Nonlinear Optimization Notes

Haolin Li

September 17, 2024

Abstract

This course explores unconstrained optimization problems. We begin with certificates for optimality and basic ingredients, like results from convexity. Then go through algorithms that have great impact over this field of study.

Contents

Chapter 1

Introduction and Recap

In this chapter, we will go through ideas that inspired the development of this field of study, together with math required for this class and some basic results from convexity analysis. The kind of problem we are trying to answer has the form:

$$\underset{x \in \mathbb{R}^d}{\text{minimize}} \quad f(x) \tag{1.1}$$

Often in machine learning, what we care more about is not the value itself, but the minimizer associated with. Because the objective $f(x)$ serves as a loss function, and what we want is the parameters. As a result, we sometimes slightly change the formation into:

$$\underset{x \in \mathbb{R}^d}{\arg\min} \quad f(x) \tag{1.2}$$

1.1 Calculus Review

Given a function $f(x) \in \mathbb{R}^d \to \mathbb{R}$, if it's smooth, then we can define its gradient as

$$\nabla f(x) = \begin{pmatrix} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{pmatrix}$$

The gradient at point x is the direction of steepest growth. It originates from the attempt to find a local approximation for any smooth function with an affine function. Specifically, the affine approximation is:

$$\tilde{x} \mapsto f(x) + \nabla f^T(x)(\tilde{x} - x)$$

> **Note.** Even if $f(x)$ has partial derivatives for all of the coordinates, the gradient might not exists. Take $f(x) = \sqrt{x_1^2 + x_2^2}$ as an example.

A natural idea is can we find a better approximation with a quadratic? If $f(x)$ is twice differentiable, the Hessian $\nabla^2 f(x) : \mathbb{R}^d \to \mathbb{R}^d$ helps our approximation with the following quadratic form:

$$\tilde{x} \mapsto f(x) + \nabla f^T(x)(\tilde{x} - x) + (\tilde{x} - x)^T \nabla^2 f(x)(\tilde{x} - x)$$

It's not hard to see that these are just special cases of the Taylor Series for high dimensional functions. But in optimization problems we mostly/only care about the first and second derivetives because of the optimality conditions that we will be talking about later on. No matter affine or quadratic, they are all approximations, so exactly how much deviation do they have is important. Before that, we introduce a tool we will find handy in the future.

Definition 1.1.1. Given $f : \mathbb{R}^d \to \mathbb{R}$, for any $x, s \in \mathbb{R}^d$ fixed, we define the "slice" of $f(x)$ in direction s as:

$$\varphi(t) = f(x + ts)$$

Lemma 1.1.1. Let $f : \mathbb{R}^d \to \mathbb{R}$, then $\forall x, s \in \mathbb{R}^d$, the following two statements hold:

1. If $f(x)$ is smooth, so is $\varphi(t)$ with $\varphi'(t) = s^T \nabla f(x + ts)$.
2. If $f(x)$ is twice differentiable, so is $\varphi(t)$ with $\varphi''(t) = s^T \nabla^2 f(x)s$

It's not hard to see how the differentiability is passed down to $\varphi(t)$. With this lemma, we can upper bound the approximation error in any direction with the following two theorems.

Theorem 1.1.1 (Taylor Expansion First-Order Approximation)**.** Let $f(x)$ have L-Lipschitz continuous $\nabla f(x)$. Then for any $x, s \in \mathbb{R}^d$, we have

$$|f(x + ts) - (f(x) + t\nabla f^T(x)s)| \leq \frac{L}{2} t^2 \|s\|^2 \tag{1.3}$$

In addition, if $f(x)$ has a Q-Lipschitz Hessian(operator norm). We have

$$|f(x + ts) - (f(x) + t\nabla f^T(x)s + \frac{t^2}{2} s^T \nabla^2 f(x)s)| \leq \frac{Q}{6} t^3 \|s\|^3 \tag{1.4}$$

This means, if we enforce Lipschitz condition on $f(x)$, which is not very strict in practice, then the approximation error can be upper bounded by the square or the cude of $t\|s\|$. This shouldn't be surprising since it matches the residual of Taylor expansion.

1.2 Optimality Conditions

In this section, we use the proposed theorem to explain why do we care specifically about the first and second derivatives. Local optimality can be checked by examining the first and second order derivatives.

Theorem 1.2.1 (First-order necessary)**.**
Suppose $f(x) \in C^1$, then $x*$ is a local minimizer$\Rightarrow$ $\nabla f(x*) = 0$

Theorem 1.2.2 (First-order sufficient)**.**
Suppose $f(x) \in C^1$ and is also convex, then $x*$ is a local minimizer $\Leftrightarrow$ $\nabla f(x*) = 0$

Theorem 1.2.3 (Second-order necessary)**.**
Suppose $f(x) \in C^2$, then $x*$ is a local minimizer $\Rightarrow$ $\nabla^2 f(x*) \succeq 0$

Theorem 1.2.4 (Second-order necessary)**.**
Suppose $f(x) \in C^2$, then $x*$ is a local minimizer $\Rightarrow$ $\nabla^2 f(x*) \succ 0$

1.3 Basics of Convexity

Definition 1.3.1 (Convex Set)**.** A set $C \subseteq \mathbb{R}^d$ is convex if given any $x, y \in \mathbb{R}^d$ and $\lambda \in [0, 1]$, we have $tx + (1 - t)y \in C$.

Definition 1.3.2 (Epigraph)**.** Let $f(x) : \mathbb{R}^d \to \mathbb{R}$, then $epi(f) = \{(x, t) : t \geq f(x)\}$

Lemma 1.3.1. $f(x)$ is convex $\Leftrightarrow$ $epi(f)$ is convex

Lemma 1.3.2 (Operations Perserving Convexity). Assume $C_1, C_2 \subseteq \mathbb{R}^d$ and $C_3 \subseteq \mathbb{R}^n$ are convex sets.

1. (Scaling) $\mathbb{R}_+ \cdot C_1$
2. (Minkovski Sum) $C_1 + C_2$
3. (Intersections) $C_1 \cap C_2$
4. (Affine image and preimage) Let $\mathcal{A} : \mathbb{R}^d \to \mathbb{R}^n$ be affine, then $\mathcal{A}(C_1)$ and $\mathcal{A}^{-1}C_3$ are convex.

Now we will check how can we characterize smooth convex functions with the gradient.

Proposition 1.3.1 (First-order Characterization of Smooth Convex Function). Suppose $f(x) : \mathbb{R}^n \to \mathbb{R}$ is differentiable, then TFAE

1. $f(x)$ is convex
2. $\forall x, y \in \mathbb{R}^d$, we have $f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle$
3. $\forall x, y \in \mathbb{R}^d$, we have $\langle \nabla f(y) - \nabla f(x), y - x \rangle \geq 0$

Note. To memorize the direction of the inequality, we should think the first order approximation supporing the entire convex function from below.

Lemma 1.3.3 (Second-order Characterization of Convex Function). Assume convex function $f : \mathbb{R}^d \to \mathbb{R}$ is twice differentiable, then we have $f(x)$ is convex $\Leftrightarrow \nabla^2 f(x) \succeq 0, \forall x \in \mathbb{R}^d$

In the above discussion, we assumed that the convex functions are at least differentiable. Then how can we verify optimality if $\nabla f(x)$ doesn't exist? We introduce subdifferential as a loose local linear approximation.

Definition 1.3.3 (Subdifferential). The subdifferential fo f at $x \in \mathbb{R}^d$ is

$$\partial f(x) = \{v \in \mathbb{R}^d : f(y) \geq f(x) + \langle v, y - x \rangle, \ \forall y\}$$

From the definition, it's not hard to see that the idea of subgradient originates from proposition 1.3.1, which is a mathematical way to formulate the intuition we just talked about. The introduction of subdifferential and subgradient is useful because of the following theorem.

Theorem 1.3.1. Suppose $f : \mathbb{R}^d \to \mathbb{R}$ is convex, then x^* is a global minimizer $\Leftrightarrow 0 \in \partial f(x^*)$

When we are calculating the subdifferential of a function, often we will need to deal with operations between functions like add and composition.

Proposition 1.3.2 (Subdifferential Calculus). Suppoese $f, h : \mathbb{R}^d \to \mathbb{R}$ are convex, and $A : \mathbb{R}^n \to \mathbb{R}^d$, then

1. $\partial(f + h)(x) = \partial f(x) + \partial f(x)$
2. $\forall \alpha \in \mathbb{R}, \ \partial(\alpha f(x)) = \alpha \partial f(x)$
3. $\partial(f \circ A(x)) = A^T \partial f(Ax)$
4. If f is differentiable, then $\partial f(x) = \{\nabla f(x)\}$
5. Given $x \in \mathbb{R}^d$. Define $M(x) = \{i \in \{1, 2\} : \max_{j=1,2} = f_i(x)\}$. Then we have, $\partial(\max(f_1, f_2))(x) = conv\{\partial f_i(x) : i \in M(x)\}$

Chapter 2

Algorithms

In this section, we will go through several optimization algorithms and some bounds.

2.1 Gradient Descent

One of the most famous and frequently used optimization algorithms is the gradient descent, which can be formulated as:

$$x_{k+1} \leftarrow x_k - \alpha_k \nabla f(x_k) \tag{2.1}$$

Interestingly, this update formula can be interpreted in the following way:

$$x_{k+1} = \arg\min_x \{f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + \frac{1}{2\alpha_k} \|x - x_k\|^2\}$$

In fact, if we denote $h(x) = f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + \frac{1}{2\alpha_k} \|x - x_k\|^2$, by checking the first-order necessary condition, we have

$$\nabla h(x) = \nabla f(x_k) + \frac{1}{\alpha_k}(x - x_k)$$

By setting $\nabla h(x) = 0$, we derive $x_{k+1} = x = x_k - \alpha_k \nabla f(x_k)$, which is the same as formula 2.1.

Note. We can see that $h_k(x)$ is the Taylor expansion of $f(x)$ at x_k.

Since we are interested in the minimization and maximization potential of the update algorithm, it's worth to explore how the corresponding value $f(x)$ changes.

Lemma 2.1.1 (Descent Lemma). Let $f : \mathbb{R}^d \to \mathbb{R}$ have L-Lipschitz gradient and $k \geq 0$, we have

$$f(x_{k+1}) \leq f(x_k) - (\alpha_k - \frac{L\alpha_k^2}{2})\|\nabla f(x_k)\|^2$$

Proof. Only need to use the first-order Taylor bound:

$$|f(x_{k+1}) - f(x_k) - \langle \nabla f(x_k), x_{k+1} - x_k \rangle| \leq \frac{L}{2}\|x_{k+1} - x_k\|^2$$

■

To minimize $f(x)$, we want the upper bound in lemma 2.1.1 to be as small as possible, meaning that we need to maximize $\alpha_k - \frac{L\alpha_k^2}{2}$. By solving this simple quadratic, we derive the ideal step size $\alpha_k = \frac{1}{L}$. However, this can be quite impractical since even if it's not a strong assumption to take functions we see in rea life over a fintie domain as Lipschitz, it can be difficult to determine the Lipschitz constant L. If we examine the update formula 2.1 closely, the only thing that needs external care is the step size α_k. As a result, in the following we will discuss how to find a both effectively and pratically way to determine the step size α_k.

2.1.1 Exact Linesearch

A straight forward approach is to find the solution of the following optimization problem, which is also referred to as the exact line search:

$$\alpha_k = \arg\min_{\alpha \in \mathbb{R}} f(x_k - \nabla f(x_k))$$

Yet this can also be impractical because it means we have to solve an optimization problem at each step of update. In replacement to exact linesearch, we introduce the following backtracking linesearch algorithm.

2.1.2 Backtracking Linesearch

The idea of backtracking linesearch is quite simple, we start with a step size large enough, and then shrink it little by little until we observe sufficient descent. This can be formalized with the following two steps:

1. Pick $\alpha \in \mathbb{R}_+$ and $\tau \in (0, 1)$, decrease step size by $\alpha_n = a\tau^n$.
2. To measure the descent, we use the so-called Armijo condition.

Definition 2.1.1 (Armijo Condition). Pick $\eta \in (0, 1)$, declare sufficient descent when $f(x_k - \alpha \nabla f(x_k)) \leq f(x_k) - \eta\alpha \|\nabla f(x_k)\|^2$

Therefore, we can restate the backtracking algorithm as:

$$\alpha_k = \max_n \{a\tau^n : \text{Armijo condition holds for } \alpha = a\tau^n\}$$

One nice thing about Armijo's condition is that it clarifies the vague term "sufficient decsent". The right hand side of Armijo's condition is affine in α, but the descent lemma 2.1.1 was in the quadratic of α. This implies the existence of an α small enough such that the quadratic bound is lower than the linear bound. To be even clearer, we have the following range for step size α.

Lemma 2.1.2. The Armijo condition holds for

$$\alpha \in [0, \frac{2(1-\eta)}{L}]$$

Proof. Using the descent lemma, and bound the right hand side with linear upper bound, we have

$$f(x_k - \alpha \nabla f(x_k)) \leq f(x_k) - (\alpha - \frac{L\alpha^2}{2})\|\nabla f(x_k)\|^2 \leq f(x_k) - \eta\alpha\|\nabla f(x_k)\|^2$$

This is the same as

$$\eta\alpha \leq \alpha - \frac{L}{2}\alpha^2$$

■

Note (Consequences of Backtracking Linesearch). Firstly, for each search for step size α, it needs no more than $\lceil \log_{\frac{1}{\tau}}(\frac{\alpha L}{2(1-\eta)}) \rceil$ steps to terminate. This bound is derived by asking $\alpha\tau^n \leq$ the bound in 2.1.2. Secondly, we have $\alpha_k \geq \min\{a, \frac{2\tau(1-\eta)}{L}\}$ because we either find the sufficient descent at the first step, or find it no smaller than τ times the upper bound in 2.1.2, which is the next iteration of step size. So we can rewrite the Armijo condition as:

$$f(x_{k+1}) \leq f(x_k) - \eta\alpha_k\|\nabla f(x_k)\|^2 \tag{2.2}$$

$$\leq f(x_k) - \eta\min\{a, \frac{2\tau(1-\eta)}{L}\} \cdot \|\nabla f(x_k)\|^2 \tag{2.3}$$

Now in the special case where $a \geq \frac{1}{L}$ and $\eta = \tau = \frac{1}{2}$, we have

$$
\begin{aligned}
(2.3) &\leq f(x_k) - \frac{1}{2}\min\{\frac{1}{L}, \frac{1}{2L}\} \cdot \|\nabla f(x_k)\|^2 \\
&\leq f(x_k) - \frac{1}{4L}\|\nabla f(x_k)\|^2
\end{aligned}
$$

which does not result in a great sacrifice on the tightness of the bound compared with the quadratic in the descent lemma when the step size is small enough or asking $L > 1$.

A plot is available at *Nonlinear_ Optimization/plot_ maker/Armijo_ and_ Descent_ Bound/plot.py*, python=3.11.5 + holoviews.

2.2 NonConvex Smooth Optimization Guarantee

Equipped with the bounding results on $f(x_{k+1})$ from the previous chapter, without convexity, we will see what we can say about L-smooth function and update rule:

$$x_{k+1} \leftarrow x_k - \alpha_k \nabla f(x_k)$$

Theorem 2.2.1 (Bound on Average Gradient Norm for Nonconvex Functions)**.** Suppose $f(x)$ is L-smooth, then for any $T > 0$, we have:

$$\frac{1}{T}\sum_{k=0}^{T-1}\|\nabla f(x_k)\|^2 \leq \frac{2L}{T}(f(x_0) - \min_{0\leq k\leq T-1} f(x_k)) \leq \frac{2L}{T}(f(x_0) - \min_{x\in\mathbb{R}^d} f(x))$$

when we set $\alpha_k = \frac{1}{L}$, i.e. exact linesearch.

In addition, with Armijo backtracking, we have:

$$\frac{1}{T}\sum_{k=0}^{T-1}\|\nabla f(x_k)\|^2 \leq \frac{\max\{\frac{1}{\eta a}, \frac{L}{2\tau\eta(1-\eta)}\}}{T}(f(x_0) - \min_{x\in\mathbb{R}^d} f(x))$$

Proof. Use the bound of $f(x_{k+1})$ we derived from the previous section, i.e. descent lemma 2.1.1 and Armijo. Sum all of the inequalities up, then change $f(x_T)$ to $\min_k f(x_k)$ ■

This theorem describes the average pattern of the sequence of the gradients $\{\nabla f(x_k)\}$, it indicates that there exists at least one x_k^* such that $\|\nabla f(x_k^*)\|$ itself satisfies the inequality. Now let's first define what is a second-order critial point, and then see what we can get if we enforce convexity on the function.

Definition 2.2.1 (Second-Order Critial Point)**.** Given $f(x)$ twice differentiable, a point $x^* \in \mathbb{R}^d$ is said to be the critial point of $f(x)$ if we have

$$\nabla f(x^*) = 0 \text{ and } \nabla^2 f(x^*) \succeq \lambda I_n, \ \exists\lambda > 0$$

Theorem 2.2.2. Assume $f(x) : \mathbb{R}^d \to \mathbb{R}$ is twice differentiable and x^* being its second-order critial point, and there exists $\epsilon > 0$, such that if $x_k \in B_\epsilon(x^*)$ then $x_l \in B_\epsilon(x^*)$, $\forall l \geq k$. Then, if x_0 is close to x^* we have:

$$f(x_T) - f(x^*) \leq (1 - \frac{\lambda^2}{4L^2})(f(x_{T-1}) - f(x^*))$$

This result has a strong implication, it means if it satisfies the conditions stated, then we have a quite fast(exponential) approximation to the optimal function value. Because both λ and L are constant once $f(x)$ is settled.