Chapter 1

Numerical Methods for finding Critical Points

Practical optimization problems often require the use of computational methods for their solution. In this chapter we discuss some of these methods. While some of the methods are limited to optimization problems, other algorithms make use of more general root finding methods such as bisection and Newton's method.

Of course root finding algorithms are relevant because if X^* is an interior minimizer for a differentiable function $g : \mathbb{R}^N \to \mathbb{R}$, then

$$\nabla g(X^*) = (\partial f/\partial x_1, \ldots, \partial f/\partial x_N)(X^*) = 0.$$

We begin by discussing convexity and root finding for functions of one variable. We then discuss the steepest descent approach to finding minima for functions of several variables. Finally, we return to Newton's method in the context of several variables.

1.1 Functions of one variable

Without severe restrictions on the nature of a function (polynomial, convex, etc.), it is difficult to provide any guarantees of correctness for algorithms designed to find minimizers. One severe restriction is that $f''(x) > 0$ for all xı$\mathbb{R}$. Functions satisfying this condition are said to be convex. (The notion of convexity is a bit more general.) As we've seen earlier, this condition guarantees that any critical point is the unique global minimum.

1.1.1 Bisection

Based on our earlier results, our goal is to find the roots of $f(x) = g'(x)$, where $g(x)$ is the function we would like to minimize. For functions of one variable a very effective root finding technique is provided by bisection.

Suppose $f : I \to \mathbb{R}$ is a function defined on the interval I, that f is continuous on I, and there are points $a, b \in I$ such that $a < b$ and $f(a)f(b) < 0$. This last condition simply means that f is positive at one of the two points, and negative at the other point. By the intermediate value theorem there must be a root in the interval $[a, b]$. There is no loss of generality in assuming that $f(a) < 0$ and $f(b) > 0$ since if the signs are switched we can simply replace f by the function $g = -f$. The functions f and g have the same roots.

We will now define two sequences of points $\{a_n\}$, $\{b_n\}$, starting with $a_0 = a$ and $b_0 = b$. The definition of the subsequent points in the sequence is given recursively. The points a_n and b_n will always be chosen so that $f(a_n)f(b_n) \le 0$, and if $f(a_n)f(b_n) = 0$ then either a_n or b_n is a root.

Let c_n be the midpoint of the interval $[a_n, b_n]$, or

$$c_n = \frac{a_n + b_n}{2}.$$

If $f(c_n) = 0$, we have a root and can stop. If $f(c_n) < 0$, define $a_{n+1} = c_n$, and $b_{n+1} = b_n$. If $f(c_n) > 0$, define $a_{n+1} = a_n$, and $b_{n+1} = c_n$. Since $f(a_{n+1}) < 0$ and $f(b_{n+1}) > 0$, the intermediate value theorem implies that a root lies in the interval $[a_{n+1}, b_{n+1}]$.

Finally, notice that

$$|b_{n+1} - a_{n+1}| = \frac{1}{2}|b_n - a_n|.$$

By induction this means that

$$|b_n - a_n| = 2^{-n}|b - a|.$$

The intervals $[a_n, b_n]$ are nested, and the lengths $|b_n - a_n|$ have limit 0, so by the nested interval principle there is a number r such that

$$\lim_{n\to\infty} a_n = r = \lim_{n\to\infty} b_n.$$

Since $f(a_n) < 0$,

$$f(r) = \lim_{n\to\infty} f(a_n) \le 0.$$

On the other hand, $f(b_n) > 0$, so

$$f(r) = \lim_{n\to\infty} f(b_n) \geq 0,$$

and $f(r) = 0$. Moreover

$$|b_n - r| \leq |b_n - a_n| = 2^{-n}|b - a|,$$

so an accurate estimate of the root is obtained rapidly.

1.1.2 Newton's Method

Bisection is an efficient method, but there are faster methods which also generalize well to functions of several variables. The prototype is Newton's method, an old, popular, and powerful technique for obtaining numerical solutions of root finding problems $f(x) = 0$. Geometrically, the idea is to begin with an initial guess x_0. One then approximates f by the tangent line to the graph at x_0. If the slope $f'(x_0)$ is not 0, the point x_1 where the tangent line intercepts the x-axis is taken as the next estimate x_1, and the process is repeated. Thus the algorithm starts with the initial guess x_0 and defines a sequence of points

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}.$$

To analyze the performance of Newton's method one may begin with Taylor's Theorem Lite:

$$f(x) = f(x_n) + (x - x_n)f'(x_n) + \frac{(x - x_n)^2}{2} f''(\xi), \tag{1.1.1}$$

with ξ between x and x_n. Suppose the root we seek is at $x = r$, so that $f(r) = 0$. Solving for r in (1.1.1) gives

$$r = x_n - \frac{f(x_n)}{f'(x_n)} - \frac{(r - x_n)^2}{2} \frac{f''(\xi_n)}{f'(x_n)},$$

or

$$r - x_{n+1} = -(r - x_n)^2 \frac{f''(\xi_n)}{2f'(x_n)}. \tag{1.1.2}$$

Here is a theorem [?] describing the performance of Newton's method.

Theorem 1.1.1. *Suppose that $f : (a, b) \to \mathbb{R}$ has two continuous derivatives. Assume that $f(r) = 0$, but $f'(r) \neq 0$, for some $r \in (a, b)$. If x_0 is chosen sufficiently close to r, the sequence*

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

will converge to r.

Moreover, either $r = x_n$ for some n, or

$$\lim_{n\to\infty} \frac{r - x_{n+1}}{(r - x_n)^2} = -\frac{f''(r)}{2f'(r)}.$$

Proof. By picking a sufficiently small $\epsilon > 0$ we have

$$f'(x) \neq 0, \quad x \in I = [r - \epsilon, r + \epsilon].$$

Let

$$M = \frac{\max_{x\in I} |f''(x)|}{2 \min_{x\in I} |f'(x)|}.$$

Without loss of generality we may assume $M > 0$.

From (1.1.2)

$$|r - x_1| \leq M|r - x_0|^2,$$

and

$$M|r - x_1| \leq (M|r - x_0|)^2.$$

Pick $|r-x_0| \leq \epsilon$ and $M|r-x_0| < 1$. Then $M|r-x_1| < 1$ and $M|r-x_1| \leq M|r - x_0|$, which gives $|r - x_1| \leq \epsilon$. Applying the same idea repeatedly gives $|r - x_n| \leq \epsilon$ and $M|r - x_n| < 1$ for $n = 1, 2, 3, \ldots$.

Now (1.1.2) gives

$$|r - x_{n+1}| \leq M|r - x_n|^2,$$

$$M|r - x_{n+1}| \leq (M|r - x_n|)^2,$$

so

$$M|r - x_n| \leq (M|r - x_0|)^{2^n},$$

or

$$|r - x_n| \leq \frac{1}{M}(M|r - x_0|)^{2^n}.$$

Since $M|r - x_0| < 1$, $\lim_{n\to\infty} x_n = r$.

In (1.1.2) we also find $\xi_n \to r$. Unless $r = x_n$ for some n, we get

$$\lim_{n\to\infty} \frac{r - x_{n+1}}{(r - x_n)^2} = -\frac{f''(\xi_n)}{2f'(x_n)} = -\frac{f''(r)}{2f'(r)}.$$

□

Notice that the estimate for the difference between x_n and the root r has the form

$$|r - x_n| \le \frac{1}{M}(M|r - x_0|)^{2^n}$$

for Newton's method, while for the bisection method the corresponding estimate is

$$|r - x_n| \le [b - a]\frac{1}{2^n}.$$

So if we made

$$M|r - x_0| < 1/2$$

the Newton's method accuracy after n steps would look like the bisection accuracy after 2^n steps.

Here is another perspective on Newton's method. Suppose $g(x)$ is a convex quadratic polynomial

$$g(x) = ax^2 + bx + c, \quad a > 0.$$

Then g has a unique minimizer at $x = \frac{-b}{2a}$, which is the root of

$$f(x) = g'(x) = 2ax + b.$$

If we apply Newton's method to f with any initial point x_0 we have

$$x_1 = x_0 - \frac{g'(x_0)}{g''(x_0)}$$

$$= x_0 - \frac{2ax_0 + b}{2a} = -\frac{b}{2a}.$$

Thus we could say that Newton's method determines the quadratic polynomial $g(x)$ (except for the value of c, which does not effect the position of the minimizer) from the values of $f(x_0) = g'(x_0)$ and $f'(x_0) = g''(x_0)$. The method then estimates the root position to be the location of the minimizer of the quadratic polynomial.

1.1.3 Lagrange interpolation

Following up on the interpretation of Newton's method as an approach that acts as if the function we're trying to minimize is a quadratic polynomial, here is a brief introduction to Lagrange interpolation. In a practical optimization problem it is quite likely that there is no convenient formula for the function we want to minimize. For instance, the function values may be the result of a computer simulation, or they may come from an analysis of functions such as the following.

Find the minimum value of

$$\int_0^1 (g'(x))^2 + \sin^2(x) g^2(x)\ dx,$$

where g is allowed to be any continuously differentiable function.

Suppose we're given $N+1$ values of a function $f(x_0), f(x_1), \ldots, f(x_N)$ at distinct points $x_n \in \mathbb{R}$. Lagrange interpolation finds the unique polynomial $p(x)$ of degree at most N which satisfies

$$p(x_n) = f(x_n), \quad n = 0, \ldots, N.$$

The idea is to first find polynomials $p_n(x)$ satisfying $p_n(x_m) = 1$, if $m = n$ and $p_n(x_m) = 0$ if $m \neq n$.

Start with

$$q_n(x) = \frac{\prod_{m=0}^{N}(x - x_m)}{x - x_n} = \prod_{m \neq n} (x - x_m).$$

The polynomial $q_n(x)$ has degree N. Fix the value at x_n with

$$p_n(x) = \frac{q_n(x)}{q_n(x_n)}.$$

Now simply define

$$p(x) = \sum_{n=0}^{N} f(x_n) p_n(x).$$

For optimization the most obvious case is $N = 2$. Given values y_0, y_1, y_2,

$$p(x) = \frac{y_0}{(x_0 - x_1)(x_0 - x_2)}(x - x_1)(x - x_2) + \frac{y_1}{(x_1 - x_0)(x_1 - x_2)}(x - x_0)(x - x_2)$$
$$+ \frac{y_2}{(x_2 - x_0)(x_2 - x_1)}(x - x_0)(x - x_1).$$

1.2 Descent Methods

We return to the basic problem of minimizing a continuously differentiable function $g : \mathbb{R}^N \to \mathbb{R}$. A simple but effective idea is to 'walk downhill'. That is, starting with some initial point X_0, one produces a sequence of points X_n such that $g(X_{n+1}) \le g(X_n)$. For general functions such ideas are rarely effective, but the story is happier for convex functions (here defined as $Hg(X)$ is positive definite for all $X \in \mathbb{R}^N$) .

1.2.1 Basic algorithms

The first method, called the nonlinear Gauss-Siedel algorithm, reduces the value of g one coordinate at a time. To have an explicit model problem in mind, consider the function

$$g(x, y) = x^2 + 2xy + 2y^2 = (x + y)^2 + y^2.$$

By inspection the minimum is at $(x^*, y^*) = (0, 0)$.

Let E_n denote the n-th standard basis vector. Given an initial point X_0, the algorithm produces a descent sequence via a double loop structure. The elements of the sequence are denoted $X(i, n)$.

$X(1, 0) = X_0$.
for $i = 1$ **to** ∞ **do**
for $n = 1$ **to** N **do**
  Find t^* **minimizing** $g(X(i, n - 1) + tE_n)$.
  Define $X(i, n) = X(i, n - 1) + t^* E_n$.
end do
$X(i + 1, 0) = X(i, N)$
end do

Suppose $g(x, y) = x^2 + 2xy + 2y^2$, and $X_0 = (1, 1)$. The algorithm essentially says to freeze all variables except the first. As an equivalent version of the algorithm we could first minimize

$$x^2 + 2x + 2,$$

finding $x = -1$ and $X(1, 1) = (-1, 1)$. Next we minimize

$$1 - 2y + 2y^2,$$

finding $y = 1/2$ and $X(1,2) = (-1, 1/2)$. We then return to the first variable, minimizing

$$x^2 + x + 1/2,$$

finding $x = -1/2$ and $X(2,1) = (-1/2, 1/2)$. Next we minimize

$$1/4 - y + 2y^2,$$

finding $y = 1/4$ and $X(2,2) = (-1/2, 1/4)$. The computation continues in this fashion.

The second method, called steepest descent, walks downhill in the most promising direction. The following calculation is usually encountered in calculus.

Let U be any unit vector and consider the directional derivative of the function $\phi(t) = g(X + tU)$,

$$\frac{d}{dt}\phi = \frac{d}{dt} g(X + tU) = \nabla g(X + tU) \bullet U.$$

The derivative at $t = 0$ is

$$\frac{d}{dt} g(X + tU)\Big|_{t=0} = \nabla g(X) \bullet U.$$

By the Cauchy-Schwarz inequality the minimum value for this derivative is achieved if

$$U = -\nabla g(X)/\|\nabla g(X)\|.$$

That is, the direction of minus the gradient is the direction in which the function decreases most rapidly, at least for small variations.

Given X_0, this calculation leads to the steepest descent algorithm.

for $i = 0$ **to** ∞ **do**

Find t^* **minimizing** $g(X_i + t\nabla g(X_i))$**.**

Define $X_{i+1} = X_i + t^* \nabla g(X_i)$**.**

end do

Here are the first few computations for the example $g(x,y) = 4x^2 - 4xy + 2y^2 = 2x^2 + 2(x-y)^2$, with $X_0 = (2,3)$. The gradient is

$$\nabla g = (\frac{\partial g}{\partial x}, \frac{\partial g}{\partial y}) = (8x - 4y, -4x + 4y).$$

Since $X_0 = (2,3)$,

$$\nabla g(X_0) = (4,4),$$

and the initial problem is to minimize

$$g((2,3) + t(4,4)) = 32t^2 + 32t + 10.$$

The minimum occurs at $t = -1/2$, so

$$X_1 = (2,3) - \frac{1}{2}(4,4) = (0,1).$$

For the next step use

$$\nabla g(X_1) = (-4,4).$$

The problem now is to minimize

$$g((0,1) + t(-4,4)) = 160t^2 + 32t + 2.$$

The minimum occurs at $t = -1/10$, so

$$X_2 = (0,1) - \frac{1}{10}(-4,4) = (0.4, 0.6).$$

1.2.2 Convergence theorem

To a considerable extent the Gauss-Siedel and steepest descent methods may be analyzed in common. For notational convenience, assume that both methods start with the initial point X_0, and produce a *descent sequence* X_n. Start with an initial observation.

Lemma 1.2.1. *Suppose X_{i+1} is obtained from X_i by minimizing g along a line $X_i + tU$. If $\nabla g(X_i) \bullet U \neq 0$ then $g(X_{i+1}) < g(X_i)$.*

Proof. We are minimizing the function

$$\phi(t) = g(X_i + tU).$$

The derivative is

$$\frac{d\phi}{dt} = \nabla g(X_i + tU) \bullet U,$$

and its value at $t = 0$ is

$$\frac{d\phi}{dt}(0) = \nabla g(X_i) \bullet U.$$

If $\frac{d\phi}{dt}(0) < 0$ then $\phi(t) < \phi(0)$ for a sufficiently small $t > 0$, with the obvious change if $\frac{d\phi}{dt}(0) > 0$. □

Corollary 1.2.2. *Suppose $\nabla g(X_i) \neq 0$. Then for the steepest descent algorithm $g(X_{i+1}) < g(X_i)$, and for the Gauss-Siedel algorithm $g(X_{i+N}) < g(X_i)$.*

Proof. For the steepest descent algorithm we apply Lemma 1.2.1 with $U = \nabla g(X_i)/\|\nabla g(X_i)\|$.

For the Gauss-Siedel method

$$\frac{\partial g}{\partial x_n}(X_i) \neq 0$$

for at least one of the partial derivatives, and for such directions we may again apply Lemma 1.2.1. □

Theorem 1.2.3. *Suppose $f : \mathbb{R}^N \to \mathbb{R}$ is a continuous coercive function with continuous first partial derivatives. If X_0 is any initial point and $\{X_i\}$ is the corresponding steepest descent sequence, then some subsequence $\{X_{i(j)}\}$ of $\{X_i\}$ converges. The limit of any subsequence of the steepest descent sequence is a critical point for f.*

Proof. Since f is coercive, there is an $R > 0$ such that the set $S = \{X \in \mathbb{R}^N \,|f(X) \leq f(X_0)\}$ is contained in the closed ball B_R of radius R centered at $0 \in \mathbb{R}^N$. The set B_R is compact (closed and bounded), and since $f(X_i) \leq f(X_0)$ for all X_i in the descent sequence, each X_i lies in B_R. By compactness, the sequence $\{X_i\}$ has a convergent subsequence $\{\chi_i\}$ with limit Y.

Contrary to the claim of the theorem, suppose that $\nabla f(Y) \neq 0$. Starting at Y, define Y_1 as the next step using steepest descent, $Y_1 = f(Y + t^* \nabla f(Y))$, and notice that by Lemma 1.2.1 we have $f(Y_1) < f(Y)$.

Since the original steepest descent sequence is nonincreasing and χ_i is a subsequence of $\{X_i\}$,

$$f(\chi_{i+1}) \leq f(\chi_i + t^* \nabla f(\chi_i)).$$

Since f and ∇f are continous, we get the following limits:

$$f(Y) = \lim_{i \to \infty} f(\chi_{i+1}) \leq \lim_{i \to \infty} f(\chi_i + t^* \nabla f(\chi_i)) = f(Y_1).$$

The contradictory conclusions $f(Y) \leq f(Y_1)$ and $f(Y_1) < f(Y)$ imply that $\nabla f(Y) = 0$. □

The convergence guarantees are even stronger for strictly convex coercive functions.

Theorem 1.2.4. *Suppose $f : \mathbb{R}^N \to \mathbb{R}$ is a continuous strictly convex coercive function with continuous first partial derivatives. If X_0 is any initial point and $\{X_i\}$ is the corresponding steepest descent sequence, then $\{X_i\}$ converges to the unique global minimizer of f.*

Proof. Since f is continuous and coercive, it has a global minimizer Y, which is unique by strict convexity, and a critical point since f has partial derivatives. Moreover Y is the only critical point for f.

Suppose the sequence $\{X_i\}$ does not converge to Y. Then there is a number $\epsilon > 0$ and a subsequence X_j such that $d(X_j, Y) \geq \epsilon$. By the previous theorem the sequence X_j has a subsequence which converges to a point Y_1, and $\nabla f(Y_1) = 0$. Since $d(Y_1, Y) \geq \epsilon$, we have contradicted the uniqueness of the critical points for strictly convex functions.

□

Theorem 1.2.5. *Theorems Theorem 1.2.3 and Theorem 1.2.4 are valid if $\{X_i\}$ is a Gauss-Siedel sequence for f starting at X_0.*

Proof. In the proof of Theorem 1.2.3 we only have to redefine Y_1 to be

$$Y_1 = f(Y + t^* \frac{\partial f}{\partial x_n}(Y)),$$

where n is chosen so that $\frac{\partial f}{\partial x_n}(Y) \neq 0$. The proof of Theorem 1.2.4 does not require any changes.

□