(c) The BFGS version has the advantage over a pure Newton's method in that the latter requires second derivatives, whereas the former requires only first derivatives to get the search direction.

6.40 For the quasi-Newton method discussed in Section 6.4, give the values of the elements of the approximate to the Hessian (inverse Hessian) matrix for the first two stages of search for the following problems:

(a) Maximize: $f(\mathbf{x}) = -x_1^2 + x_1 - x_2^2 + x_2 + 4$

(b) Minimize: $f(\mathbf{x}) = x_1^3 \exp[x_2 - x_1^2 - 10(x_1 - x_2)^2]$

$$f(\mathbf{x}) = x_1^2 + x_2^2 + x_3^2 + x_4^2$$

starting from the point (1, 1) or (1, 1, 1, 1) as the case may be.

6.41 Estimate the values of the parameters k_1 and k_2 by minimizing the sum of the squares of the deviations

$$\phi = \sum_{i=1}^{n} (y_{\text{observed}} - y_{\text{predieted}})_i^2$$

where

$$y_{\text{predicted}} = \frac{k_1}{k_1 - k_2}(e^{-k_2 t} - e^{-k_1 t})$$

for the following data:

t	y_{observed}
0.5	0.263
1.0	0.455
1.5	0.548

Plot the sum-of-squares surface with the estimated coefficients.

6.42 Repeat Problem 6.41 for the following model and data:

$$y = \frac{k_1 x_1}{1 + k_2 x_1 + k_3 x_2}$$

$y_{observed}$	x_1	x_2
0.126	1	1
0.219	2	1
0.076	1	2
0.126	2	2
0.186	0.1	0

6.43 Approximate the minimum value of the integral

$$\int_0^1 \left[\left(\frac{dy}{dx} \right)^2 - 2yx^2 \right] dx$$

subject to the boundary conditions $dy/dx = 0$ at $x = 0$ and $y = 0$ at $x = 1$.

Hint: Assume a trial function $y(x) = a(1 - x^2)$ that satisfies the boundary conditions and find the value of a that minimizes the integral. Will a more complicated trial function that satisfies the boundary conditions improve the estimate of the minimum of the integral?

6.44 In a decision problem it is desired to minimize the expected risk defined as follows:

$$\varepsilon\{\text{risk}\} = (1 - P)c_1[1 - F(b)] + Pc_2\theta\left(\frac{b}{2} + \frac{2\pi}{4}\right)F\left(\frac{b}{2} - \frac{\sqrt{2\pi}}{4}\right)$$

where $F(b) = \int_{-\infty}^{b} e^{-u^2/2\theta^2} du$ (normal probability function)

$$c_1 = 1.25 \times 10^5$$

$$c_2 = 15$$

$$\theta = 2000$$

$$P = 0.25$$

Find the minimum expected risk and b.

6.45 The function

$$f(\mathbf{x}) = (1 + 8x_1 - 7x_1^2 + \tfrac{7}{3}x_1^3 - \tfrac{1}{4}x_1^4)(x_2^2 e^{-x_2})F(\mathbf{x}_3)$$

has two maxima and a saddle point. For (a) $F(\mathbf{x}_3) = 1$ and (b) $F(\mathbf{x}_3) = x_3 e^{-(x_3+1)}$, locate the global optimum by a search technique.

Answer: (a) $\mathbf{x}^* = [4\ \ 2]^T$ and (b) $\mathbf{x}^* = [4\ \ 2\ \ 1]^T$.

6.46 By starting with (a) $\mathbf{x}^0 = [2\ 1]^T$ and (b) $\mathbf{x}^0 = [2\ 1\ 1]^T$, can you reach the solution for Problem 6.45? Repeat for (a) $\mathbf{x}^0 = [2\ 2]^T$ and (b) $\mathbf{x}^0 = [2\ 2\ 1]^T$.

Hint: [2 2 1] is a saddle point.

6.47 Estimate the coefficients in the correlation

$$y = ax_1^{b_1}x_2^{b_2}$$

from the following experimental data by minimizing the sum of the square of the deviations between the experimental and predicted values of y.

y_{exptl}	x_1	x_2
46.5	2.0	36.0
591	6.0	8.0
1285	9.0	3.0
36.8	2.5	6.25
241	4.5	7.84
1075	9.5	1.44
1024	8.0	4.0
151	4.0	7.0
80	3.0	9.0
485	7.0	2.0
632	6.5	5.0

6.48 The cost of refined oil when shipped via the Malacca Straits to Japan in dollars per kiloliter was given (Uchiyama, 1968) as the linear sum of the crude oil cost, the insurance, customs, freight cost for the oil, loading and unloading cost, sea berth cost, submarine pipe cost, storage cost, tank area cost, refining cost, and freight cost of products as

$$c = c_c + c_i + c_x + \frac{2.09 \times 10^4 t^{-0.3017}}{360} + \frac{1.064 \times 10^6 at^{0.4925}}{52.47q(360)}$$

$$+ \frac{4.242 \times 10^4 at^{0.7952} + 1.813ip(nt + 1.2q)^{0.861}}{52.47q(360)}$$

$$+ \frac{4.25 \times 10^3 a(nt + 1.2q)}{52.47q(360)} + \frac{5.042 \times 10^3 q^{-0.1899}}{360}$$

$$+ \frac{0.1049q^{0.671}}{360}$$

where a = annual fixed charges, fraction (0.20)
c_c = crude oil price, \$/kL (12.50)
c_i = insurance cost, \$/kL (0.50)
c_x = customs cost, \$/kL (0.90)
i = interest rate (0.10)

n = number of ports (2)
p = land price, \$/m^2 (7000)
q = refinery capacity, bbl/day
t = tanker size, kL

Given the values indicated in parentheses, use a computer code to compute the minimum cost of oil and the optimum tanker size t and refinery size q by Newton's method and the quasi-Newton method (note that 1 kL = 6.29 bbl).

(The answers in the reference were

$$t = 427{,}000 \text{ dwt} \approx 485{,}000 \text{ kL}$$

$$q = 185{,}000 \text{ bbl/day})$$