Nonlinear Optimization Notes

Haolin Li

October 4, 2024

Abstract

This course explores unconstrained optimization problems. We begin with conditions for optimality and basic ingredients, like results from convexity. Then go through algorithms that have great impact over this field of study together with analysis on their convergence behavior under certain conditions.

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:

$$\begin{array}{ll} \text{minimize} \\ x \in \mathbb{R}^d \end{array} \quad f(x) \tag{1.1}$$

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:

$$\begin{array}{ll} \arg\min \\ x \in \mathbb{R}^d \end{array} \quad f(x) \tag{1.2}$$

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 theorems.

Theorem 1.1.1 (Taylor 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.

Note. Taylor approximation comes in handy when we are trying to interpolate between function fluctutaions and the distance of change in x. It means whenever you have a bound in terms of the norm of s, you can rewrite it in terms of function values. Specifically, one can substitute the abs as brackets if x is the minimizer of convex function $f(x)$.

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 $\langle \nabla f(y) - \nabla f(x), y - x \rangle \geq 0$

Remark. To memorize the direction of the inequality in the second statement, we should think the first order approximation as the hyperplane supporting the entire epigraph from below. In addition, the last property is also referred to as monotonicity in the gradient, which is easy to verify for $\mathbb{R} \to \mathbb{R}$ where convexity equals to $f^{''}(x) \geq 0$. In higher dimensional spaces like $\mathbb{R}^d \to \mathbb{R}$, convexity actually implies the same monotonicity in directional derivatives. To formalize, we know that the directional derivative of f at x in direction d is :

$$D_f(x; d) = \nabla f(x)^T d$$

For the sake of the arguement, let's say x is fixed and we can move y around. Then for any direction $y - x$ pointing out of x, from the last condition in 1.3.1,

$$D_f(x; y - x) = \nabla f(x)^T (y - x)$$

Now, let's fix y as well and compute the directional gradient of y still in the directin of $y - x$, we have

$$D_f(y; y - x) = \nabla f(y)^T (y - x)$$

Combining these two and the convexity condition, we have

$$D_f(y; y - x) - D_f(x; y - x) = \langle \nabla f(y) - \nabla f(x), y - x \rangle \geq 0$$

To encapsulate, for any given point x and direction h, directional derivative $D_f(x; th)$ is monotonically non-decreasing in t, which can also be equally characterized by the following lemma.

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

Grandient Descent

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

2.1 Algorithm: 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:

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

This means x_{k+1} is the minimizer of the first-order approximation under quadratic distance penalty. 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. One straightforward understanding of α_k is the step size taken as illustrated in 2.1. However, as a quadratic in $h(x)$, it also represents how tolerant we are to distant condidates. Higher α_k indicates greater tolerance, corresponding to the gentler speed of change in the quadratic.

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:

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

■

Corollary 2.1.1. If we take $\alpha_k = \frac{1}{L}$ as in exact linesearch, then we have:

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

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 real life over a fintie domain as Lipschitz, it can be difficult to determine the Lipschitz constant L for an unknown function. 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:

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

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 by *Larry Armijo*.

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 in $f(x)$. This can be formalized with the following two steps:

1. Pick $\alpha \in \mathbb{R}_{>0}$ 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.

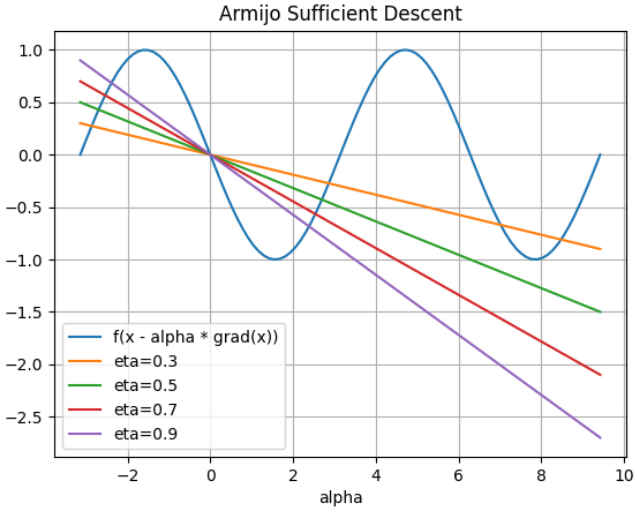
Definition 2.1.1 (Armijo Condition). Pick $\eta \in (0, 1)$, declare sufficient descent when

$$f(x_{k+1}) = f(x_k - \alpha \nabla f(x_k)) \leq f(x_k) - \eta\alpha \|\nabla f(x_k)\|^2$$

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". Beacuse we have $\eta \in (0, 1)$, we certainly have the linear upper bound in 2.1.1 to be greater than $f(x - \alpha \cdot \nabla f(x))$ for $\alpha \in (0, \epsilon)$. One thing worth noticing is that, for η small enough, it is possible for us to obtain multiple disconnected intervals with α satisfying the sufficient descent condition like $\eta = 0.3$.



Specifically, the following lemma describes the first Armijo interval near 0.

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, in searching 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

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

$$\leq f(x_k) - \frac{1}{4L}\|\nabla f(x_k)\|^2$$

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 $f(x_k)$ to $\min f(x)$ ■

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 for functions with second-order critial points, or looser strongly convex functions, we have a quite fast(exponential) rate of approximation to the minimum. Because both λ and L are constant once $f(x)$ is settled.

2.3 Better Guarantee for Convex Functions

In the previous section, we've seen that under L-smooth assumption only, the guarantee in 2.2.1 is barely informative because it only tells about the average behavior. Although we do have the desired exponential drop, it requires the minimizer to be second-order critical. In this section we study what nice properties can be guaranteed if we add convexity to $f(x)$. We begin with analytical properties of L-smooth convex functions.

Lemma 2.3.1 (Charactrization of L-Smooth Convex Functions). Suppose $f : \mathbb{R}^d \to \mathbb{R}$ is convex and differentiable, TFAE

1. $f(x)$ has L-Lipschitz gradient / L-smooth
2. $\frac{L}{2}\|\cdot\|^2 - f(\cdot)$ is convex
3. $f(y) \leq f(x) + \langle \nabla f(x), y - x\rangle + \frac{L}{2}\|x - y\|^2,\ \forall x, y \in \mathbb{R}^d$
4. $\langle \nabla f(y) - \nabla f(x), y - x\rangle \geq \frac{1}{L}\|\nabla f(y) - \nabla f(x)\|^2$
5. If $f(x)$ is twice differentiable, $\nabla^2 f(x) \preceq L I_d,\ \forall x \in \mathbb{R}^d$

Remark. The interpretation of the statements can be interesting. The second property puts an "upper bound" on the convexity of $f(x)$ to be no greater than that of the quadratic $\frac{L}{2}\|\cdot\|$. As a L-smooth function, the growth rate of $f(x)$ can be limited by the first-order Taylor approximation, thus can be upper bounded by a quadratic while all convex functions are lower bounded by the supporting hyperplane/subgradient, aligning with the third statement. While the direct definition for L-smoothness is the upper bounded rate of change for the gradient with respect to the drift of x, condition 4 introduces a new perspective. It states that the rate of change of $\nabla f(x)$ is lower bounded by the change of the gradient itself.

Now given these charactrizations, we can provide some guarantees of convergence using gradient descent. Next, we will see what we can get with exact linesearch. But first, we will state an interesting property for exact line search over an L-smooth function.

Proposition 2.3.1. Let $f : \mathbb{R}^d \to \mathbb{R}$ be L-smooth and x^* be its minimizer. Then once $x_k \in B_\epsilon(x^*)$, we have $x_l \in B_\epsilon(x^*)$, $\forall l \geq k$.

Theorem 2.3.1 (Convex L-Smooth Convergence Guarantee). If $f : \mathbb{R}^d \to \mathbb{R}$ is convex and L-smooth, let x^* be a minimizer of $f(x)$, then by setting $\alpha_k = \frac{1}{L}$ we have:

$$f(x_{k+1}) - \min_{x \in \mathbb{R}^d} f(x) \leq \frac{2L}{k} \|x_0 - x^*\|^2$$

Note. Some tricks used in the proof is worth reviewing, can be useful in some sequential-relation equations. This theorem states that the speed of convergence for convex L-smooth functions are determined by both the quality of initialization(x_0) and the number of iterations.

Corollary 2.3.1. We can derive a similar guarantee for Armijo's backtracking linesearch by substituting the constant L with something else.

2.4 Even Better for Strongly Convex Functions

Convexity is nice, and we have seen what we can get out of functions with their Hessians upper bounded by LI_n. In this section, we explore functions with even stronger convexity conditions.

Definition 2.4.1 (Strong Convexity). A function $f : \mathbb{R}^d \to \mathbb{R}$ is μ-strongly convex if the following is also convex

$$x \mapsto f(x) - \frac{\mu}{2}\|x\|^2$$

Lemma 2.4.1 (Charactrization of μ-Strongly Convex Functions). If $f : \mathbb{R}^d \to \mathbb{R}$, TFAE

1. $f(x)$ is μ-strongly convex
2. $f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle + \frac{\mu}{2}\|y - x\|^2, \ \forall x, y$
3. $\langle \nabla f(y) - \nabla f(x), y - x \rangle \geq \mu \|y - x\|^2, \ \forall x, y$
4. If $f(x)$ is further twice differentiable, then $\nabla^2 f(x) \succeq \mu I_d$

Remark. Since convexity is about having a lower bound on curvature, strong convexity basically implies stronger curvatures. In addition to the linear term, the second statement offers a quadratic lower bound on the increase of function values. The monotonicity condition moves to a strictly positive number from 0 as indicated in the third condition. The last one is in great contrast to L-smoothness where the equivalent condition has the opposite direction of inequality $\nabla^2 f(x) \preceq LI_d$.

Again, provided the the equivalent charactrizations, let's explore the gurantees for the convergence of strongly convex functions.

Theorem 2.4.1 (Better Guarantees for Strongly Convex Functions). Let $f : \mathbb{R}^d \to \mathbb{R}$ be μ-strongly convex that's L-smooth. Then, Gradient Descent with $\alpha_k = \frac{2}{\mu + L}$ satisfies that

$$f(x_{k+1}) - \min_{x \in \mathbb{R}} f(x) \leq \frac{L}{2}\left(\frac{\kappa - 1}{\kappa + 1}\right)^{2k} \|x_0 - x^*\|^2$$

where we have $\kappa = \frac{L}{\mu}$.

Remark. For L-smooth functions we have $\nabla^2 f(x) \preceq LI_d$, and if it is also μ-strongly convex we have $\nabla^2 f(x) \succeq \mu I_d$. In other words, we have

$$LI_d \preceq \nabla^2 f(x) \preceq \mu I_d$$

To prove this theorem, we can use the following lemma who lower bounds the rate of increase in terms of both the drift in the input and the difference in the function gradients.

Lemma 2.4.2. Given any function $f(x)$ L-smooth and μ-strongly convex, we have that

$$\langle \nabla f(y) - \nabla f(x), y - x \rangle \geq \frac{\mu L}{\mu + L} \|x - y\|^2 + \frac{1}{\mu + L} \|\nabla f(x) - \nabla f(y)\|^2$$

Proof of 2.4.1. We want to bound the difference in function values, one approach is to first derive bounds on $\|x_{k+1} - x^*\|^2$ and then transfer that inequality condition to function difference via first-order Taylor approximation. We first expand the term $\|x_{k+1} - x^*\|^2$ by the definition of the norm. By using the lemma above, we can see why the step size was constructed in a way that the gradient term $\|\nabla f(x_k)\|^2$ can be eliminated. Then apply Taylor approximation. ■

2.5 Algorithm: Nesterov's Accelerated Gradient Descent

The following table summarizes the rate of convergence for L-smooth(and strongly convex) functions with exact linesearch update rule. We have also seen the importance of step sizes over the convergence behavior of $f(x)$. The question is, can we do better? Previously we relied on stronger properties of $f(x)$ to get better results, is there a better update rule such that all of the functions can benefit from? The answer is Nesterov's accelerated gradient descent, which is also the topic for this section.

Function Condition	$f(x_k) - \min f(x)$
L-Smooth	$O\left(\frac{1}{k}\right) \cdot \|x_0 - x^*\|^2$
L-smooth and μ-strongly convex	$O\left(\left(\frac{\kappa-1}{\kappa+1}\right)^{2k}\right) \cdot \|x_0 - x^*\|^2$

Table 2.1: Recap on the rate of convergence for functions with different properties.

Algorithm 2.1: Corrected Nesterov's Accelerated Gradient Descent

Input: x_0, $f(x)$, K, L
Output: Minimizer x_K
Initialize: $\lambda_0 \leftarrow 0$, $y_0 \leftarrow x_0$;
for $k = 0$ **to** $K - 1$ **do**
  $y_{k+1} \leftarrow x_k - \frac{1}{L}\nabla f(x_k)$;
  $\lambda_{k+1} \leftarrow \frac{1+\sqrt{1+4\lambda_k^2}}{2}$;
  $\gamma_k \leftarrow \frac{\lambda_k - 1}{\lambda_{k+1}}$;
  $x_{k+1} \leftarrow y_{k+1} + \gamma_k(y_{k+1} - y_k)$;
return x_K

This algorithm is also named as the *momentum gradient descent* and the guarantee we will offer is over variables y_k, in other words, x_k are taken as intermediate states in this case. In this sense, we should start our interpretation with $x_k = y_k + \gamma_k(y_k - y_{k-1})$, which is a momentum shift to the direction $y_k - y_{k-1}$. Then moving from x_k, an exact linesearch step forward to obtain y_{k+1}. Although this might seem like a torsion compared with the statement of the algorithm above, they are actually the same thing. The algorithm computes y_{k+1} in the first place because we need it to initiate the algorithm. But once we have started, the interpretation should work like what's stated here.

Lemma 2.5.1. The sequence $\{\lambda_k\}$ satisfies

$$\lambda_{k+1}^2 - \lambda_{k+1} = \lambda_k \text{ and } \lambda_k \geq \frac{k+1}{2}$$

Lemma 2.5.2. For any $u, v \in \mathbb{R}^d$, given $f(x)$ convex and L-smooth, we have

$$f(u - \frac{1}{L}\nabla f(u)) - f(v) \geq -\frac{1}{2L}\|\nabla f(u)\|^2 + \nabla f(u)^T(u - v)$$

Theorem 2.5.1 (Nesterov's Convergence for L-smooth Convex Functions)**.** Let $f(x)$ be convex and L-smooth, and $x^* \in \arg\min_{\mathbb{R}^d} f(x)$, then we have

$$f(y_k) - \min f(x) \leq \frac{2L}{k^2}\|x_0 - x^*\|^2$$

2.6 Lower Bound

Until this point, we have seen many different algorithms under the regime of gradient descent that offers guarantees on the rate of convergence under certain conditions, like L-smooth, convex, even strongly convex. One intersting and vital question to ask is: what is the best we can do? What is the best we can do with gradient descent algorithms? In this section, we offer an lower bound on the rate of convergence using gradient descent.

To be clear of our scope of study, we restrict ourselves with the following way of update:

$$x_k \in span\{x_0, \nabla f(x_0), \ldots, \nabla f(x_{k-1})\}$$

This relation should directly give us an intuition on why this method has a lower bound, especially when k is small. As the dimension of x getting larger, we need more gradients serving as a new basis of the input space so as to explore in any direction we want, which should be what we need when we are trying to optimize a hard function. This idea corresponds to the following theorem:

Theorem 2.6.1 (Lower Bound on Convergence with Small k)**.** For any $1 \leq k \leq \frac{1}{2}(d-1)$ and $L \geq 0$. There exists a function $f : \mathbb{R}^d \to \mathbb{R}$ with L-Lipschitz gradient such that any algorithm falling in the above regime has the following holds:

$$f(x_k) - \min f \geq \frac{3L}{32}\frac{\|x_0 - x^*\|^2}{(k+1)^2}$$

and $\|x_k - x^*\| \geq \frac{1}{2}\|x_0 - x^*\|^2$.

In other words, no matter how smart your algorithm is, as long as it's gradient-based, there always exists a hard function that can't be optimzied well when k is small. However, one important thing to notice is that this belief is invalid once we go beyond $\frac{1}{2}(d-1)$ number of iterations. As a matter of fact, there exists an algorithm specifically tailored for minimizing $\|Ax - b\|^2$ whose loss suddenly drops steeply as soon as k gets larger than a threshold. So it's worth noticing that after the threshould the convergence behavior is mainly controlled by the upper bound we studied before.

Chapter 3

Proximal Method

In this chapter we will study proximal method which can be seen as a generalization of gradient descent. Before formally introduce the formulation, we will first describe a few examples and how the idea is developed.

3.1 Motivating Problems

Many problems in real life can be ill-posed due to the lack of information. For example, in solving the linear system

$$Ax = b, \quad A \in \mathbb{R}^{n \times p}$$

It can be solved if A is tall and thin, i.e. $n \gg p$. But for most of the time in science, what available is a fat and short one. This means the matrix A is probably not full-rank, which means the solution is not unique. But picking from a space of solution can be subjective sometimes and hard to justify. A natural idea is to add additional restrictions(structure) on top of the existing problem so as to narrow down the parameter space where we are exploring for answers. In this way, we can hope/guarantee the uniqueness of the solution. One way to solve this problem is through adding L^1 norm restriction on the parameters x, which is also famously known as Lasso regression:

$$\min_{x \in \mathbb{R}^d} \|Ax - b\|^2 + \lambda \|x\|_1$$

By adding this restriction, we are favoring smaller parameters in hope for better robustness.

Another interesting exmaple is the Netflix Recommendation System challenge which can be mathematically formulated as a matrix recovery problem. Given a matrix $X \in \mathbb{R}^{d_1 \times d_2}$ such that

$$\mathcal{A}(X) = b$$

where operator $\mathcal{A}$ maps $X \mapsto \mathbb{R}^m$ with $m \ll d_1 \times d_2$. Similarly, in order to preserve the low-rank property, the problem is formulated as:

$$\min \|\mathcal{A}(X) - b\|_2^2 + \lambda \|X\|_*$$

where $\|X\|_* = \sum_{i=1}^{\min d_1, d_2} \sigma_i(X)$ refers to the nuclear norm.

> **Remark.** The common pattern between the above two examples is that they are all formed in the following way
>
> $$\min_x f(x) + h(x)$$
>
> where $f(x)$ is smooth/differentiable, and $h(x)$ is convex and has a nice decomposition.

This insight leads us into the object we are going to study.

3.2 Proximal Operator

Algorithms are usually inspired by the idea of approximation, just like gradient descent. We proved that the update rule

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

is equivalent to solving the following approximation

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

As a result, gradient descent can be viewed as a special case of proximal method where $f(x) = f(x_k) + \langle \nabla f(x_k), x - x_k \rangle$ is differentiable, and $h(x) = \frac{1}{2\alpha_k}\|x - x_k\|^2$ is convex. To formulate this, we consider any given closed convex function $h : \mathbb{R}^d \to \mathbb{R} \cup \{+\infty\}$, the proximal operator is defined as

$$prox_{\alpha h}(x) = \arg\min_{z \in \mathbb{R}^d}\{h(z) + \frac{1}{2\alpha}\|z - x\|^2\} \tag{3.1}$$

One thing that needs clarification is that we used $=$ instead of $\in$ in the definition of the proximal operator, this is guaranteed by the fact that $h(z) + \frac{1}{2\alpha}\|z - x\|^2$ is a strongly convex function, resulting in a unique minimizer. This gives the following statement.

Lemma 3.2.1. The proximal operator is well defined.

The proximal operator can be also seen as a special case of the following configuration:

$$\arg\min_{x \in \mathbb{R}^d} f(x) + h(x)$$

where $f(x)$ is smooth and $h(x)$ closed convex. We have the following necessary condition for this optimization problem.

Lemma 3.2.2. Let $h : \mathbb{R}^d \to \mathbb{R} \cup \{+\infty\}$ be a closed convex function and $f : \mathbb{R}^d \to \mathbb{R}$ be a smooth function. Let $x^* \in \arg\min f(x) + h(x)$, then we have

$$-\nabla f(x^*) \in \partial h(x^*)$$

Proof. The proof is done by using the definition of subdifferential, the definition of the gradients, and that x^* is the minimizer. ■

Note. We can't prove this using subdifferential calculus since it requires both $f(x)$ and $h(x)$ to be convex.

Furthermore, if we ask $f(x)$ to be also convex, then this is not only necessary, but the sufficient condition.

Lemma 3.2.3. Let $h(x)$ be closed convex and $f(x)$ smooth convex. Then we have

$$x^* \in \arg\min_x f(x) + h(x) \Leftrightarrow -\nabla f(x^*) \in \partial h(x^*)$$

As a special case of the above lemma, where $f(z) = \frac{1}{2\alpha}\|x - z\|^2$ is smooth convex, the output of proximal operator satisfies the necessary sufficient condition.

Corollary 3.2.1.

$$x^+ = prox_{\alpha h}(x) \Leftrightarrow \frac{x - x^+}{\alpha} \in \partial h(x^+)$$

3.3 Interpretation on Proximal Operator

The interpretation can be explain from two different perspectives, the first one arises from the analysis of the general composite optimization problem

$$\arg\min_{x \in \mathbb{R}^d} f(x) + h(x)$$

with smooth $f(x)$ and closed convex $h(x)$.

First View

Naturally we want to update our step by gradient descent, but the non-differentiablility of $h(x)$ stops us from doing so. As a result, we take a step back and try to analyze the approximation problem by substituting $f(x)$ with $f_k(x)$, where

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

We introduce iteration update with footnote k because we are not making exact minimization calculation but an approximation, and hoping to obtain some convergence behavior with big k. Right now the problem of exact minimization is reduced to the following iteration update

$$x_{k+1} \leftarrow \arg\min_{x \in \mathbb{R}^d} f_k(x) + h(x)$$

By using the necessary condition lemma 3.2.2, we derive the following relation

$$\frac{x_k - \alpha_k \nabla f(x_k) - x_{k+1}}{\alpha_k} \in \partial h(x_{k+1})$$

This is quite indicative, because by referring to corollary 3.2.1, we actually get

$$x_{k+1} = prox_{\alpha_k h}(x_k - \alpha_k \nabla f(x_k)) \tag{3.2}$$

Second View

The second interpretation stems from equation 3.2. It is easy to see that the input of the proximal operator is in fact one step of gradient descent with respect to $f(x)$ only. Then based on this gradient descent updated point $x_k - \alpha_k \nabla f(x_k)$, we try to update $h(x)$ with an additional penalty who prevents the second update from going too crazy. The second step is achieved through the proximal operator.

3.4 Forward-Backward Method

Now let's consider the general optimization problem

$$\min_{x \in \mathbb{R}^d} f(x) + h(x)$$

with smooth $f(x)$ and closed convex $h(x)$. Recall that gradient descent can be derived as a proximal operator, similarly, for this problem we can derive the following linear approximation

$$x_{k+1} \leftarrow \arg\min_{x \in \mathbb{R}^d} \underbrace{h(x)}_{\text{closed convex}} + \underbrace{f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + \frac{1}{2\alpha_k}\|x - x_k\|^2}_{\text{smooth}}$$

TBD

3.5 Relation between Newton's Method and Projection

Let sets $C_1 = \{(x, y) : y \leq 0\}$ and $C_2 = epi(f)$ with $f(x)$ closed convex. For the sake of arguement, let's assume there exists some $x^* \in \mathbb{R}$ such that $f(x^*) = 0$ and $f(x) > 0$, $\forall x > x^*$. In other words, we will analyze the Newton's method from the right hand side of x^*.

As shown in the figure, we start at some p_n on the boundary of C_2, where $p_n = (x_n, f(x_n)) \in epi(f)$. According to the update rule of Newton's method, we have

$$x_{n+1} \leftarrow x_n - \frac{f(x_n)}{s_n}$$

where $s_n \in \partial f(x_n)$. We will prove that, by firstly projecting p_n from C_2 to C_1 we obtain $(x_n, 0) = Proj_{C_1}(p_n)$, and then projecting $(x_n, 0)$ back to C_2 where $p_{n+1} = Proj_{C_2}((x_n, 0)) = (\hat{x}_{n+1}, f(\hat{x}_{n+1}))$, we have

$$x_{n+1} = \hat{x}_{n+1}$$

To begin with, it's quite easy to see why the projection of p_n to C_1 is $(x_n, 0)$. Then, a key observation is that the direction of $(x_n, 0) - p_{n+1}$ points to the same direction as the normal vector of the supporting hyperplane at p_{n+1}.

By denoting (a, r) as the normal vector associated with the supporting hyperplane at p_{n+1}, we know that there exists some $\lambda > 0$ such that $p_{n+1} + \lambda(a, r) = (x_n, 0)$. This results in the following system

$$\begin{cases} x_{n+1} + \lambda a & = x_n \\ f(x_{n+1}) + \lambda r & = 0 \end{cases}$$