2.12 Unconstrained Optimization

Optimization refers to finding the maximum or minimum of a continuous function $f(x_1,\ldots,x_m)$. This is an extremely important problem, lying at the heart of modern industrial engineering, management science, and other areas. This section discusses some methods and perspectives for calculating the minimum or maximum of a function $f(x_1,\ldots,x_m)$. No formal algorithms are given, since this would require too extensive of a development.

Vector notation is used in much of the presentation, to give results for a general number m of variables. We consider only the unconstrained optimization problem, in which there are no limitations on $(x_1,\ldots,x_m)$. For simplicity only, we also assume $f(x_1,\ldots,x_m)$ is defined for all $(x_1,\ldots,x_m)$.

Because the behaviour of a function $f(\mathbf{x})$ can be quite varied, the problem must be further limited. A point $\boldsymbol{\alpha}$ is called a strict local minimum of f if $f(\mathbf{x})>f(\boldsymbol{\alpha})$ for all x close to $\boldsymbol{\alpha}$, $\mathbf{x}\neq\boldsymbol{\alpha}$. We will limit ourselves to finding a strict local minimum of $f(\mathbf{x})$. Generally an initial guess $\mathbf{x}_0$ of α will be known; and $f(\mathbf{x})$ will be assumed to be twice continuously differentiable with respect to its variables $x_1,\ldots x_m$

Reformulation as a nonlinear system With the assumption of differentiability, a necessary condition for $\boldsymbol{\alpha}$ to be a strict local minimum is that

$$\frac{\partial f(\boldsymbol{\alpha})}{\partial x_i} = 0. \qquad i=1,\ldots,m \tag{2.12.1}$$

Thus the nonlinear system

$$\frac{\partial f(\mathbf{x})}{\partial x_i} = 0, \qquad i=1,\dots,m \tag{2.12.2}$$

can be solved, and each calculated solution can be checked as to whether it is a local maximum, minimum, or neither. For notation, introduce the gradient vector

$$\nabla f(\mathbf{x}) = \begin{bmatrix} \partial f/\partial x_1 \\ \vdots \\ \partial f/\partial x_m \end{bmatrix}$$

Using this, the system (2.12.2) is written more compactly as

$$\nabla f(\mathbf{x}) = 0 \tag{2.12.3}$$

To solve (2.12.2), Newton's method (2.11.4) can be used, as well as other rootfinding methods for nonlinear systems. Using Newton's method leads to

$$\mathbf{x}_{n+1} = \mathbf{x}_n - H(\mathbf{x}_n)^{-1}\nabla f(\mathbf{x}_n), \quad n\geq 0 \tag{2.12.4}$$

with $H(\mathbf{x})$ the Hessian matrix of f,

$$H(\mathbf{x})_{ij} = \frac{\partial^2 f(\mathbf{x})}{\partial x_i \partial x_j}, \qquad 1\leq i,j\leq m.$$

If α is a strict local minimum of f, then Taylor's theorem (1.1.12) can be used to show that $H(\alpha)$ is a nonsingular matrix; and then $H(\mathbf{x})$ will be nonsingular for $\mathbf{x}$ close to α. For convergence, the analysis of Newton's method in the preceding section can be used to prove quadratic convergence of $\mathbf{x}_n$ to α is chosen sufficiently close to α.

The main drawbacks with the iteration (2.12.4) are the same as those given in the last section for Newton's method for solving nonlinear systems. There are other, more efficient, optimization methods that seek to approximate α by using only $f(\mathbf{x})$ and $\nabla f(\mathbf{x})$. These methods may require more iterations, but generally their total computing time will be much less than with

Newton's method. In addition, these methods seek to obtain convergence for a larger set of initial values x_0.

Descent methods Suppose we are trying to minimize a function $f(x)$. Most methods for doing so are based on the following general two step iteration process.

Step D1: At x_n, pick a direction d_n such that $f(x)$ will decrease as x moves away from x_n in the direction d_n.

Step D2: Let $x_{n+1} = x_n + sd_n$, with s chosen to minimize

$$\varphi(s) = f(x_n + sd_n), \quad s \geq 0 \qquad (2.12.5)$$

Usually s is chosen as the smallest positive relative minimum of $\varphi(s)$.

Such methods are called descent methods. With each iteration,

$$f(x_{n+1}) < f(x_n).$$

Descent methods are guaranteed to converge under more general conditions than for Newton's method (2.12.4). Consider the level surface

$$C = \{x \mid f(x)=f(x_0)\},$$

and consider only the connected portion of it that contains x_0. Then if C is bounded, descent methods will converge under very general conditions. For the two variable case, this is illustrated in Figure 2.7. Several level curves $f(x_1,x_2)=c$ are shown for a set of values c approaching $f(\alpha)$. The vectors d_n are directions in which $f(x)$ is decreasing.

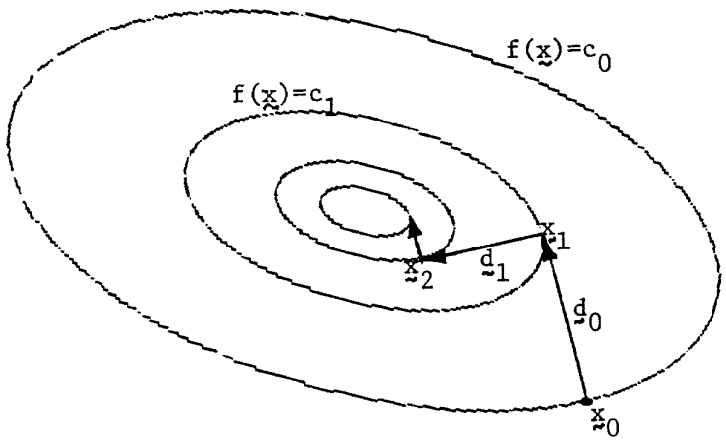

Figure 2.7 Illustration of descent method

There are a number of ways for choosing the directions $\mathbf{d}_n$, and the best known are as follows:

1. The method of steepest descent. Here $\mathbf{d}_n=-\nabla f(\mathbf{x}_n)$. It is the direction in which $f(\mathbf{x})$ decreases most rapidly when moving away from $\mathbf{x}_n$. It is a good strategy near $\mathbf{x}_n$, but it usually turns out to be a poor strategy for rapid convergence to α.
2. Quasi-Newton methods. These methods can be viewed as approximations of Newton's method (2.12.4). They use easily computable approximations of $H(\mathbf{x}_n)$ or $H(\mathbf{x}_n)^{-1}$, and they are also descent methods. The best known examples are the Davidon-Fletcher-Powell method and the Broyden methods.
3. The conjugate gradient method. This uses a generalization of the idea of an orthogonal basis for a vector space to generate the directions $\mathbf{d}_n$, with the directions related in an optimal way to the function $f(\mathbf{x})$ being minimized. In Chapter 8, the conjugate gradient method will be used for solving systems of linear equations.

There are many other approaches to minimizing a function, but they are too numerous to include here. As general references

to the above ideas, see Dennis-Schnabel (1983), Fletcher (1980), Gill-Murray-Wright (1981), and Luenberger (1984). An important and very different approach to minimizing a function is the simplex method given in Nelder-Mead (1965), with a discussion given in Gill-Murray-Wright (1981,p.94) and Woods (1985, Chapter 2). It uses only function values (no derivative values), and it seems to be especially suitable for noisy functions.

An important project to develop programs for solving optimization problems and nonlinear systems is under way at Argonne National Laboratory. The program package is called MINPACK, and version 1 is available; see Moré-Garbow-Hillstrom (1980) and Moré-Sorenson-Garbow-Hillstrom (1984). It contains routines for nonlinear systems and nonlinear least squares problems. Future versions are intended to include programs for both unconstrained and constrained optimization problems.

One variable minimization Motivated by the need to minimize the function $\varphi(s)$ in (2.12.5), we consider the problem of minimizing a function $f(x)$ of a single real variable x. As with systems, the problem can be converted to a rootfinding problem,

$$f'(x) = 0,$$

which can be solved by applying the methods given earlier in this chapter. In contrast, we will only consider minimization methods that do not use $f'(x)$.

The first method to be discussed is comparable in some ways to the bisection method for rootfinding. Beginning with an interval $[a,b]$ that contains the desired minimum point α, the interval is reduced to smaller intervals, each of which will also

contain α. In addition, the length of the interval will be reduced by a fixed percentage with each iteration.

We will assume that f(x) is <u>unimodal</u> on an interval [a,b], in which the minimum α is located. For f to be unimodal means there is an $\alpha \in [a,b]$ with

$$\begin{aligned} a \le x \le y \le \alpha &\Rightarrow f(x) > f(y) \\ \alpha \le x < y \le b &\Rightarrow f(x) < f(y) \end{aligned} \tag{2.12.6}$$

Thus α is a unique local and global minimum of f(x) on [a,b].

The principle used for reducing the interval [a,b] to a smaller interval is as follows. For f unimodal on [a,b] and for $a < x_1 < x_2 < b$,

$$\begin{aligned} f(x_1) \ge f(x_2) &\Rightarrow x_1 \le \alpha \le b \\ f(x_1) \le f(x_2) &\Rightarrow a \le \alpha \le x_2 \end{aligned} \tag{2.12.7}$$

Let $[a_{i-1}, b_{i-1}]$ and $[a_i, b_i]$ denote the intervals in use at the beginning and end of step i, respectively. We will require that

$$\frac{b_i - a_i}{b_{i-1} - a_{i-1}} = \lambda, \quad i \ge 1, \tag{2.12.8}$$

for some constant $\lambda < 1$, fixed independently of i. Let $x_1^{(i-1)}$, $x_2^{(i-1)}$ be two points in $[a_{i-1}, b_{i-1}]$, with $x_1^{(i-1)} < x_2^{(i-1)}$. These two points will be used in conjunction with (2.12.7) to obtain a smaller subinterval $[a_i, b_i]$ containing α. Specifically,

$$\begin{aligned} f(x_1^{(i-1)}) \ge f(x_2^{(i-1)}) &\Rightarrow [a_i, b_i] := [x_1^{(i-1)}, b_{i-1}] \\ f(x_1^{(i-1)}) < f(x_2^{(i-1)}) &\Rightarrow [a_i, b_i] := [a_{i-1}, x_2^{(i-1)}] \end{aligned} \tag{2.12.9}$$

In addition, the points $x_1^{(i-1)}$ and $x_2^{(i-1)}$ are to be chosen so

that (2.12.8) is satisfied. This requires

$$x_2^{(i-1)}-a_{i-1} = b_{i-1}-x_1^{(i-1)} = \lambda(b_{i-1}-a_{i-1}) \qquad (2.12.10)$$

With this, the above algorithm will result in a decrease by a factor of λ in the length of the interval at each step. From (2.12.8) and the ordering of the points x_1 and x_2, the factor $\lambda > \frac{1}{2}$.

With a given $\frac{1}{2}<\lambda<1$, the points x_1 and x_2 can be chosen from (2.12.10) at each step. The test (2.12.7) will then require two evaluations of f per iteration. A natural question is whether λ can be so chosen that there is only one evaluation of f per iterate. This is illustrated in Figure 2.8. To find the needed value of λ, we consider the first step. For definiteness, suppose that

$$[a_1,b_1] = [a_0,x_2^{(0)}] \qquad (2.12.13)$$

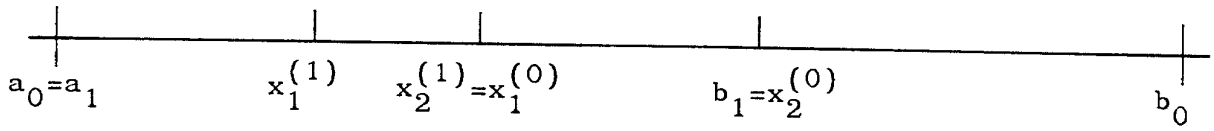

Figure 2.8 The golden section search

Then

$$\frac{x_2^{(0)}-a_0}{b_0-a_0} = \frac{b_0-x^{(0)}}{b_0-a_0} = \lambda \qquad (2.12.14)$$

We want the point $x_1^{(0)}$, which is interior to $[a_1,b_1]$, to be one of the points $x_1^{(1)},x_2^{(1)}$. With this, $f(x_1^{(0)})$ can be used again in the next step of the iteration.

It is straightforward to show that $x_1^{(1)}=x_1^{(0)}$ is not possible. Thus we want $x_2^{(1)}=x_1^{(0)}$, as shown in Figure 2.8. From

(2.12.8) and (2.12.13),

$$\lambda = \frac{x_2^{(1)}-a_1}{b_1-a_1} = \frac{x_1^{(0)}-a_0}{\lambda(b_0-a_0)}$$

$$\lambda^2 = \frac{x_1^{(0)}-a_0}{b_0-a_0} = 1 - \frac{b_0-x_1^{(0)}}{b_0-a_0} = 1-\lambda$$

The positive solution of $\lambda^2 = 1-\lambda$ is

$$\lambda = \frac{\sqrt{5}-1}{2} \doteq .618, \qquad (2.12.15)$$

which is called the golden mean.

With this value of λ. the above algorithm requires only one evaluation of f per iterate (except for the first step). In this case, the above algorithm is called the golden section search. The length of $[a_i,b_i]$ decreases by a factor of about .618 per step, allowing a rigorous error bound for α. If f(x) is not uninodal on [a,b], then this algorithm will still converge to a relative minimum of f on [a,b], but it may not be the absolute minimum of f on [a,b].

Example Minimize $f(x) = x^2 + 1/x$. Table 2.21 contains the results of applying the golden section search to the minimization

Table 2.21 Examples of golden section search

i	a_i	b_i	i	a_i	b_i
0	.20000	1.00000	8	.78359	.80062
1	.50557	1.00000	9	.79010	.80062
2	.69443	1.00000	10	.79010	.79660
3	.69443	.88328	11	.79258	.79660
4	.76656	.88328	12	.79258	.79507
5	.76656	.83870	13	.79258	.79412
6	.76656	.81115	14	.79317	.79412
7	.78359	.81115	15	.79353	.79412

of this f(x) on [.2,1]. To obtain an interval of length $\leq 10^{-7}$ will require 33 iterates. In the table, $b_{15}-a_{15} = .000587$. The iteration converges steadily, but slowly.

A second method for minimizing f(x) is based on using quadratic interpolation. Suppose that x_1,x_2,x_3 are given estimates of α. The quadratic polynomial that interpolates f(x) at x_1,x_2,x_3 is given by

$$P(x)=\frac{(x-x_2)(x-x_3)}{(x_1-x_2)(x_1-x_3)}f(x_1)+\frac{(x-x_1)(x-x_3)}{(x_2-x_1)(x_2-x_3)}f(x_2)+\frac{(x-x_1)(x-x_2)}{(x_3-x_1)(x_3-x_2)}f(x_3).$$

The topic of interpolation is not taken up until Chapter 3, but this formula can be checked directly by substitution to show that $P(x_i)=f(x_i)$ for i=1,2,3. We use the minimum of P(x) to approximate the minimum of f(x). For points x_1,x_2,x_3 sufficiently close to α, this will be valid, and the minimum of P(x) is given by

$$x_4 = \frac{1}{2}\cdot\frac{y_{23}f(x_1)+y_{31}f(x_2)+y_{12}f(x_3)}{x_{23}f(x_1)+x_{31}f(x_2)+x_{12}f(x_3)} \qquad (2.12.16)$$

where $x_{ij}=x_i-x_j$, $y_{ij}=x_i^2-x_j^2$. Next use x_2,x_3,x_4 to find x_5, and continue in a like manner. We will call this the <u>quadratic fit method</u>.

For points x_1,x_2,x_3 chosen sufficiently close to α, and for f(x) sufficiently differentiable, it can be shown that this method converges. Moreover, the error will satisfy

$$|\alpha-x_{n+1}| \leq C|\alpha-x_n|^r \qquad (2.12.17)$$

for some constant C, with $r\doteq 1.3$. This shows that the quadratic fit method will converge more rapidly than the golden section

search, once the iterates are sufficiently close to α.

Example Let $f(x) = x^2 + 1/x$, as in the last example. The results of using the quadratic fit method are shown in Table 2.22. They were computed in double precision Fortran on an IBM-PC.

Table 2.22 Example of quadratic fit method

i	x_i	$f(x_i)$	$\alpha - x_i$
1	.2	5.04	5.94E-1
2	1.0	2.0	-2.06E-1
3	.6	2.026666667	1.94E-1
4	.8035714286	1.890171485	-9.87E-3
5	.8108433735	1.890750780	-1.71E-2
6	.7960124360	1.889897579	-2.31E-3
7	.7936042221	1.889881603	9.63E-5
8	.7936847983	1.889881576	1.57E-5
9	.7937006340	1.889881575	-1.08E-7
10	.7937005254	1.889881575	6.36E-10

A program combining the golden section search and the quadratic fit method is given in Brent (1973, Chapter 5). It has the guaranteed convergence of the golden section search while generally it will converge with the speed of the quadratic fit method. It also takes into account the effects of using computer arithmetic. The program is available in many widely used computer libraries.

Uncertainty in calculating the minimum The calculation of a minimum will contain significant uncertainty if the derivative is not used. To see this, we consider the case of minimizing $f(x)$, a function of a real variable. At the minimum α,

$$f(\alpha+\epsilon) = f(\alpha) + \epsilon f'(\alpha) + \frac{\epsilon^2}{2} f''(\xi),$$

with ξ between α and $\alpha+\epsilon$. Since α is a minimum, $f'(\alpha)=0$. Thus

$$f(\alpha+\epsilon) \doteq f(\alpha) + \frac{\epsilon^2}{2} f''(\alpha), \qquad (2.12.18)$$

if ϵ is sufficiently small.

To the machine,

$$f(\alpha+\epsilon) = f(\alpha) \qquad (2.12.19)$$

if ϵ is sufficiently small. Using the unit round δ [see (1.2.11)], this will be true if

$$\frac{\epsilon^2}{2} \cdot |f''(\alpha)| \leq |f(\alpha)|\delta,$$

assuming $f(\alpha)\neq 0$, $f''(\alpha)\neq 0$. This is equivalent to

$$|\epsilon| \leq \left[\frac{2\delta|f(\alpha)|}{|f''(\alpha)|}\right]^{1/2} \qquad (2.12.20)$$

For the example in Table 2.22, this becomes

$$|\epsilon| \leq 8.36\text{E-}9$$

Since the associated arithmetic has 15-16 decimal digits, this shows that (2.12.19) is true over a fairly large interval relative to that arithmetic.

This same phenomenon is also true for multivariable problems. For a further discussion, see Gill-Murray-Wright (1981,p.335).