Note on One-Dimensional Optimization

Makoto Nakajima, UIUC
January 2006

1 Introduction

We study various methods to optimize a function with one variable.

2 The Problem

Problem 1
We want to solve the following minimization problem with a real valued function $f(x)$:

$$\min_{x\in[\underline{x},\overline{x}]} f(x)$$

A small remark:

1. Maximization problem is just putting minus at the top of $f(x)$ function in the minimization problem.

3 Golden Section Search

The method doesn't require a derivative, but need to bracket at first the optimum. After bracketing the optimum, the method keeps shrinking the bracket until the bracket converges to a point, which is the solution. Below is the algorithm:

Algorithm 1 (Golden Section Search)

1. *Set the tolerance parameter ϵ*
2. *Find a bracket $[a,b]$ which contains the optimum.*
3. *If $|a-b|<\epsilon$, stop. Solution is $\frac{a+b}{2}$*
4. *Compute $f(a)$ and $f(b)$.*
5. *Construct two more points:*

$$c = a + \frac{3-\sqrt{5}}{2}(b-a)$$

$$d = a + \frac{\sqrt{5}-1}{2}(b-a)$$

6. *Evaluate $f(c)$ and $f(d)$.*

7. *If $f(c) \geq f(d)$, then the minimum is in $[c, b]$. Let $a = c$ and go back to step 3. Notice that, in the next step, the current d is exactly c in the next step. So we can reduce one evaluation of $f(x)$.*

8. *If $f(c) < f(d)$, then the minimum is in $[a, d]$. Let $b = d$ and go back to step 3. Notice that, in the next step, the current c is exactly d in the next step. So we can reduce one evaluation of $f(x)$.*

Some comments:

1. Does not require derivatives.
2. Not Slower for a nice shaped $f(x)$.
3. Danger of being stuck at a local optimum.
4. Cannot extend to a multi-dimensional problem.

4 Inverse Parabolic Method

It's easy to compute the optimum of a quadratic function. This method is basically approximate the original $f(x)$ function by a quadratic function and the approximation gives immediately the location of an optimum.

Algorithm 2 (Inverse Parabolic Method)

1. *Set a tolerance parameter ϵ*
2. *Pick three points a, b, c from $[\underline{x}, \overline{x}]$.*
3. *Interpolate the three points using a quadratic function.*
4. *Compute analytically the optimum of the quadratic function and denote the point d.*
5. *Pick $\max\{a, b, c\}$ and replace it with d.*
6. *If $\max\{|a - b|, |b - c|, |a - c|\} < \epsilon$, stop. Otherwise, go back to step 3.*

Some comments:

1. Does not require derivatives.
2. Efficient for a nice shaped $f(x)$.
3. Danger of being stuck at a local optimum.
4. In the Numerical Recipes, Brent's method is introduced. Brent's method is a robust version of the inverse parabolic method. As long as the inverse parabolic method gives an acceptable new guess, the algorithm keeps using the result from the inverse parabolic method. If the inverse parabolic method gives a new guess which is not plausible, the algorithm instead uses golden section search. This is robust but efficient optimization algorithm which doesn't require derivatives.

5 Newton's Method

Basic idea is to use second order Taylor approximation to approximate the function $f(x)$ and use it to find a optimum. The method requires first and second derivatives. More specifically:

Algorithm 3 (Newton's Method)

1. *Set a tolerance parameter* ϵ
2. *Set a guess* x_0
3. *Compute first and second derivatives of* $f(x)$ *at* x_0*. If* $f'(x_0) < \epsilon$*, done.*
4. *Apply second order Taylor series expansion around* x_0 *and obtain:*

$$\tilde{f}(x) = f(x_0) + f'(x_0)(x - x_0) + \frac{1}{2}f''(x_0)(x - x_0)^2$$

   Compute numerical derivatives if not available.

5. *Solve for* $\tilde{f}'(x_1) = 0$*. Notice that we only need to solve:*

$$x_1 = x_0 - [f''(x_0)]^{-1} f'(x_0)$$

6. *Update* $x_0 = x_1$ *and go back to step 3.*

Comments below:

1. As long as the initial guess x_0 is close enough to the optimum, it is proved that iterations quadratically converges to the optimum.
2. But how to get close from the beginning?
3. Need derivatives.
4. Straightforward to extend to multidimensional case.

6 Quasi-Newton Method

Computing second derivatives is a costly operation. Not so much for one-dimensional case, but more so for multidimensional case. Therefore, lots of methods which approximate the second derivative are devised. These are called Quasi-Newton Method.

One example is below:

Algorithm 4 (Simple Quasi-Newton Method)

1. *Set a tolerance parameter* ϵ
2. *Set a guess* x_0
3. *Compute first derivative of* $f(x)$ *at* x_0*. If* $f'(x_0) < \epsilon$*, done.*

4. *As in the case of Newton's Method, we solve the following to update x:*

$$x_1 = x_0 - [f''(x_0)]^{-1} f'(x_0)$$

But instead of using $f''(x_0)$, we use some constant. A simple choice is 1. *In other words, x_1 is obtained by:*

$$x_1 = x_0 - f'(x_0)$$

5. *Update $x_0 = x_1$ and go back to step 3.*

Comments below:

1. Variants of Quasi-Newton Method are really important for multidimensional optimization, as it becomes costly to compute Hessian matrix, and it becomes difficult to obtain positive definite Hessian matrix.