CS 596: Theoretical Machine Learning

Lecturer: Pranjal Awasthi
Scribe: He Chen

Lecture #15?
November 29, 2016

Definitions:

1. Convex function: f is convex, iff $\forall x, y, \lambda \in [0,1], f(\lambda x+(1-\lambda)y) \leq \lambda f(x)+(1-\lambda)f(y)$.
2. Convex set: set K is convex, iff $\forall x, y, \lambda \in [0,1], \lambda x + (1-\lambda)y \in K$.
3. G-bounded gradient: Function f is G-bounded, if $\|\nabla f(x)\| \leq G$ or $\|g(x)\| \leq G$ (if f is not differentiable, $g(x)$ is the sub-gradient of f).
4. β-Lipschitz: Function f is β-Lipschitz, if $\forall x, y, \|\nabla f(x) - \nabla f(y)\| \leq \beta \|x - y\|$.

SVMs and soft-SVMs are examples of convex optimizations. In SVMs, we need to solve $\min \frac{1}{2}\|w\|^2$, such that $y_i(wx_i) \geq 1$. In soft-SVMs, we need to solve $\min \frac{1}{m}\sum_{i=1}^{m}(1 - y_i(wx_i))_+ + \lambda\|w\|^2$. Another example is matrix factorization, where we need to minimize $f(x) = \|M - XY^T\|_F^2$. Although the function itself is not convex, we iteratively minimize it by solving a convex function in either X or Y and fixing the other variable. Convexity is useful. One property of it is algorithmic stability. It can lead to tractable and fast algorithms. And some general techniques used in convex optimization could also be used in non-convex optimization. To check the convexity of a function, we could use the basic definition. Or if f is twice differentiable, we could compute the Hessian matrix. If the Hessian matrix is semi-positive definite, then f is convex. However, in general, it's NP-hard to check whether a given function is convex.

Basic Facts for convex functions:

- Local minimum is global minimum.
- If function f is also differentiable, then $\forall x, y, f(y) \geq f(x) + \nabla f(x)(y - x)$.
- $\forall x, \exists g_x \in \mathbb{R}^d$, such that $f(y) \geq f(x) + g_x \cdot (y - x)$, g_x is called the sub-gradient of f at x.

An example for the third fact is $f = \min \|w\|_1$. This function is 1-bounded. In this case, $\nabla f(x)_{x>0} = 1, \nabla f(x)_{x<0} = -1$. Based on $f(y) \geq f(0) + g(0)y$, we have $|y| \geq g(0)y$, which tells us $g(0) = \{+1, 0, -1\}$.

Algorithms for minimizing convex function:

We are assuming the algorithms have access to gradient for free. The basic approach is to start from an initial guess, compute the next point and so on so forth. The algorithms will generate a sequence of points:

$$x_0 \to x_1 \to x_2 \to \cdots$$

The goal is to output $x' \in K \subseteq \mathbb{R}^d$, such that

$$f(x') \leq \min_{x \in K} f(x) + \epsilon.$$

There are two general methods: interior point method and 1^{st} order methods. Suppose T is the number of steps. The former method gives $T = O(d^3 \log \frac{1}{\epsilon})$, while the later could provide $T \approx \frac{1}{\epsilon}, \frac{1}{\epsilon^2}$, or $\sqrt{\frac{1}{\epsilon}}$. Gradient descent (GD) is one kind of 1^{st} order method. It works as follows. At first, we choose the starting point $x_0 = 0$. For $t = 1, 2, \cdots$, let $x_{t+1} = x_t - \eta \cdot \nabla f(x_t)$ if function f is differentiable. Otherwise, let $x_{t+1} = x_t - \eta \cdot g(x_t)$. The guarantees for GD are determined by the properties of function f. Formally, we have theorems below.

Theorem 1. *If f is convex and G-bouned, suppose $x^* = \min_x f(x)$, $\|x^*\| \leq R$, then after T steps of gradient descent, with $\eta = \frac{R}{G\sqrt{T}}$, we have*

$$\frac{1}{T}\sum_{t=1}^{T} f(x_t) - f(x^*) \leq \frac{RG}{\sqrt{T}}.$$

Theorem 2. *If f is convex and β-Lipschitz, suppose $x^* = \min_x f(x)$, $\|x^*\| \leq R$, then after T steps of gradient descent, with $\eta = \frac{1}{\beta}$, we have*

$$f(x_T) - f(x^*) \leq \frac{\beta R^2}{T}.$$

It's clear β-Lipschitz functions have better bound based on the relationships with respect to T. Moreover, Theorem 1 only bounds the average while Theorem 2 bounds the last point x_T. The reason is that, G-bounded functions without β-Lipschitz property can not guarantee that the gradient would descrease while approaching the goal. Taking $f(x) = |x|$ for example, the gradient is either +1, 0, or -1. When it's near the goal, it may oscillate and never reach the goal. The solution is to decrease η as we go along, as shown in the proof of Theorem 1.

Proof of Theorem 1. Assume there is no constraint, which means $K = \mathbb{R}^d$. We have $x_{t+1} = x_t - \eta \cdot g(x_t)$. Also by convexity of f, $f(y) \geq f(x) + g_x \cdot (y - x)$. Apply the previous equation with $y = x^*, x = x_t$,

$$\begin{aligned}
f(x^*) &\geq f(x_t) + g(x_t) \cdot (x^* - x_t) \\
\Rightarrow f(x_t) - f(x^*) &\leq g(x_t) \cdot (x_t - x^*) \\
&= \frac{1}{\eta}(x_t - x_{t+1})(x_t - x^*) \\
&= \frac{1}{2\eta}(\|x_t - x^*\|^2 - \|x_{t+1} - x^*\|^2 + \|x_t - x_{t+1}\|^2) \\
&= \frac{1}{2\eta}(\|x_t - x^*\|^2 - \|x_{t+1} - x^*\|^2) + \frac{1}{2\eta}\eta^2\|g(x_t)\|^2 \\
&\leq \frac{1}{2\eta}(\|x_t - x*\|^2 - \|x_{t+1} - x^*\|^2) + \frac{\eta G^2}{2}
\end{aligned}$$

$$\begin{aligned}\Rightarrow \frac{1}{T}\sum_{t=0}^{T} f(x_t) - f(x^*) &\le \frac{1}{2\eta T}\sum_{t=0}^{T}(\|x_t - x^*\|^2 - \|x_{t+1} - x^*\|^2) + \frac{\eta G^2}{2} \\ &= \frac{1}{2\eta T}(\|x_0 - x^*\|^2 - \|x_{T+1} - x^*\|^2) + \frac{\eta G^2}{2} \\ &\le \frac{1}{2\eta T}\|x_0 - x^*\|^2 + \frac{\eta G^2}{2} \\ &\le \frac{R^2}{2\eta T} + \frac{\eta G^2}{2} \\ &\le \frac{RG}{\sqrt{T}}, (\eta = \frac{R}{G\sqrt{T}})\end{aligned}$$

□

Some notes:

- A related algorithm called Mirror Descent works if the function is G bounded in a different norm rather than the ℓ_2 norm.
- Instead of using $\eta = \frac{R}{G}\sqrt{\frac{1}{T}}$, we could use $\eta_t = \frac{R}{G}\sqrt{\frac{1}{t}}$ and lose $\log t$ factor in the bound.
- Our original problem is $\min_{x \in K} f(x)$, where $K \subseteq \mathbb{R}^d$. It requires one more projection step compared with normal gradient descent

  $$y_{t+1} = x_t - \eta \cdot g(x_t)$$

  $$x_{t+1} = Proj_K(y_{t+1})$$

  where the projection step is $x_{t+1} = argmin_{x \in K}\|x - y_{t+1}\|^2$. The projection step finds a better point than y_{t+1}, since $\|y_{t+1} - x^*\|^2 \ge \|x_{t+1} - x^*\|^2$. The same analysis as above goes through with the same rate. The only issue is that in some case projection might be an expensive operation. There is an algorithm called the Frank-Wolfe method that runs gradient descent without projections.
- For G-bounded functions, $\frac{RG}{\sqrt{T}}$ is tight.

Additional Readings

- The book by Boyd and Vandenberghe. `http://stanford.edu/~boyd/cvxbook/`.
- Lecture notes by Sébastian Bubeck. `https://blogs.princeton.edu/imabandit/orf523-the-complexities-of-optimization/`.