Newton Directions in Descent Methods for Optimization

University of Colorado Boulder
Department of Applied Mathematics

1 Project Summary

In class, we discussed the steepest descent method applied to problem of finding roots of some $F : \mathbb{R}^n \to \mathbb{R}^m$. After re-writing it in terms of finding the minimum for the quadratic $f : \mathbb{R}^n \to \mathbb{R}$ defined as $f(x) = \sum_{j=1}^{m} F_j(x)^2 = ||F(x)||_2^2$, we derived the method with an iteration of the form $x_{k+1} = x_k - \alpha_k \nabla f(x_k)$. The direction $-\nabla f(x_k)$ at step k is a specific instance of a descent direction (the "steepest") within a more general class of descent directions. We can choose α_k in a variety of ways, but the method can be slow on difficult problems. Another important descent direction is the Newton direction, which has a natural unit step length. In this project, you will derive the Newton direction $p_k = -(\nabla^2 f(x_k))^{-1} \nabla f(x_k)$ and associated method, and numerically verify your implementation on a root finding problem. You will also explore some of the quasi versions of this method which address the inefficiencies with inverting the Hessian matrix. For the independent directions, you may consider variations when the Hessian is not positive definite, and/or apply your implementations to a different optimization problem of interest to you, including data-driven applications.

2 Project Background

We are interested in solving the following unconstrained optimization problem with objective function $f : \mathbb{R}^n \to \mathbb{R}$:

$$\min_x f(x)$$

using a descent method of the form

$$x_{k+1} = x_k + \alpha_k p_k.$$

Here, α controls the step we take in the descent direction p_k. A *descent direction* p_k is any direction for which $f(x_{k+1}) < f(x_k)$. Depending on p_k, one typically wants to determine an optimal step size α_k that will decrease the value at $f(x_{k+1})$ the most. This is the case with line search strategies in the steepest descent method, where $p_k = \nabla f(x_k)$. However, when f is well enough represented by a quadratic model, the Newton direction

$$p_k = -(\nabla^2 f(x_k))^{-1} \nabla f(x_k)$$

can be used to provide quadratic local convergence with a natural step length of $\alpha_k = 1$ for every k. The derivation of this method requires only Taylor's theorem for $f : \mathbb{R}^n \to \mathbb{R}$. We will provide it here.

Theorem 1. *Suppose that $f : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable and consider $p \in \mathbb{R}^n$. Then, we have that*

$$f(x + p) = f(x) + \nabla f(x + tp)^T p,$$

for some $t \in (0, 1)$. Moreover, if f is twice continuously differentiable, we have that

$$f(x + tp) = f(x) + \nabla f(x)^T p + \frac{1}{2} p^T \nabla^2 f(x + tp) p,$$

for some $t \in (0, 1)$.

2.1 Derive the Newton direction

The following and steps will help derive the descent direction.

1. Consider the Taylor expansion of $f(x_k + \alpha p)$. What is the rate of change in f along the direction p at x_k.

2. Use your answer to 1 to find the unit direction p of most rapid decrease. *Hint: minimize the answer from 1 over p. You should recover the steepest descent direction.*

3. Now, consider a quadratic model $m_k(p)$ of $f(x_k+p)$. Assuming that $\nabla^2 f(x_k)$ is positive definite, find the direction p which minimizes $m_k(p)$. This is the Newton direction.

4. When is the Newton direction reliable?

5. Why did we assume $\nabla^2 f(x_k)$ is positive definite? *Hint: Think about the descent property* $\nabla f(x_k)^T p_k < 0$.

2.2 Implement and test your method

Now, implement a descent method with Newton direction and apply it to the problem of finding roots. You can consider any function $F : \mathbb{R}^n \to \mathbb{R}^m$ for which you want to find roots, as long as n is large enough (say, ≥ 3).

1. Provide a descent algorithm which uses the Newton direction.

2. Analytically show the order of convergence when near the minimizer.

3. Implement your algorithm and empirically verify the observed order of convergence.

2.3 Variations of Newton descent

We know that computing and inverting the Hessian matrix $\nabla^2 f(x_k)$ which appears in the Newton direction can be expensive for large n, and the expense is compounded if we require many iterations for convergence. In class and lab, we encountered a similar issue with the Jacobian in Newton's method, and considered variants such as "lazy Newton" and rank-one updates to the Jacobian (Broyden's method). We have analogs for Newton descent involving re-use or updates to the Hessian.

1. Implement a lazy version of Newton descent which re-uses the Hessian inverse computed at the first step. How does it perform?

2. Implement the Broyden-Fletcher-Goldfarb-Shannon rank-one update (BFGS) formula and compare performance. There are many online resources describing the update, and it can also be found in the Nocedal-Wright book.

3 Software expectations

You may use software for the independent portion of this project, but must implement your own routines for the exercises. You may use built-in functions in `MATLAB` or `Python` (for computing norms, inverses, etc.). A convex optimization toolbox that may be of interest for the independent exploration are `cvx`[1] and `cvxpy`[2], which are the `MATLAB` and `Python` versions respectively. If using external tools, make sure the routines you use are analogous to the methods considered in the project.

4 Independent Directions

Possible extensions to this project include, but are not limited to:

1. Explore variations that tackle the case of an invertible, but not positive definite Hessian.
2. Solve an optimization problem of your choosing with the methods you implemented, or external software. You may use a toy construction for the objective, or a data-driven problem. When you don't have the functional form of the objective, but only input/output data, you can use finite differences to approximate the needed derivatives. This will be helpful if you choose to try a data-driven application.

5 Helpful Sources

1. Nocedal, Wright, Numerical Optimization, Chapters 2, 3.
2. Martinez, Practical Quasi-Newton methods for solving nonlinear systems,
3. Dembo, Eisenstat, Steinhaug, Inexact Newton Methods, SIAM Journal on Numerical Analysis,

[1] `http://cvxr.com/cvx/`
[2] `https://www.cvxpy.org/`