1 Overview

In the previous lecture we introduced the gradient descent algorithm, and mentioned that it falls under a broader category of methods. In this lecture we describe this general approach called *steepest descent*. We will explain how gradient descent is an example of this method, and also introduce the *coordinate descent* algorithm which is another example of the steepest descent method. Lastly, we will present Newton's method. Newton's method is a general approach for solving systems of non-linear equations. Newton's method can conceptually be seen as a steepest descent method, and we will show how it can be applied for convex optimization.

2 Steepest Descent

As discussed in the previous lecture, one can consider a search for a stationary point as an iterative procedure of generating a point $\mathbf{x}^{(k+1)}$ which takes steps of certain length t_k at direction $\Delta\mathbf{x}^{(k)}$ from the previous point $\mathbf{x}^{(k)}$. The direction $\Delta\mathbf{x}^{(k)}$ decides which direction we search next, and the step size determines how far we go in that particular direction. We can write this update rule as:

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + t_k\Delta\mathbf{x}^{(k)}$$

A *steepest descent* algorithm would be an algorithm which follows the above update rule, where at each iteration, the direction $\Delta\mathbf{x}^{(k)}$ is the *steepest* direction we can take. That is, the algorithm continues its search in the direction which will minimize the value of function, given the current point. Or in other words, given a particular point $\mathbf{x}$, we would like to find the direction $\mathbf{d}$ s.t. $f(\mathbf{x}+\mathbf{d})$ is minimized.

Finding the steepest direction. In order to find the steepest direction, we can approximate the function via a first-order Taylor expansion:

$$f(\mathbf{x}+\mathbf{d}) \approx f(\mathbf{x}) + \nabla f(\mathbf{x})^\intercal\mathbf{d}$$

The direction $\mathbf{d}$ that minimizes the function implies the following optimization problem[1]:

$$\min_{\mathbf{d}:\|v\|=1} \nabla f(\mathbf{x})^\intercal\mathbf{d}$$

In general, one may consider various norms for the minimization problem. As we will now see, the interpretation of steepest descent with different norms leads to different algorithms.

[1]Recall that a *direction* is a vector of unit length.

2.1 Steepest descent in ℓ_2: gradient descent

In the case of the ℓ_2 norm, let us first find $\mathbf{d}^\star \in \arg\max_{\|d\|_2=1} \nabla f(\mathbf{x})^\intercal \mathbf{d}$. From the Cauchy-Schwarz inequality we know that $\nabla f(\mathbf{x})^\intercal \mathbf{d} \leq \|\nabla f(\mathbf{x})\| \|\mathbf{d}\|$ with equality when $\mathbf{d} = \lambda \nabla f(x),\ \lambda \in \mathbb{R}$. Since $\|\mathbf{d}\| = 1$ this implies:

$$\mathbf{d}^\star = \frac{\nabla f(\mathbf{x})}{\|\nabla f(\mathbf{x})\|}$$

Since we aim to *minimize* the function, we seek $\hat{\mathbf{d}} \in \text{argmin}_{\|\mathbf{d}\|_2=1} \nabla f(\mathbf{x})^\intercal \mathbf{d}$ which is simply $-\mathbf{d}^\star$. So the iterations of steepest descent in ℓ_2 norm are:

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

where t_k is some step size (think about this as some term multiplied by $\|\nabla f(\mathbf{x})\|^{-1}$). Notice that this is the gradient descent algorithm.

2.2 Steepest descent in ℓ_1: coordinate descent

In the case of the ℓ_1 norm, the steepest direction is:

$$\mathbf{d} \in \text{argmin}_{\mathbf{d}:\sum_i |d_i|=1} \nabla f(\mathbf{x}) \intercal \mathbf{d}$$

This requires solving a linear program over $\mathbf{d}$ where $\nabla f(\mathbf{x})$ is a fixed vector:

$$\begin{aligned} &\min \nabla f(\mathbf{x})^\intercal \mathbf{d} \\ &\text{s.t.} \sum_{i=1}^{n} |d_i| = 1 \end{aligned}$$

We discussed these special cases in lecture 5 when we proved the minimax theorem: in this case the optimal solution is to find the index $i \in [n]$ for which the absolute value $\partial f(\mathbf{x})/\partial x_i$ is largest, and the optimal solution is then $\texttt{sign}(e_i)$, where $\texttt{sign}$ attributes a negative sign to e_i if $\partial f(\mathbf{x})/\partial x_i$ is positive, and a negative sign otherwise, and e_i is the standard basis vector with 1 in index i and 0s in all other indices). So the steepest direction is the direction $\mathbf{d}$ which respects:

$$\mathbf{d} = \texttt{sign}(e_i)\ , \quad i \in \text{argmax}_{i\in[n]} \left| \frac{\partial f(\mathbf{x})}{\partial x_i} \right|$$

Algorithm 1 Coordinate Descent

1: Guess $\mathbf{x}^{(0)}$, set $k \leftarrow 0$
2: **while** $||\nabla f(\mathbf{x}^{(k)})|| \geq \epsilon$ **do**
3: $\quad i \leftarrow \text{argmax}_{i\in[n]} \left| \frac{\partial f(\mathbf{x})}{\partial x_i} \right|$
4: $\quad \mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + t_k \cdot \texttt{sign}(e_i)$
5: $\quad k \leftarrow k + 1$
6: **end while**
7: **return** $\mathbf{x}^{(k)}$

This gives us a very simple algorithm called *coordinate descent*: we start with an arbitrary point, and at every stage we iterate over all the coordinates of $\nabla f(\mathbf{x})$, find the one with largest absolute value, and descent in that direction.

3 Newton's Method

The goal of Newton's method is to find solutions $\mathbf{x}$ for which $h(\mathbf{x}) = 0$, for some function $h : \mathbb{R}^n \to \mathbb{R}^n$. The main idea is to start with an arbitrary point, and iteratively try to find the point at which the function evaluates to zero by find the point of the tangent line of the function at the current point evaluates to zero,

tangent lines of the function and a given point.

Newton's method for single dimensional functions. For simplicity let's start with functions $h : \mathbb{R} \to \mathbb{R}$. The tangent line of a function h at point $x^{(k)}$ can be expressed as the following equation:

$$h(x^{(k)}) + h'(x^{(k)})(x - x^{(k)})$$

The point x at which this line evaluates to zero is:

$$h(x^{(k)}) + h'(x^{(k)})(x - x^{(k)}) = 0 \iff x = x^{(k)} - \frac{h(x^{(k)})}{h'(x^{(k)})}$$

At every step k, Newton's method finds sets the next guess to be $x^{(k+1)} = x^{(k)} - h(x^{(k)})/h'(x^{(k)})$, until finding a point close enough to zero. We formally describe the algorithm below.

Algorithm 2 Newton's method

1: Guess $x^{(0)}$, set $k \leftarrow 0$
2: **while** $||h(x^{(k)})|| \geq \epsilon$ **do**
3: $\quad x^{(k+1)} \leftarrow x^{(k)} - \frac{h(x^{(k)})}{h'(x^{(k)})}$
4: $\quad k \leftarrow k + 1$
5: **end while**
6: **return** $x^{(k)}$

Newton's method for multidimensional functions. More generally, Newton's method can be applied for multidimensional functions $g : \mathbb{R}^n \to \mathbb{R}^n$, where $g(\mathbf{x}) = (g_1(\mathbf{x}), g_2(\mathbf{x}), \ldots, g_n(\mathbf{x}))$. In this case, Newton iterates satisfy:

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - g(\mathbf{x}^{(k)}) \nabla g(\mathbf{x}^{(k)})^{-1}$$

Where here $\nabla g(\mathbf{x})$ is the Jacobian matrix of g at $\mathbf{x}$.

Definition. *For a function $g : \mathbb{R}^n \to \mathbb{R}^n$, the **Jacobian matrix** of g at point $\mathbf{x} \in \mathbb{R}^n$ is a matrix M for which $M_{ij} = \frac{\partial g_i(\boldsymbol{x})}{\partial x_j}$.*

3.1 Applying Newton's method to minimize a convex function

In our case, we are interested in minimizing a convex function $f : \mathbb{R}^n \to \mathbb{R}$. In the previous lecture we showed that if the function is convex then a stationary point is the minimizer. We there would

like to find the solution $\mathbf{x}$ for which $\nabla f(\mathbf{x}) = 0$. Note that the Jacobian of ∇f is in fact the Hessian, and we therefore have that for $\nabla f(\mathbf{x})$ Newton's method iterates are:

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

We get the following method for minimizing a convex function.

Algorithm 3 Newton's method for minimizing a convex function

1: Guess $\mathbf{x}^0$, set $k = 1$
2: **while** $||\nabla f(\mathbf{x}^{(k)})|| \geq \epsilon$ **do**
3: $\quad \mathbf{x}^{(k+1)} \leftarrow \mathbf{x}^{(k)} - \nabla f(\mathbf{x}^{(k)}) H_f(\mathbf{x}^{(k)})^{-1}$
4: $\quad k \leftarrow k + 1$
5: **end while**
6: **return** $x^{(k)}$

Example. Consider the problem of minimizing $x^4 + 2x^2y^2 + y^4$ using Newton's method. Let's first write the gradient and the Hessian:

$$\nabla f(x, y) = \begin{pmatrix} \frac{\partial f(x,y)}{\partial x} \\ \frac{\partial f(x,y)}{\partial y} \end{pmatrix} = \begin{pmatrix} 4x^3 + 4xy^2 \\ 4x^2y + 4y^3 \end{pmatrix}$$

$$H_f(x, y) = \begin{pmatrix} 12x^2 + 4y^2 & 8xy \\ 8xy & 4x^2 + 12y^2 \end{pmatrix}$$

Let's assume we start from a guess $(x, y) = (a, a)$ for some $a \in \mathbb{R}$. An equivalent way to write our condition for the next iteration is:

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

Thus the next point $\mathbf{x}^{(1)} = (x, y)$ is the one where the values are x, y respect:

$$\begin{pmatrix} -8a^3 \\ -8a^3 \end{pmatrix} = \begin{pmatrix} 16a^2 & 8a^2 \\ 8a^2 & 16a^2 \end{pmatrix} \begin{pmatrix} x - a \\ y - a \end{pmatrix}$$

One can check that this system is solved by $(x, y) = 2/3(a, a)$. In general the kth iteration will be:

$$\left(\frac{2}{3}\right)^k \cdot \begin{pmatrix} a \\ a \end{pmatrix}$$

which converges to zero exponentially fast in k.

3.2 Convergence of Newton's method

It is important to note that Newton's method will not always converge. For the (non-convex) function $f(x) = x^3 - 2x + 2$ for example, if we start at $x^{(0)} = 0$, the points will oscillate between 0 and 1 and will not converge to the root. That is, for all $t \in \mathbb{N}$, $x^{(k)} = 1$ for all $k = 2t + 1$ and

$x^{(k)} = 0$ for all $k = 2t$. For the function $f(x) = \frac{2}{3} \cdot |x|^{3/2}$ one can check that Newton's method will oscillate as well. So when can we at least guarantee that Newton's method does not oscillate?

Definition. *A vector $\in \mathbb{R}^n$ is called a* ***descending direction*** *for $f : \mathbb{R}^n \to \mathbb{R}$ if for all $\boldsymbol{x} \in \mathbb{R}^n$ we have that $\nabla f(\boldsymbol{x})^T \boldsymbol{d} < 0$.*

Proposition 1. *For a given function f, if its Hessian H_f is positive definite then $\boldsymbol{d} = -H_f(\boldsymbol{x}^k)^{-1} \nabla f(\boldsymbol{x}^k)$ is a descent direction. That is, Newton's method decreases the function value at every iteration.*

Proof. Since H_f is positive definite this implies that its inverse is also positive definite. Thus:

$$\nabla f(\mathbf{x}^{(k)})^T \mathbf{d} = -\nabla f(\mathbf{x}^{(k)})^T - H_f(\mathbf{x}^{(k)})^{-1} \nabla f(\mathbf{x}^{(k)}) < 0. \qquad \square$$

4 Further Reading

For further reading on steepest descent and Newton's method see Chapter 9 of the Convex Optimization book by Boyd and Vandenberghe.