DS-GA 3001 Modern Topics on Statistical Learning Spring 2023

Lecture 5 — Feb. 24, 2023

Prof. Qi Lei *Scribe: Nikita Anand, Bale Chen, Swapnil Sharma*

Recap

In the first three lectures, we talked about *generalization*, which solves the key question – if the global optimum of the empirical risk is found, does the optimal solution generalize to the population risk function? Specifically, for

$$\mathcal{L}(\theta) : \text{population risk} \ , \ \widehat{\mathcal{L}}_n(\theta) : \text{empirical risk}$$

we tried to show uniform convergence that with high probability $1-\delta$,

$$\sup_{\theta\in\Theta} |\widehat{\mathcal{L}}_n(\theta) - \mathcal{L}(\theta)| \leq \epsilon$$

Uniform convergence implies generalization bound, which means the excess risk $R(\hat{\theta})$ is bounded and "can be" small. We have shown that

$$R(\hat{\theta}) = |\widehat{\mathcal{L}}_n(\hat{\theta}) - \mathcal{L}(\theta^*)| \leq 2\epsilon$$

where

$$\hat{\theta} = \arg\min_{\theta\in\Theta} \widehat{\mathcal{L}}_n(\theta)$$

$$\theta^* = \arg\min_{\theta\in\Theta} \mathcal{L}(\theta)$$

Throughout the process, we assumed that $\hat{\theta} \leftarrow \arg\min_{\theta\in\Theta} \widehat{\mathcal{L}}_n(\theta)$, but we didn't discuss how we actually solve for $\hat{\theta}$. Recall that *generalization* together with *optimization* forms the learning theory and this lecture we start to touch upon the field of *optimization*.

In this lecture, we are going to introduce algorithms that optimize the objective and hopefully in polynomial time.

1 Overview

A typical minimization program can be expressed as

$$\min_{\theta\in\Theta} f(\theta)$$

where we find the θ that (approximately) minimizes objective function f over the parameter space Θ. The result of such program is generally in the form of:

> With a particular algorithm (e.g. gradient descent, Newton's Method), we can achieve the ϵ-*suboptimality* or ϵ-*stationary point* with timestep $t \geq \text{poly}(\frac{1}{\epsilon}, d, f(\theta_0) - f(\theta^*))$.

Definition 1 (ϵ-suboptimality) *Given an objective funtion $f(\theta)$ and $\epsilon \in \mathbb{R}$, the ϵ-suboptimality is achieved at timestep $t \in \mathbb{N}$ when*

$$||\theta_t - \theta^*|| \leq \epsilon$$

or,

$$f(\theta_t) - f(\theta^*) \leq \epsilon$$

where θ_t represents the value of θ at timestep t and θ^* is the optimal θ.

Definition 2 (ϵ-stationary point) *Given an objective function $f(\theta)$ and $\epsilon \in \mathbb{R}$, at timestep $t \in \mathbb{N}$, $\theta = \theta_t$ is an ϵ-stationary point when*

$$||\nabla f(\theta_t)|| \leq \epsilon$$

The time complexity of the algorithms depends on the conditions of the objection function f. Typically, we focus on the convexity and smoothness properties. As for the algorithms, we will go over the Gradient Descent (GD), Stochastic Gradient Descent (SGD), and Accelerated Gradient Descent (AGD). Note that the time complexity of SGD is beyond the scope of our class. Table 1 depicts the time complexity for each algorithm to reach the ϵ-suboptimality or ϵ-stationary point under certain conditions of f. In Section 4, we present the proofs under non-convex & L-smooth scenario with GD as well as the μ-strongly convex & L-smooth scenario with AGD.

Conditions of f		Algorithm	
Convexity	Smoothness	GD	AGD
Non-convex	L-smooth	$\mathcal{O}\left(\frac{1}{\epsilon^2}\right)$ (ϵ-stationary point)	$\mathcal{O}\left(\frac{1}{\epsilon^{7/4}}\right)$ (ϵ-stationary point) [1]
Convex	Not L-smooth	$\mathcal{O}\left(\frac{1}{\epsilon^2}\right)$ (ϵ-suboptimality)	\
Convex	L-smooth	$\mathcal{O}\left(\frac{1}{\epsilon}\right)$	$\mathcal{O}\left(\sqrt{\frac{1}{\epsilon}}\right)$
μ-strongly convex	L-smooth	$\frac{L}{\mu}\log\left(\frac{1}{\epsilon}\right)$ [3]	$\sqrt{\frac{L}{\mu}}\log\left(\frac{1}{\epsilon}\right)$ (ϵ-suboptimality)

Table 1: The time complexity of GD and AGD under different conditions of f

2 Algorithms

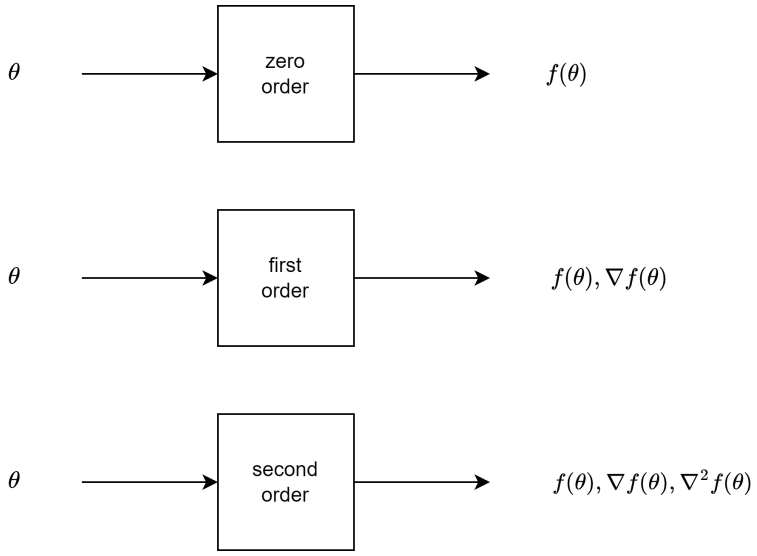
2.1 Focus of this Class

Depending on the problem we are focusing we have:

1. **Zero-order** algorithms are challenging but useful when gradient and Hessian information are difficult to obtain.
2. **First-order** algorithms are the most popular nowadays and our focus. It only requires the function to be differentiable and the information is gained from function and its derivative.

3. **Second-order** algorithms are rare because computing Hessian is computationally redundant and very time consuming.

Note that the order of the algorithm indicates how much information about the gradient is required from the algorithm. For instance, the first-order algorithm requires the function itself and the first-order derivative of the function. In this class, we are only focusing on first-order algorithms.



2.2 Gradient Descent (GD)

It's the simplest gradient descent method where the objective is to minimize a differentiable convex function f. Starting from an initial point, the gradient descent method updates the search point θ according to:

$$\theta_0 \leftarrow \text{Random Initialization}$$

$$\theta_{t+1} := \theta_t - \eta_t \nabla f(\theta_t)$$

$$\text{where } \eta_t \text{ is the learning rate or step size}$$

The rationale behind the algorithm is to move a small step along the direction that minimizes the local first-order Taylor approximation of f.

2.3 Stochastic Gradient Descent (SGD)

This version of gradient descent is more commonly used in machine learning, due to the immense computational cost of empirical risk. The algorithm is described as follow:

$$\theta_{t+1} := \theta_t - \eta_t \cdot g_t$$

where g_t is computed by

$$g_t = \nabla \ell \left(f_{\theta_t}(x_i), y_i \right)$$

$$\mathbb{E}[g_t] = \nabla \widehat{\mathcal{L}}_n(\theta_t)$$

where ℓ is the loss function, i is randomly sampled from $\{1, 2 \dots n\}$. Thus, g_t a random variable where $\mathbb{E}[g_t] = \nabla \widehat{\mathcal{L}}_n(\theta_t)$.

2.4 Accelerated Gradient Descent (AGD)

Accelerated Gradient Descent (AGD) is the most popular optimization algorithm in practice nowadays. Typical examples include Adam, Adagrad, etc.

$$\theta_{t+1} := w_t - \eta_t \nabla f(w_t)$$

$$w_{t_1} = \theta_{t+1} + r(\theta_{t+1} - \theta_t)$$

This algorithm is called "accelerated" because of its similarities with physics, where there is a momentum when descending. It's especially useful when the risk function is ill-conditioned. In other words, for a certain point on the function surface, different direction has different rate of decreasing.

For more details on the algorithms, refer *First-order and Stochastic Optimization Methods for Machine Learning*[2].

3 Conditions

3.1 L-smooth

Definition 3 *A differentiable function f is L-smooth when it satisfies the following:*

$$f(v) \leq f(w) + \langle \nabla f(w), v - w \rangle + \frac{L}{2} ||v - w||_2^2$$

or,

$$||\nabla f(w) - \nabla f(v)|| \leq L||w - v|| \quad \forall\ v,w$$

This property asserts that the gradient can not change too rapidly in a small local region.

In theory, we consider $\eta \leq \frac{2}{L}$ is capable of descending.

Edge of Stability With non-smooth function, the learning rate (or step size) needs to be decreased to avoid oscillating. Whereas with an L-smooth function, the gradient norm is automatically descreasing around stationary point $\theta : \nabla f(\theta) = 0$ and η doesn't need to be decreased. This condition makes the GD algorithms more efficient.

3.2 Convexity

3.2.1 Zero Order View

Definition 4 *For a function f:*

Convexity is given by,

$$f\left(\alpha x + (1-\alpha)x'\right) \leq \alpha f(x) + (1-\alpha)f(x')$$
$$\forall\ \alpha \in [0,1] \quad \forall\ x,\ x'$$

μ-strong convexity is given by,

$$f\left(\alpha x + (1-\alpha)x'\right) \leq \alpha f(x) + (1-\alpha)f(x') - \frac{\mu\alpha(1-\alpha)}{2}\|x - x'\|^2$$
$$\forall\ \alpha \in [0,1],\ \forall\ x,\ x'$$

3.2.2 First Order View

Definition 5 *For a differentiable function f:*

Convexity is given by,

$$f(x') \geq f(x) + \left\langle \nabla f(x), x' - x \right\rangle \quad \forall\ x,\ x'$$

μ-strong convexity is given by,

$$f(x') \geq f(x) + \left\langle \nabla f(x), x' - x \right\rangle + \frac{\mu}{2}||x' - x||^2 \quad \textit{where } \mu \geq 0$$

3.2.3 Second Order View

Definition 6 *For a twice differentiable function f:*

Convexity is given by,

$$\nabla^2 f(x) \succeq 0$$

μ-strong convexity is given by,

$$\nabla^2 f(x) \succeq \mu I$$

or,

$$\nabla^2 f(x) - \mu I \textit{ is positive semi-definite}$$

Note

From the first order view, we also have:

$$f \text{ is convex } \Rightarrow \left\langle \nabla f(x') - \nabla f(x), x' - x \right\rangle \geq 0 \quad \forall\ x, x'$$

We can use this property to choose a stationary point such that,

$$\left\langle \nabla f(x') - 0, x - x^* \right\rangle \geq 0$$

Moreover, the convexity of the objective function f also guarantees that when gradient descent is performed, the updated paramter is always steering towards the global optimum.

4 Proofs of Time Complexity

4.1 Non-convex and L-Smooth Scenario with Gradient Descent

By definition of Gradient Descent we have,

$$\theta^+ \leftarrow \theta - \eta \nabla f(\theta)$$

Let $\eta = \frac{1}{L}$. By the definition of L-smoothness,

$$f(\theta^+) = f(\theta) - \eta \nabla f(\theta) \leq f(\theta) + \left\langle \nabla f(\theta), \theta^+ - \theta \right\rangle + \frac{L}{2} \|\theta^+ - \theta\|^2$$

Plug in $\theta^+ - \theta = -\eta \nabla f(\theta)$ (update rule)

$$\begin{aligned} &= f(\theta) - \frac{\|\nabla f(\theta)\|^2}{2L} \\ &\iff \|\nabla f(\theta_t)\|^2 \leq 2L \left(f(\theta_t) - f(\theta_{t+1})\right) \quad \forall t = 0, \cdots \end{aligned}$$

We sum up over t. Notice that RHS can telescope.

$$\sum_{t=1}^{T} \|\nabla f(\theta_t)\|^2 \leq 2L \left(f(\theta_t) - f(\theta_{t+1})\right)$$

Since we know,

$$T \cdot \min_{t \in [T]} \|\nabla f(\theta_t)\|^2 \leq \sum_{t=1}^{T} \|\nabla f(\theta_t)\|^2$$

Then,

$$\min_{t \in [T]} \|\nabla f(\theta_t)\|^2 \leq \frac{2L}{T} \left(f(\theta_t) - f(\theta_{t+1})\right) \propto \sqrt{\frac{1}{T}}$$

$$\implies \|\nabla f(\theta_t)\| \propto \sqrt{\frac{1}{T}}$$

To reach ϵ-stationary point by definition 2, we need to have,

$$\theta : \|\nabla f(\theta)\| \leq \epsilon$$

$$\implies T \geq \Omega\left(\frac{1}{\epsilon^2}\right)$$

4.2 μ-strongly convex and L-Smooth Scenario with AGD

Recall the update formula for AGD where the learning rate is $\frac{1}{L}$,

$$w_{t+1} \leftarrow w_t - \frac{1}{L} \nabla f(w_t)$$

By definition of L-Smooth,

$$\begin{aligned} f(w_{t+1}) &\leq f(w_t) + \langle \nabla f(w_t), w_{t+1} - w_t \rangle + \frac{L}{2}||w_{t+1} - w_t||^2 \quad \forall w_{t+1}, w_t \\ &\leq \min_w \{f(w_t) + \langle \nabla f(w_t), w - w_t \rangle\} + \frac{L}{2}\|w - w_t\|^2 \qquad \text{(update rule)} \\ &\leq \min_{\delta \in [0,1]} \{f(w_t) + \langle \nabla f(w_t), \delta(w^* - w_t) \rangle + \frac{L}{2}\|\delta(w^* - w_t)\|^2\} \qquad (1) \end{aligned}$$

where changing to minimization over $\delta \in [0, 1]$ would form a smaller set for optimizing w.

By definition of μ-strong convexity,

$$f(w_{t+1}) \geq f(w_t) + \langle \nabla f(w_t), w_{t+1} - w_t \rangle + \frac{\mu}{2}\|w_{t+1} - w_t\|^2$$

Rearranging terms.

$$\langle \nabla f(w_t), w_{t+1} - w_t \rangle \leq f(w_{t+1}) - f(w_t) - \frac{\mu}{2}\|w_{t+1} - w_t\|^2$$

Plugging it in (1),

$$f(w_{t+1}) \leq \min_{\delta \in [0,1]} \{f(w_t) + \delta\left(f(w^*) - f(w_t)\right) - \frac{\delta^2 \mu}{2}\|w^* - w_t\|^2 + \frac{L\delta^2}{2}\|w^* - w_t\|^2\}$$

This will be proven in later classes,

$$f(w_{t+1}) \leq \min_{\delta} \{\delta\left(f(w^*) - f(w_t)\right) + \frac{(L-\mu)\delta^2}{2}\|w^* - w_t\|^2 + f(w_t)\}$$

$$\leq \min_{\delta} \{\delta\left(f(w^*) - f(w_t)\right) + \frac{(L-\mu)\delta^2}{2}\left(f(w^*) - f(w_t)\right) + f(w_t)\}$$

By setting δ,

$$\delta = \frac{\mu}{2(L-\mu)}$$

We eventually get,

$$f(w_{t+1}) \leq \frac{\mu}{4(L-\mu)}\left(f(w^*) - f(w_t)\right) + f(w_t)$$

$$f(w_{t+1}) - f(w^*) \leq \left(1 - \frac{\mu}{4(L-\mu)}\right)\left(f(w_t) - f(w^*)\right)$$

$$f(w_{t+1}) - f(w^*) \leq \left(1 - \frac{\mu}{4(L-\mu)}\right)^t \left(f(w_1) - f(w^*)\right)$$

References

[1] Chi Jin, Praneeth Netrapalli, and Michael I. Jordan. *Accelerated Gradient Descent Escapes Saddle Points Faster than Gradient Descent*. Nov. 28, 2017. arXiv: 1711.10456[cs,math,stat]. URL: http://arxiv.org/abs/1711.10456.

[2] Guanghui Lan. *First-order and Stochastic Optimization Methods for Machine Learning*. Springer Series in the Data Sciences. Cham: Springer International Publishing, 2020. ISBN: 978-3-030-39567-4 978-3-030-39568-1. DOI: 10.1007/978-3-030-39568-1.

[3] Yu Nesterov. *Introductory Lectures on Convex Programming Volume I: Basic course*.