Techniques in Optimization and Sampling (book in progress with improving citations)

Yin Tat Lee[1]
University of Washington & Microsoft Research

Santosh Vempala[2]
Georgia Tech & Vempala Research

April 15, 2021

[1]yintat@uw.edu. This work is supported in part by CCF-1749609, DMS-1839116, DMS-2023166, Microsoft Research Faculty Fellowship, Sloan Research Fellowship and Packard Fellowships.

[2]vempala@gatech.edu. This work is supported in part by CCF-1563838, E2CDA-1640081, CCF-1717349 and DMS-1839323.

Contents

[1] Anything marked with * means it is probably too mathematical and could be skipped.

Preliminaries

We use $B(x, r)$ to denote the Euclidean (or ℓ_2) ball centered at x with radius r: $\{y : \|y - x\|_2 \le r\}$. We use $\mathrm{conv}(X)$ to denote the convex hull of X, namely $\mathrm{conv}(X) = \{\sum \alpha_i x_i : \ \alpha_i \ge 0, \sum \alpha_i = 1, x_i \in X\}$. We use $\langle x, y\rangle$ to denote the ℓ_2 inner product $x^T y$ of x and y. For any two points $x, y \in \mathbb{R}^n$, we view them as column vectors, and use $[x, y]$ to denote $\mathrm{conv}(\{x, y\})$, namely, the line segment between x and y. Unless specified otherwise, $\|x\|$ will be the ℓ_2 norm $\|x\|_2$.

We use e_i to denote the coordinate vector with i-th coordinate is 1 and 0 otherwise.

Functions

Definition. For any $L \ge 0$, a function $f : V \to W$ is L-Lipschitz if $\|f(x) - f(y)\|_W \le L \|x - y\|_V$ where the norms $\|\cdot\|_V$ and $\|\cdot\|_W$ are ℓ_2 norms if unspecified.

Definition. A function $f \in \mathcal{C}^k(\mathbb{R}^n)$ if f is a k-differentiable function and its k^{th} derivative is continuous.

Theorem 0.0.1 (Taylor's Remainder Theorem)**.** *For any $g \in \mathcal{C}^{k+1}(\mathbb{R})$, and any x and y, there is a $\zeta \in [x, y]$ such that*

$$g(y) = \sum_{j=0}^{k} g^{(j)}(x) \frac{(y-x)^j}{j!} + g^{(k+1)}(\zeta) \frac{(y-x)^{k+1}}{(k+1)!}.$$

Linear Algebra

Definition 0.0.2. A real symmetric matrix A is positive semi-definite (PSD) if $x^\top A x \ge 0$ for all $x \in \mathbb{R}^n$. Equivalently, a real symmetric matrix with all nonnegative eigenvalues is PSD. We write $A \succeq B$ if $A - B$ is PSD.

Definition. For any matrix A, we define its trace, $\mathrm{tr}A = \sum A_{ii}$, Frobenius norm, $\|A\|_F^2 = \mathrm{tr}(A^\top A) = \sum_{i,j} A_{ij}^2$, and operator norm, $\|A\|_{\mathrm{op}} = \sup_{\|x\|_2 \le 1} \|Ax\|_2$.

For symmetric A, we have $\mathrm{tr}A = \sum_i \lambda_i$, $\|A\|_F^2 = \sum_i \lambda_i^2$ and $\|A\|_{\mathrm{op}} = \max_i |\lambda_i|$ where λ_i are the eigenvalues of A.

For a vector x, $\|x\|_A = \sqrt{x^T A x}$; for a matrix B,

$$\|B\|_A = \sup_x \frac{\|Bx\|_A}{\|x\|_A}.$$

Probability

Definition. The KL-divergence of a density ρ with respect to another density ν is

$$D_{\mathrm{KL}}(\rho \| v) = \int \rho(x) \log \frac{\rho(x)}{\nu(x)} dx.$$

Part I

Optimization

Chapter 1

Introduction

In this book[1], we will study two topics involving convexity, namely optimization and sampling. Given a multivariate, real-valued function f, (1) how fast can we find a point that minimizes f? (2) how fast can we sample a point according to the distribution with density defined by f, e.g., proportional to e^{-f}? These problems are quite closely connected, e.g., sampling from such distributions can be used to find near-optimal points. These problems are intractable in full generality, and have exponential (in dimension) complexity even under smoothness assumptions.

Convexity and its natural extensions are a current frontier of tractable, i.e., polynomial-time, computation. The assumption of convexity induces structure in instances that makes them amenable to efficient algorithms. For example, the local minimum of a convex function is a global minimum. Convexity is maintained by natural operations such as intersection (for sets) and addition (for functions). Perhaps less obvious, but also crucial, is that convex sets can be approximated by ellipsoids in various ways.

We will learn several techniques that lead us to some polynomial-time algorithms for both problems and (nearly) linear-time algorithms for the case f is close to a quadratic function.

Although convex optimization has been studied since the 19th century[2] with many tight results emerging, there are still many basic open problems. Here is an example:

Open Problem. Given an $n \times n$ random 0/1 matrix A with $O(n)$ nonzero entries, can we solve $Ax = b$ in $o(n^2)$ time?

Computing the volume is an ancient problem, the early Egyptians and Greeks developed formulas for specific shapes of interest. Unlike convex optimization, even computing the volume of a convex body is intractable, as we will see later. Nevertheless, there are efficient randomized algorithms that can estimate the volume of convex bodies to arbitrary accuracy in time polynomial in the dimension and the desired accuracy. This extends to efficient algorithms for integrating *logconcave* functions, i.e., functions of the form e^{-f} where f is convex. The core ingredient is sampling in high dimension. Sampling and volume computation will be the motivating problems for the second part of this book. Again, many basic problems remain open. To illustrate:

Open Problem. Given a polytope defined by $\{x : Ax \leq b\}$, can we estimate its volume to within a constant factor in nearly linear time?

1.1 Why non-convex functions can be difficult

Before discussing convex functions, we note that optimizing general functions can be difficult. Consider the function

$$f(x) = \begin{cases} 1 & \text{if } x \neq x^* \\ 0 & \text{if } x = x^* \end{cases}$$

and suppose that we can only access the function by computing its function value. This function f always returns 1 unless we know x^*. Hence, one can prove that it takes infinitely many calls to f to find x^*.

[1] In the UW course, only the optimization part is covered.

[2] Augustin-Louis Cauchy introduced gradient descent in 1847.

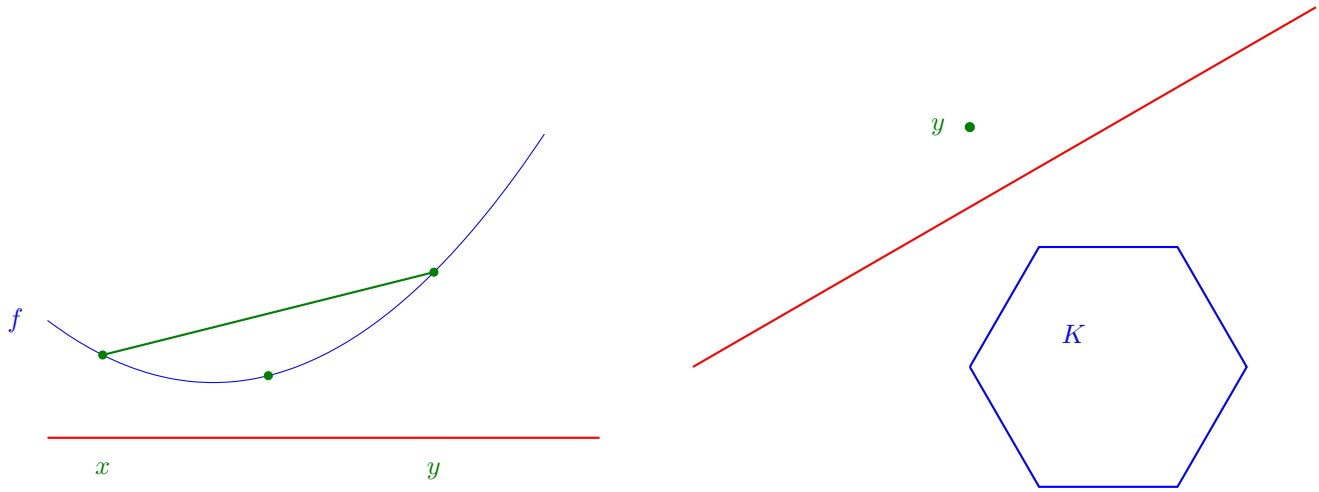


Figure 1.2.1: Convex function; Convex set

This function is difficult to optimize, and not merely because of its discontinuity. Similar functions can be constructed that are continuous. Consider the function $f : B(0_n, 1) \to \mathbb{R}$ defined by

$$f(x) = \min(\|x - x^*\|_2, \epsilon) \tag{1.1.1}$$

where $B(0_n, 1)$ is the unit ball centered at the origin, 0_n. This function is 1-Lipschitz and unless we query $f(x)$ with x that is ϵ-close to x^*, it will always return ϵ. Since the region where f is not ϵ has volume $\frac{1}{\epsilon^n}$ times the volume of the unit ball, one can show that it takes $\Omega\left(\frac{1}{\epsilon^n}\right)$ calls to f to find x^*. The following exercise asks you to prove that this bound is tight.

Exercise 1.1.1. Show that if f is 1-Lipschitz on $B(0_n, 1)$, we can find x such that $f(x) - \min_x f(x) \leq \epsilon$ by calling $f(x)$ at $O(\frac{1}{\epsilon})^n$ points.

Thus $O(1/\epsilon)^n$ is the best possible bound for optimizing general 1-Lipschitz functions. Similar constructions can also be made for infinitely differentiable functions. We note that it is easy to find local minima for all the functions above. In Section 2.2, we will show that it is easy to find an approximate local minimum of a continuously differentiable function.

1.2 Why is Convexity Useful? Linear Separability!

In the last section, we saw that general functions are difficult to optimize because we can only find the minimum via exhaustive search. Now we define convex sets, convex functions and convex problems. One benefit of convexity is that it enables binary search.

Definition 1.2.1. A set K in $\mathbb{R}^n$ is *convex* if for every pair of points $x, y \in K$, we have $[x, y] \subseteq K$.

Definition 1.2.2. A function $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is *convex* if for any $\lambda \in [0, 1]$, we have

$$f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda) f(y).$$

Definition 1.2.3. An optimization problem $\min_{x \in K} f(x)$ is convex if K and f are convex.

Any point not in a convex set can be separated by a hyperplane from the set. We will see in Chapter 3 that separating hyperplanes allow us to do binary search to find a point in a convex set. This is the basis of all polynomial-time algorithms for optimizing general convex functions. We will explore these binary search algorithms in Chapter 3.

Theorem 1.2.4 (Hyperplane separation theorem)**.** *Let K be a closed convex set in $\mathbb{R}^n$ and $y \notin K$. There is a non-zero $\theta \in \mathbb{R}^n$ such that*

$$\langle \theta, y \rangle > \max_{x \in K} \langle \theta, x \rangle .$$

Proof. Let x^* be a point in K closest to y, namely $x^* \in \arg\min_{x\in K} \|x-y\|_2^2$ (such a minimizer always exists for closed convex sets; this is sometimes called Hilbert's projection theorem). Using convexity of K, for any $x \in K$ and any $0 \le t \le 1$, we have that $(1-t)x^* + tx \in K$ and hence

$$\|y-(1-t)x^*-tx\|_2^2 \ge \min_{x\in K} \|y-x\|_2^2 = \|y-x^*\|_2^2 .$$

Expanding the LHS, we have

$$\begin{aligned}\|y-(1-t)x^*-tx\|_2^2 &= \|y-x^*+t(x^*-x)\|_2^2 \\ &= \|y-x^*\|_2^2 + 2t\langle y-x^*, x^*-x\rangle + t^2\|x^*-x\|_2^2 .\end{aligned}$$

Canceling the term $\|x^*-y\|_2^2$ and dividing both sides by t,

$$2\langle y-x^*, x^*-x\rangle + t\|x^*-x\|_2^2 \ge 0.$$

Taking $t \to 0^+$, we have that

$$\langle y-x^*, x^*-x\rangle \ge 0 \text{ for all } x \in K. \tag{1.2.1}$$

Taking $\theta = y-x^*$ and using (1.2.1), for all $x \in K$, we have that

$$\begin{aligned}\langle \theta, y-x\rangle &= \langle \theta, y-x^*\rangle + \langle y-x^*, x^*-x\rangle \\ &= \|\theta\|^2 + \langle y-x^*, x^*-x\rangle \\ &> 0\end{aligned}$$

where we used that $y \notin K$ and hence $\|\theta\|^2 > 0$. □

Theorem 1.2.4 shows that a polytope (a finite intersection of halfspaces) is essentially as general as a convex set.

Corollary 1.2.5. *Any closed convex set K can be written as the intersection of halfspaces as follows*

$$K = \bigcap_{\theta\in\mathbb{R}^n} \left\{ x : \ \langle \theta, x\rangle \le \max_{y\in K} \langle \theta, y\rangle \right\}.$$

In other words, any convex set is a limit of a sequence of polyhedra.

Proof. Let $L \stackrel{\text{def}}{=} \bigcap_{\theta\in\mathbb{R}^n} \{x : \ \langle \theta, x\rangle \le \max_{y\in K}\langle \theta, y\rangle\}$. Since $K \subset \{x : \ \langle \theta, x\rangle \le \max_{y\in K}\langle \theta, y\rangle\}$, we have $K \subset L$.

For any $x \notin K$, Theorem 1.2.4 shows that there is a θ such that $\theta^\top x > \max_{y\in K} \theta^\top y$. Hence, we have $x \notin L$ and hence $L \subset K$. □

This shows that convex optimization is related to linear programs (optimize linear functions over polytopes) as follows:

$$\min_{x\in K} f(x) = \min_{(x,y)\in\{x\in K, y\in\mathbb{R}: y\ge f(x)\}} y$$

where the set $\{x \in K, y \in \mathbb{R} : y \ge f(x)\}$ then can be approximated by intersection of halfspaces, namely $\{A\binom{x}{y} \le b\}$ for some matrix $A \in \mathbb{R}^{m\times(n+1)}$ and vector $b \in \mathbb{R}^m$ with $m \to +\infty$.

Similar to convex sets, we have a separation theorem similar to Theorem 1.2.4 for convex functions. This shows that one can use binary search to find minimum of convex functions. (See Chapter 3.)

Theorem 1.2.6. *Let $f \in \mathcal{C}^1(\mathbb{R}^n)$ be convex. Then,*

$$f(y) \ge f(x) + \nabla f(x)^\top (y-x) \text{ for all } x, y \in \mathbb{R}^n \tag{1.2.2}$$

Proof. Fix any $x, y \in \mathbb{R}^n$. Let $g(t) = f((1-t)x+ty)$. Since f is convex, so is g over $[0,1]$. Then, we have

$$g(t) \le (1-t)g(0) + tg(1)$$

which implies that

$$g(1) \ge g(0) + \frac{g(t)-g(0)}{t}.$$

Taking $t \to 0^+$, we have that $g(1) \geq g(0) + g'(0)$. In other words, using the chain rule for derivatives, we have $g'(0) = \langle \nabla f(x), y - x \rangle$ and hence

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

□

This theorem shows that $\nabla f(x) = 0$ (local minimum) implies x is a global minimum.

Theorem 1.2.7 (Optimality condition for unconstrained problems)**.** *Let $f \in \mathcal{C}^1(\mathbb{R}^n)$ be convex. Then, $x \in \mathbb{R}^n$ is a minimizer of $f(x)$ if and only if $\nabla f(x) = 0$.*

Proof. If $\nabla f(x) \neq 0$, then

$$f(x - \varepsilon \nabla f(x)) = f(x) - \varepsilon \left\| \nabla f(x) \right\|_2^2 + O(\varepsilon^2) \left\| \nabla f(x) \right\|_2^2 < f(x)$$

for small enough ε. Hence, such a point cannot be the minimizer.

On the other hand, if $\nabla f(x) = 0$, Theorem 1.2.6 shows that

$$f(y) \geq f(x) + \nabla f(x)^\top (y - x) = f(x) \text{ for all } y.$$

□

We note that the proof above is in fact a constructive proof. If x is not a minimum, it suggests a point that has better function value. This will be the discussion of an upcoming section. For continuous convex functions, there is a weaker notion of gradient called sub-differential, which is a set instead of a vector. Both theorems above holds with gradients replaced by sub-differentials.

1.3 Convex Problems are Everywhere!

In this section, we give a few examples of convex problems to illustrate the wide applicability of convex optimization.

Minimum Cost Flow Problem (Computer Science)

The min cost flow problem has lots of applications such as route planning, airline scheduling, image segmentation, recommendation systems, etc. In this problem, we are given a graph $G = (V, E)$ with $m \stackrel{\text{def}}{=} |E|$ edges and $n \stackrel{\text{def}}{=} |V|$ vertices. Each edge $e \in E$ has capacity $u_e > 0$ and cost c_e. The problem is to minimize the total cost of sending d amount of flow from a source vertex $s \in V$ to a sink vertex $t \in V$. Formally, the problem can be written as an optimization problem $\min_{f \in \mathbb{R}^{|E|}} \sum_{e \in E} c_e \cdot f_e$ subject to the constraints:

- Capacity constraints: $0 \leq f_e \leq u_e$ for all $e \in E$.
- Flow conservation: $\sum_{e \text{ enters } v} f_e = \sum_{e \text{ leaves } v} f_e$ for all $v \in V \setminus \{s, t\}$.
- Demand: $\sum_{e \text{ enters } t} f_e = \sum_{e \text{ leaves } s} f_e = d$.

To check this is a convex problem, we note that the objective function is a linear function $c^\top f$ which is convex. The domain is the intersection of three sets (the three set of equations above). The first set is a scaled hypercube, the second and last set is a linear subspace. All of them are convex and so is their intersection. Therefore, this is a convex problem.

Linear Programs (Operation Research/Economics)

Consider the diet problem: find a cheapest diet plan that satisfies the nutrients requirements. Formally, suppose there are n different foods and m different nutrients. The food i has unit cost c_i, a_{1i} unit of nutrient 1, a_{2i} unit of nutrient 2, $\cdots$. Furthermore, every day we need b_j unit of the nutrient j. Then, the problem is simply find an assignment x such that

$$\min_{x \geq 0, Ax \geq b} c^\top x$$

where $c \in \mathbb{R}^n$ is the cost vector, $b \in \mathbb{R}^m$ is the intake requirement vector and $A \in \mathbb{R}^{m \times n}$ is the matrix of nutrients contents of each food. Unfortunately, Nobel Laureate Stigler showed that [58] the optimal meal is evaporated milk, cabbage, dried navy beans, and beef liver.

Joking aside, both the diet problem and the minimum cost flow problem can be reformulated into the form

$$\min_{Ax=b, x \geq 0} c^\top x \tag{1.3.1}$$

for some vectors c, b and some matrix A. These problems are called linear programs and have many applications in resource allocation. Special cases of linear programs are also of great interest; for example the diet problem is a packing/covering LP.

Exercise 1.3.1. Show that both the minimum cost flow problem and the diet problem can be written as (1.3.1). Also, show that (1.3.1) is a convex problem.

Logistic Regression (Machine Learning)

Consider the problem of predicting the likelihood of getting diabetes in the future. Suppose we have collected many examples $\{(x_i, y_i)\}_{i=1}^n$ where $x_i \in \mathbb{R}^d$ represents the features of a person and $y_i \in \{\pm 1\}$ represents whether that person gets diabetes. For example, the feature vector can be (age, weight, height, BMI, fasting glucose level, ...). The features in the vector may be redundant and the purpose of extra variables is to make linear functions expressive enough to be able to classify. In particular, we assume that there is a vector θ such that for most i, we have $\langle x^i, \theta \rangle < 0$ if $y_i = 1$ and $\langle x^i, \theta \rangle > 0$ if $y_i = -1$. The error of the vector θ is

$$\frac{1}{n} \sum_{i=1}^n 1_{y_i \langle x^i, \theta \rangle > 0}.$$

This function is not convex in θ. More generally, one considers the objective function (to be minimized over θ)

$$R(\theta) = \frac{1}{n} \sum_{i=1}^n f(y_i \langle x^i, \theta \rangle) + \lambda \|\theta\|_1 \tag{1.3.2}$$

where f is some function such that $f(z)$ is large when z is positive and large and $f(z) = 0$ when z is highly negative, and $\lambda \|\theta\|_1$ is a *regularization* term to make sure θ is bounded. One popular function is $f(z) = \log(1 + e^z)$, and this problem is called logistic regression. In the section 1.5, we prove that the function (1.3.2) is indeed convex when $f(z) = \log(1 + e^z)$.

Minimal Surface (Physics)

A surface $M \subset \mathbb{R}^3$ is a minimal surface if it has the minimum surface area among all surfaces with the same boundary. These surfaces appear naturally and has been studied extensively not just for $\mathbb{R}^3$ but also for different manifold. For simplicity, we consider the case that the surface is parameterized by

$$M = \{(x, y, f(x, y)) : 0 \leq x \leq 1, 0 \leq y \leq 1\}.$$

In this case, f is a minimal surface if

$$f = \text{argmin}_{\text{feasible } g} \text{SurfaceArea}(g)$$

where we say g is feasible if $g(x, y) = f(x, y)$ for all x, y on the boundary of $[0, 1]^2$. One natural question (called Plateau's problem) is to find a minimal surface with a given boundary. For this particular case we consider, we can

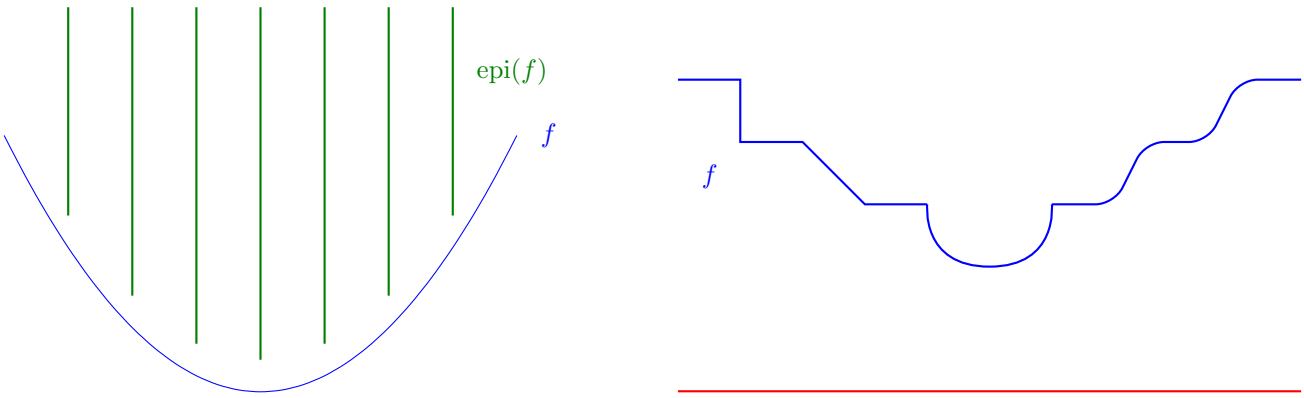


Figure 1.4.1: Epigraph of f; quasiconvex function

simply use convex optimization. Note that the constraint (g is feasible) is exactly a linear subspace on the space of functions on $[0,1]^2$. Furthermore, the objective is convex by using the fact that

$$\text{SurfaceArea}(f) = \int_0^1 \int_0^1 \sqrt{(\frac{\partial f(x,y)}{\partial x})^2 + (\frac{\partial f(x,y)}{\partial y})^2 + 1} dx dy.$$

Exercise 1.3.2. Show that surface area is convex by using the definition.

Calculus of variations is an area in mathematics studies the optimization on function spaces and there are many common theorems between this and convex optimization.

1.4 Examples of Convex Sets and Functions

There are many important convex sets and here we only list some that appear in this course. One of the most important classes of convex functions comes from convex sets.

Definition 1.4.1. For a convex set K, we define the indicator function of K by

$$\delta_K(x) = \begin{cases} 0 & \text{if } x \in K \\ +\infty & \text{otherwise} \end{cases}.$$

We can also construct convex sets by convex functions. We let the domain $\text{dom} f \stackrel{\text{def}}{=} \{x \in \mathbb{R}^n : \ f(x) < +\infty\}$. The definition of a convex function shows that $\text{dom} f$ is a convex set if f is a convex function. Alternatively, by looking at the set of points above the graph of the function, we obtain a convex set called an epigraph.

Definition 1.4.2. The *epigraph* of $f : \mathbb{R}^n \to \mathbb{R}$ is $\text{epi} f \stackrel{\text{def}}{=} \{(x,t) \subset \mathbb{R}^n \times \mathbb{R} : \ t \geq f(x)\}$.

A function f is convex if and only if $\text{epi} f$ is a convex set.

This characterization shows that $\min_x f(x)$ is the same as $\min_{(t,x) \in \text{epi} f} t$. Therefore, convex optimization is the same as optimizing a linear function over a convex set. Another important feature of a convex set is the following.

Fact 1.4.3. *Any level set $\{x \in \mathbb{R}^n : \ f(x) \leq t\}$ of a convex function f is convex.*

In particular, this shows that the set of minimizers is connected. Therefore, any local minimum is a global minimum. We note that the converse of the fact above is not true. A function is *quasiconvex* if every level set is convex. For example, Fig. 4.1.1 shows a function that is quasiconvex but not convex.

Finally, we note that many operations preserve convexity. Here is an example.

Exercise 1.4.4. Given a matrix A, a vector b, positive scalars $t_1, t_2 \geq 0$, convex functions f_1 and f_2, then $g(x) = t_1 f_1(Ax + b) + t_2 f_2(x)$ is convex.

Here are some convex sets and functions. In Section 1.5, we illustrate how to check convexity.

Example. Convex sets: polyhedron $\{x : Ax \le b\}$, polytope $\mathrm{conv}(\{v_1, \dots, v_m\})$ with $v_1, \dots, v_m \in \mathbb{R}^n$, ellipsoid $\{x : x^\top Ax \le 1\}$ with $A \succeq 0$, positive semidefinite cone $\{X \in \mathbb{R}^{n\times n} : X \succeq 0\}$, norm ball $\{x : \|x\|_p \le 1\}$ for all $p \ge 1$.

Example. Convex functions: x, $\max(x, 0)$, e^x, $|x|^a$ for $a \ge 1$, $-\log(x)$, $x \log x$, $\|x\|_p$ for $p \ge 1$, $(x, y) \to \frac{x^2}{y}$ (for $y > 0$), $A \to -\log\det A$ over PSD matrices A, $(x, Y) \to x^\top Y^{-1} x$ (for $Y \succ 0$), $\log \sum_i e^{x_i}$, $(\prod_i^n x_i)^{\frac{1}{n}}$.

Exercise 1.4.5. Show that the above sets and functions are all convex.

1.5 Checking Convexity

It is often cumbersome to check if a function is convex via (1.2.2). The major benefit of that definition is that it works for non-differentiable functions and infinite dimensional space. However, when a function is twice differentiable, one can check the convexity by simply checking if the Hessian is positive semi-definite. (Recall the definition of $A \succeq 0$ in 0.0.2.)

To show this, we first prove the following second-order Taylor theorem.

Lemma 1.5.1. *For any $f \in \mathcal{C}^2(\mathbb{R}^n)$, and any $x, y \in \mathbb{R}^n$, there is a $z \in [x, y]$ s.t.*

$$f(y) = f(x) + \nabla f(x)^\top (y - x) + \frac{1}{2}(y - x)^\top \nabla^2 f(z)(y - x).$$

Proof. Let $g(t) = f((1 - t)x + ty)$. Taylor expansion (Theorem 0.0.1) shows that

$$g(1) = g(0) + g'(0) + \frac{1}{2}g''(\zeta)$$

where $\zeta \in [0, 1]$. To see the result, note that $g(0) = f(x)$, $g'(0) = \nabla f(x)^\top (y - x)$ and $g''(\zeta) = (y - x)^\top \nabla^2 f((1 - \zeta)x + \zeta y)(y - x)$. □

Now, we show that f is convex if and only if $\nabla^2 f(x) \succeq 0$ for all x.

Theorem 1.5.2. *Let $f \in \mathcal{C}^2(\mathbb{R}^n)$. Then, the following are equivalent:*

1. *f is convex.*
2. *$f(y) \ge f(x) + \nabla f(x)^\top (y - x)$ for all $x, y \in \mathbb{R}^n$.*
3. *$\nabla^2 f(x) \succeq 0$ for all $x \in \mathbb{R}^n$.*

Proof. We have proved (1) implies (2) in Theorem 1.2.6.

Suppose (2) holds. Then, for any $x, h \in \mathbb{R}^n$

$$f(x + th) \ge f(x) + t\nabla f(x)^\top h.$$

By Taylor expansion (Lemma 1.5.1), we have that

$$f(x + th) = f(x) + t\nabla f(x)^\top h + \frac{t^2}{2} h^\top \nabla^2 f(z) h$$

where $z \in [x, x + th]$. By comparing two equations, we have that $h^\top \nabla^2 f(z) h \ge 0$. Taking $t \to 0$, we have $z \to x$ and hence $\nabla^2 f(z) \to \nabla^2 f(x)$. Therefore, we have that

$$h^\top \nabla^2 f(x) h \ge 0$$

for all x and h. Hence, this gives (3).

Suppose (3) holds. Fix $x, y \in \mathbb{R}^n$. Consider the function

$$g(\lambda) = f(\lambda x + (1-\lambda)y) - \lambda f(x) - (1-\lambda)f(y).$$

Consider $\lambda^* = \operatorname{argmax}_{\lambda\in[0,1]} g(\lambda)$. If λ^* is either 0 or 1, then we have $g(\lambda^*) = 0$. Otherwise, by Taylor's theorem, there is a $\zeta \in [\lambda^*, 1]$ such that

$$\begin{aligned} g(1) &= g(\lambda^*) + g'(\lambda^*)(1-\lambda^*) + \frac{1}{2}g''(\zeta)(1-\lambda^*)^2 \\ &= g(\lambda^*) + \frac{1}{2}g''(\zeta)(1-\lambda^*)^2 \end{aligned}$$

where we used that $g'(\lambda^*) = 0$. Note that

$$\begin{aligned} g'(\zeta) &= \nabla f(\zeta x + (1-\zeta)y)^\top (x-y) - f(x) + f(y), \\ g''(\zeta) &= (x-y)^\top \nabla^2 f(\zeta x + (1-\zeta)y)(x-y). \end{aligned}$$

By the assumption (3), we have that $g''(\zeta) \geq 0$ and hence $0 = g(1) \geq g(\lambda^*)$. Hence, in both cases, $\max_{\lambda\in[0,1]} g(\lambda) = g(\lambda^*) \leq 0$. This gives (1). □

Now, as an example, we prove that the function (1.3.2) is convex.

Example 1.5.3. The function (1.3.2) is convex for $f(z) = \log(1+\exp(z))$.

Proof. We write $R(\theta) = R_1(\theta) + R_2(\theta)$ where $R_1(\theta) = \frac{1}{n}\sum_{i=1}^n f(y_i \cdot \langle x^i, \theta\rangle)$ and $R_2(\theta) = \lambda\|\theta\|_1$. It is easy to check R_2 is convex. So, it suffices to prove R_1 is convex. Now, we use Theorem 1.5.2 to prove that R_1 is convex. Note that

$$\begin{aligned} \nabla R_1(\theta) &= \frac{1}{n}\sum_{i=1}^n f'(\langle x^i, \theta\rangle)x^i, \\ \nabla^2 R_1(\theta) &= \frac{1}{n}\sum_{i=1}^n f''(\langle x^i, \theta\rangle)x^i(x^i)^\top. \end{aligned}$$

Since $x^i(x^i)^\top \succeq 0$, it suffices to prove that $f''(\langle x^i, \theta\rangle) \geq 0$. This follows from the calculation: $f'(z) = \frac{\exp(z)}{1+\exp(z)} = 1 - \frac{1}{1+\exp(z)}$ and $f''(z) = \frac{\exp(z)}{(1+\exp(z))^2} \geq 0$. □

1.6 Logconcave Functions

Definition 1.6.1. A function $f : \mathbb{R}^n \to \mathbb{R}_+$ is *logconcave* if $\log f$ is concave, i.e., f is nonnegative and for any $\lambda \in [0,1]$, we have $f(\lambda x + (1-\lambda)y) \geq f(x)^\lambda f(y)^{1-\lambda}$.

Example 1.6.2. The indicator function of a convex set $1_K(x) = \begin{cases} 1 & \text{if } x \in K \\ 0 & \text{otherwise} \end{cases}$ is logconcave.

Lemma 1.6.3 (Dinghas; Prékopa; Leindler)**.** *The product, minimum and convolution of two logconcave functions is also logconcave; in particular, any linear transformation or any marginal of a logconcave density is logconcave; the distribution function of any logconcave density is logconcave.*

We next describe the basic theorem underlying the above properties. We will see their proofs in a later chapter.

Theorem 1.6.4 (Prékopa-Leindler)**.** *Fix $\lambda \in [0,1]$. Let $f, g, h : \mathbb{R}^n \to \mathbb{R}_+$ be functions satisfying $h(\lambda x + (1-\lambda)y) \geq f(x)^\lambda g(x)^{1-\lambda}$ for all $x \in \mathbb{R}^n$. Then,*

$$\int_{\mathbb{R}^n} h \geq \left(\int_{\mathbb{R}^n} f\right)^\lambda \left(\int_{\mathbb{R}^n} g\right)^{1-\lambda}.$$

An equivalent version of the lemma for sets in $\mathbb{R}^n$ is often useful. By a *measurable* set below, we mean Lebesgue measurable, which coincides with the definition of volume (for an axis aligned box, it is the product of the axis lengths; for any other set, it is the limit over increasingly finer partitions into boxes, of the sum of volumes of boxes that intersect the set).

Theorem 1.6.5 (Brunn-Minkowski)**.** *For any $\lambda \in [0,1]$ and measurable sets $A, B \subset \mathbb{R}^n$, we have*

$$\text{vol}(\lambda A + (1-\lambda)B)^{1/n} \geq \lambda \text{vol}(A)^{1/n} + (1-\lambda)\text{vol}(B)^{1/n}.$$

An immediate consequence of the first theorem above is that any one-dimensional marginal distribution of a convex body is logconcave; the second implies that it is in fact $(1/(n-1))$-concave (a function is s-concave if f^s is concave) if the body is in $\mathbb{R}^n$.

Exercise 1.6.6. Prove both corollaries just mentioned.

Example 1.6.7. We give one example problem related to Bayesian inference. Suppose we have a signal $\theta = (\theta_1, \theta_2, \cdots, \theta_n)$ and that we can take a measurement y_i of θ_i. The measurement only incurs unbiased Gaussian noise, i.e., $y_i = \theta_i + \epsilon_i$ where $\epsilon_i \sim N(0,1)$. The question is to recover the signal θ using y. Without any prior on θ, the only sensible recovery is $\theta = y$. With a prior, one can apply Bayes' theorem:

$$\mathbf{P}(\theta|y) = \frac{\mathbf{P}(y|\theta)\mathbf{P}(\theta)}{\mathbf{P}(y)}.$$

The Bayesian choice is to find θ with maximum likelihood, namely $\theta = \text{argmax}_\theta \log \mathbf{P}(\theta|y) = \text{argmin}_\theta - \log \mathbf{P}(\theta|y)$.

Using the noise assumption, we have that

$$\mathbf{P}(y|\theta) \propto \exp(-\frac{1}{2}\sum_i (y_i - \theta_i)^2).$$

Now, say we know the signal is smooth and we model the prior as $\mathbf{P}(\theta) \propto \exp(-\lambda \sum_i (\theta_i - \theta_{i+1})^2)$ where λ controls how smooth the signal is. Hence,

$$-\log \mathbf{P}(\theta|y) = c + \sum_i (y_i - \theta_i)^2 + \lambda \sum_i (\theta_i - \theta_{i+1})^2.$$

Since each term in the function above is convex, so is the whole formula. Hence, the recovery question becomes a convex optimization problem

$$\min_\theta \sum_i (y_i - \theta_i)^2 + \lambda \sum_i (\theta_i - \theta_{i+1})^2.$$

When we recover a signal, we want to know how confident we are because there are many choices of θ that could explain the same measurement y. One way to do this is to sample multiple $\theta \propto \mathbf{P}(\theta|y)$ and compute the empirical variance or other statistics. Note that

$$\mathbf{P}(\theta|y) \propto e^{-\sum_i (y_i - \theta_i)^2 - \lambda \sum_i (\theta_i - \theta_{i+1})^2}$$

which is a logconcave distribution. Therefore, one can study the signal and quality of signal recovery via logconcave sampling.

Chapter 2

Gradient Descent

2.1 Philosophy

Often, optimization methods follow the following framework:

Algorithm 1: `OptimizationFramework`

for $k = 0, 1, \cdots$ **do**
  Approximate f by a simpler function f_k according to the current point $x^{(k)}$
  Do something using f_k (such as set $x^{(k+1)} = \arg\min_x f_k(x)$)
end

The runtime depends on the number of iterations and the cost per iteration. Philosophically, difficulties of a problem can never be created nor destroyed, only converted from one form of difficulty to another. When we decrease the number of iterations, the cost per iteration often increase. The gain of new methods often come from avoiding some wasted computation, utilizing some forgotten information or giving a faster but tailored algorithm for a sub-problem.

One key question to answer in designing an optimization algorithm is that what the problem looks like (or how can we approximate f by a simpler function). Here are some approximation we will used in this textbook:

- First-order Approximation: $f(y) \approx f(x) + \langle \nabla f(x), y - x \rangle$ (Section 2.2)
- Second-order Approximation: $f(y) \approx f(x) + \langle \nabla f(x), y - x \rangle + (y - x)^\top \nabla^2 f(x)(y - x)$ (Section 5.3)
- Stochastic Approximation: $\sum_i f_i(x) \approx f_j(x)$ for a random j (Section 6.3)
- Matrix Approximation: Approximate A by a simpler B with $\frac{1}{2}A \preceq B \preceq 2A$ (Section 6.1 and Section 6.2)
- Set Approximation: Approximate a convex set by an ellipsoid or a polytope (Section 3.1)
- Barrier Approximation: Approximate a convex set by a smooth function that blows up on the boundary (Section 5.4)
- Polynomial Approximation: Approximate a function by a polynomial (Section 7.1)
- Partial Approximation: Split the problem into two parts and approximate only one part

Here are other approximation not covered:

- Taylor Approximation: $f(y) \approx \sum_{k=0}^{K} D^k f(x)[y - x]^k$
- Mixed ℓ^2-ℓ^p Approximation: $f(y) \approx f(x) + \langle \nabla f(x), y - x \rangle + \sum_{i=1}^{n} \alpha_i (y_i - x_i)^2 + \beta_i (y_i - x_i)^p$
- Stochastic Matrix Approximation: Approximate A by a simpler random B with $B \preceq 2A$ and $\mathbf{E}B \succeq \frac{1}{2}A$
- Homotopy Method: Approximate a function by a family of functions

- ...(Please give me more examples here)...

The second question to answer is how we maintain all the different approximations we created in each step. One simple way would be forget the approximation we got in previous steps, but this is often not optimal. Another way is to keep all previous approximations/information (such as Section 3.1). Often the best way will be combining previous and current approximation carefully to a better approximation (such as Section 7.4).

2.2 Basic Algorithm

Perhaps the most natural algorithm for optimization is gradient descent. In fact it has many variants with different guarantees. Assume that the function f to be optimized is continuously differentiable. By basic calculus, either the minimum (or point achieving the minimum) is unbounded or the gradient is zero at a minimum. So we try to find a point with gradient close to zero (which, of course, does not guarantee global optimality). The basic algorithm is the following:

Algorithm 2: `GradientDescent` (GD)

Input: Initial point $x^{(0)} \in \mathbb{R}^n$, step size $h > 0$.
for $k = 0, 1, \cdots$ **do**
  if $\|\nabla f(x^{(k)})\|_2 \le \epsilon$ **then return** $x^{(k)}$
  `// Alternatively, one can use` $x^{(k+1)} \leftarrow \texttt{argmin}_{x=x^{(k)}+t\nabla f(x^{(k)})} f(x)$.
  $x^{(k+1)} \leftarrow x^{(k)} - h \cdot \nabla f(x^{(k)})$.
end

One can view gradient descent as a greedy method for solving $\min_{x\in\mathbb{R}^n} f(x)$. At a point x, gradient descent goes to the minimizer of

$$\min_{\|\Delta\|_2 \le \eta} f(x) + \nabla f(x)^\top \Delta.$$

The term $f(x) + \nabla f(x)^\top \Delta$ is simply the first-order approximation of $f(x + \Delta)$. Note that in this problem, the current point x is fixed and we are optimizing the step Δ. Certainly, there is no inherent reason for using first-order approximation and the Euclidean norm $\|x\|_2$. For example, if you use second-order approximation, then you would get a method involving Hessian of f.

The step size of the algorithm usually either uses a fixed constant, or follows a predetermined schedule, or determined using a line search.

If the iteration stops, we get a point with $\|\nabla f(x)\|_2 \le \epsilon$. Why is this good? The hope is that x is a near-minimum in the neighborhood of x. However, this might not be true if the gradient can fluctuate wildly:

Definition 2.2.1. We call f has L-Lipschitz gradient if ∇f is L-Lipschitz, namely, $\|\nabla f(x) - \nabla f(y)\|_2 \le L\|x - y\|_2$ for all x, y.

Similar to Theorem 1.5.2, we have the following equivalent:

Theorem 2.2.2. *Let $f \in \mathcal{C}^2(\mathbb{R}^n)$. For any $\mu \ge 0$, the following are equivalent:*

1. $\|\nabla f(x) - \nabla f(y)\|_2 \le L\|x - y\|_2$ *for all* $x, y \in \mathbb{R}^n$.
2. $-L \preceq \nabla^2 f(x) \preceq L$ *for all* $x \in \mathbb{R}^n$.
3. $f(y) = f(x) + \nabla f(x)^\top (y - x) \pm \frac{L}{2}\|y - x\|_2^2$ *for all* $x, y \in \mathbb{R}^n$.

Proof. Suppose (1) holds. By the definition of $\nabla^2 f$, we have

$$\nabla^2 f(x)v = \lim_{h\to 0} \frac{\nabla f(x + hv) - \nabla f(x)}{h}.$$

Since $\|\frac{\nabla f(x+hv)-\nabla f(x)}{h}\|_2 \le \frac{L}{h}\|hv\|_2 = L\|v\|_2$, we have $\|\nabla^2 f(x)v\|_2 \le L\|v\|_2$. This proves (2).

Suppose (2) holds. Since $\nabla f(x) - \nabla f(y) = \int_0^1 \nabla^2 f(y + t(x-y))(x-y)dt$, we have that

$$\|\nabla f(x) - \nabla f(y)\|_2 \le \int_0^1 \|\nabla^2 f(y + t(x-y))\|_{\text{op}} \|x-y\|_2 dt \le L\|x-y\|_2.$$

This gives (1).

Suppose (2) holds. By Taylor expansion, we have

$$f(y) = f(x) + \nabla f(x)^\top (y-x) + \int_0^1 (1-t)(y-x)^\top \nabla^2 f(x + t(y-x))(y-x)dt.$$

Since $-L \preceq \nabla^2 f(x) \preceq L$, we have

$$(y-x)^\top \nabla^2 f(x + t(y-x))(y-x) = \pm L\|y-x\|^2.$$

Using this gives (3).

Suppose (3) holds, then $g(x) = f(x) + \frac{L}{2}\|x\|^2$ satisfies $g(y) \ge g(x) + \nabla g(x)^\top (y-x)$ for all $x, y \in \mathbb{R}^n$. Theorem 1.5.2 shows that g is convex and $\nabla^2 g(x) \succeq 0$. This shows that $\nabla^2 f(x) \succeq -L$. Similarly, by taking $g(x) = \frac{L}{2}\|x\|^2 - f(x)$, we have $\nabla^2 f(x) \preceq L$. Hence, this gives (2). □

With this equivalent definition, we can have an alternative view of gradient descent. Each step, we perform

$$x^{(k+1)} = \arg\min_y f(x^{(k)}) + \left\langle \nabla f(x^{(k)}), y - x^{(k)} \right\rangle + \frac{L}{2}\|y - x^{(k)}\|_2^2.$$

To see this is the same step, we let $g(y) = f(x^{(k)}) + \left\langle \nabla f(x^{(k)}), y - x^{(k)} \right\rangle + \frac{L}{2}\|y - x^{(k)}\|_2^2$. The optimality condition shows that $0 = \nabla g(x^{(k+1)}) = \nabla f(x^{(k)}) + L(x^{(k+1)} - x^{(k)})$. Hence, this gives the step $x^{(k+1)} = x^{(k)} - \frac{1}{L}\nabla f(x^{(k)})$.

By Theorem 2.2.2, we know that g is an upper bound of f, namely $g(y) \ge f(y)$ for all y. In general, many optimization methods involves minimizing some upper bound function every step. Note that the progress we made for f is at least the progress we made for g. If g is exactly f, we can get all the progress we can make in one step. Hence, we should believe if g is a better approximation of f, then we are making more progress. For gradient descent, it uses the simplest first-order approximation. Although this is not the best approximation one can come up, but it is robust enough to use in all sort of applications.

Analysis for general functions

Gradient descent works for both convex and non-convex functions. For non-convex function, we can only find a point with small gradient (called an approximate saddle point).

Theorem 2.2.3. *Let f be a function with L-Lipschitz gradient and x^* be any minimizer of f. The* `GradientDescent` *with step size $h = \frac{1}{L}$ outputs a point x such that $\|\nabla f(x)\|_2 \le \epsilon$ in $\frac{2L}{\epsilon^2}\left(f(x^{(0)}) - f(x^*)\right)$ iterations.*

The proof idea involves showing the function value $f(x)$ decreases by at least $\frac{\epsilon^2}{2L}$ when $\|\nabla f(x)\|_2 \ge \epsilon$. Since the function value can only decrease by at most $f(x^{(0)}) - f(x^*)$, this bounds the number of iterations.

Lemma 2.2.4. *For any $f \in \mathcal{C}^2(\mathbb{R}^n)$ with L-Lipschitz gradient, we have*

$$f(x - \frac{1}{L}\nabla f(x)) \le f(x) - \frac{1}{2L}\|\nabla f(x)\|_2^2.$$

Proof. Lemma 1.5.1 shows that

$$f(x - \frac{1}{L}\nabla f(x)) = f(x) - \frac{1}{L}\|\nabla f(x)\|_2^2 + \frac{1}{2L^2}\nabla f(x)^\top \nabla^2 f(z) \nabla f(x)$$

for some $z \in [x, x - \frac{1}{L}\nabla f(x)]$. Since $\|\nabla^2 f(x)\|_{\text{op}} \le L$, we have that $\nabla f(x)^\top \nabla^2 f(z) \nabla f(x) \le L \cdot \|\nabla f(x)\|_2^2$. Hence, we have the result. □

Proof of Theorem 2.2.3. Since each step of gradient descent decreases f by at least $\frac{\epsilon^2}{2L}$ and since we can decrease f by at most $f(x^{(0)}) - f^*$, we have the result. □

Despite the simplicity of the algorithm and the proof, it is known that this is the best one can do via any algorithm for this general setting [12].

2.3 Analysis for convex functions

Assuming the function is convex, we can prove that gradient descent in fact converges to the global minimum. In particular, when $\|\nabla f(x)\|_2$ is small, convexity shows that $f(x) - f^*$ is small (Theorem 1.2.6).

Lemma 2.3.1. *For any convex $f \in \mathcal{C}^1(\mathbb{R}^n)$, we have that $f(x) - f(y) \le \|\nabla f(x)\|_2 \cdot \|x - y\|_2$ for all x, y.*

Proof. Theorem 1.2.6 and Cauchy-Schwarz inequality shows that

$$f(x) - f(y) \le \langle \nabla f(x), x - y \rangle \le \|\nabla f(x)\|_2 \cdot \|x - y\|_2.$$

□

This in turns give a better bound on the number of iterations because the bound in Theorem 2.2.3 is affected by $f(x) - f^*$.

Theorem 2.3.2. *Let $f \in \mathcal{C}^2(\mathbb{R}^n)$ be convex with L-Lipschitz gradient and x^* be any minimizer of f. With step size $h = \frac{1}{L}$, the sequence $x^{(k)}$ in* `GradientDescent` *satisfies*

$$f(x^{(k)}) - f(x^*) \le \frac{2LR^2}{k+4} \text{ where } R = \max_{f(x) \le f(x^{(0)})} \|x - x^*\|_2.$$

Proof. Let $\epsilon_k = f(x^{(k)}) - f(x^*)$. Lemma 2.2.4 shows that

$$f(x^{(k+1)}) = f(x^{(k)} - \frac{1}{L}\nabla f(x^{(k)})) \le f(x^{(k)}) - \frac{1}{2L}\|\nabla f(x^{(k)})\|_2^2.$$

Subtracting $f(x^*)$ from both sides, we have $\epsilon_{k+1} \le \epsilon_k - \frac{1}{2L}\|\nabla f(x^{(k)})\|_2^2$. Lemma 2.3.1 shows that

$$\epsilon_k \le \|\nabla f(x^{(k)})\|_2 \cdot \|x^{(k)} - x^*\|_2 \le \|\nabla f(x^{(k)})\|_2 \cdot R.$$

Therefore, we have that

$$\epsilon_{k+1} \le \epsilon_k - \frac{1}{2L}\left(\frac{\epsilon_k}{R}\right)^2.$$

Also, we have that

$$\epsilon_0 = f(x^{(0)}) - f^* \le \nabla f(x^*)^\top (x^{(0)} - x^*) + \frac{L}{2}\|x^{(0)} - x^*\|_2^2 \le \frac{LR^2}{2}.$$

Now, we need to solve the recursion. We note that

$$\frac{1}{\epsilon_{k+1}} - \frac{1}{\epsilon_k} = \frac{\epsilon_k - \epsilon_{k+1}}{\epsilon_k \epsilon_{k+1}} \ge \frac{\epsilon_k - \epsilon_{k+1}}{\epsilon_k^2} \ge \frac{1}{2LR^2}.$$

Therefore, after k iterations, we have

$$\frac{1}{\epsilon_k} \ge \frac{1}{\epsilon_0} + \frac{k}{2LR^2} \ge \frac{2}{LR^2} + \frac{k}{2LR^2} = \frac{k+4}{2LR^2}.$$

□

This style of proof is typical in optimization. It shows that when the gradient is large, then we make large progress and when the gradient is small, we are close to optimal.

This proof did not use any property of ℓ_2 or inner product space. Therefore, it works for general norms if the gradient descent step is defined using that norm. For the case of ℓ_2, one can prove that $\|x^{(k)} - x^*\|_2$ is in fact decreasing:

Lemma 2.3.3. *For $h \le \frac{2}{L}$, we have that $\|x^{(k+1)} - x^*\|_2 \le \|x^{(k)} - x^*\|_2$. Therefore, for $h = \frac{1}{L}$, we have*

$$f(x^{(k)}) - f(x^*) \le \frac{2L\|x^{(0)} - x^*\|_2^2}{k+4}.$$

Proof. We compute the distance as follows:

$$\begin{aligned}
\|x^{(k+1)} - x^*\|_2^2 &= \|x^{(k)} - x^* - h\nabla f(x^{(k)})\|_2^2 \\
&= \|x^{(k)} - x^*\|_2^2 - 2h\left\langle \nabla f(x^{(k)}), x^{(k)} - x^* \right\rangle + h^2\|\nabla f(x^{(k)})\|^2 \\
&= \|x^{(k)} - x^*\|_2^2 - 2h\left\langle \nabla f(x^{(k)}) - \nabla f(x^*), x^{(k)} - x^* \right\rangle + h^2\|\nabla f(x^{(k)}) - \nabla f(x^*)\|^2.
\end{aligned}$$

To handle the term $\nabla f(x) - \nabla f(x^*)$, we note that

$$\nabla f(x^{(k)}) - \nabla f(x^*) = H(x^{(k)} - x^*)$$

with $H = \int_0^1 \nabla^2 f(x^* + t(x^{(k)} - x^*))dt$. Since $0 \preceq H \preceq L$ and that $H \succeq \frac{1}{L}H^2$, we have

$$\begin{aligned}
\left\langle \nabla f(x^{(k)}) - \nabla f(x^*), x^{(k)} - x^* \right\rangle &= (x^{(k)} - x^*)^\top H(x^{(k)} - x^*) \\
&\ge \frac{1}{L}(x^{(k)} - x^*)^\top H^2(x^{(k)} - x^*) \\
&= \frac{1}{L}\|\nabla f(x^{(k)}) - \nabla f(x^*)\|^2.
\end{aligned}$$

Hence, we have

$$\begin{aligned}
\|x^{(k+1)} - x^*\|_2^2 &\le \|x^{(k)} - x^*\|_2^2 - (\frac{2h}{L} - h^2)\|\nabla f(x^{(k)}) - \nabla f(x^*)\|^2 \\
&\le \|x^{(k)} - x^*\|_2^2.
\end{aligned}$$

The error estimate follows from $\|x^{(k)} - x^*\|_2^2 \le \|x^{(0)} - x^*\|_2^2$ for all k and the proof in Theorem 2.3.2. □

Rewriting the bound, Theorem 2.3.2 shows it takes $\frac{2L\|x^{(0)} - x^*\|_2^2}{\epsilon}$ iterations. Compare to the bound $\frac{2L}{\epsilon^2}\left(f(x^{(0)}) - f^*\right)$ in Theorem 2.2.3, it seems the new result has a strictly better dependence on ϵ. However, this is not true because one is measuring the error in terms of $\|\nabla f(x)\|_2$ and one is measuring the error in terms of $f(x) - f^*$. For $f(x) = x^2/2$, we have $f(x) - f^* = \|\nabla f(x)\|_2^2$ and hence both have the same dependence on ϵ for this particular function. So, the real benefit of Theorem 2.3.2 is its global convergence.

■ 2.4 Strongly Convex Functions

We note that the convergence rate ϵ^{-1} or ϵ^{-2} is not great if we need to solve the problem up to machine accuracy. Getting to machine accuracy is sometimes important if the optimization problem is used as a subroutine. We note that for the case $f(x) = \frac{1}{2}\|x\|_2^2$, gradient descent with step size $h = 1$ takes exactly 1 step. Therefore, it is natural to ask if one can improve the bound for functions close to quadratics. This motivates the following assumption:

Definition 2.4.1. We call a function $f \in \mathcal{C}^1(\mathbb{R}^n)$ is μ-strongly convex if for any $x, y \in \mathbb{R}^n$

$$f(y) \ge f(x) + \nabla f(x)^\top(y - x) + \frac{\mu}{2}\|y - x\|_2^2.$$

Similar to the convex case (Theorem 1.5.2), we have the following:

Theorem 2.4.2. *Let $f \in \mathcal{C}^2(\mathbb{R}^n)$. For any $\mu \ge 0$, the following are equivalent:*

- $f(\lambda x + (1-\lambda)y) \le \lambda f(x) + (1-\lambda)f(y) - \frac{1}{2}\mu\lambda(1-\lambda)\|x - y\|_2^2$ *for all* $x, y \in \mathbb{R}^n$ *and* $\lambda \in [0, 1]$.
- $f(y) \ge f(x) + \nabla f(x)^\top(y - x) + \frac{\mu}{2}\|y - x\|_2^2$ *for all* $x, y \in \mathbb{R}^n$.

- $\nabla^2 f(x) \succeq \mu$ *for all* $x \in \mathbb{R}^n$

Proof. It follows from applying Theorem 1.5.2 on the function $g(x) = f(x) - \frac{\mu}{2}\|x\|^2$. □

Now, we study gradient descent for μ-strongly convex functions.

Theorem 2.4.3. *Let $f \in \mathcal{C}^2(\mathbb{R}^n)$ be μ-strongly convex with L-Lipschitz gradient and x^* be any minimizer of f. With step size $h = \frac{1}{L}$, the sequence $x^{(k)}$ in* `GradientDescent` *satisfies*

$$f(x^{(k)}) - f(x^*) \leq (1 - \frac{\mu}{L})^k (f(x^{(0)}) - f(x^*)).$$

In a later chapter, we will see that an *accelerated* variant of gradient descent improves this further by replacing the $\frac{\mu}{L}$ term with $\sqrt{\frac{\mu}{L}}$.

Proof. Lemma 2.2.4 shows that

$$f(x^{(k+1)}) - f^* \leq f(x^{(k)}) - f^* - \frac{1}{2L}\|\nabla f(x^{(k)})\|_2^2. \tag{2.4.1}$$

Next, the definition of μ strong convexity implies that

$$f(x^*) \geq f(x^{(k)}) + \nabla f(x^{(k)})^\top (x^* - x^{(k)}) + \frac{\mu}{2}\|x^* - x^{(k)}\|_2^2.$$

Rearranging the term, we have

$$f(x^{(k)}) - f^* \leq \nabla f(x^{(k)})^\top (x^{(k)} - x^*) - \frac{\mu}{2}\|x^{(k)} - x^*\|_2^2 \leq \max_{\Delta} \left(\nabla f(x^{(k)})^\top \Delta - \frac{\mu}{2}\|\Delta\|_2^2 \right) = \frac{1}{2\mu}\|\nabla f(x^{(k)})\|_2^2. \tag{2.4.2}$$

Putting this into the gradient term in (2.4.1) gives

$$f(x^{(k+1)}) - f^* \leq (1 - \frac{\mu}{L})(f(x^{(k)}) - f^*).$$

The conclusion follows. □

It is natural to ask to what extent the assumption of convexity is essential for the bounds we obtained. This is the motivation for the next exercises.

Exercise 2.4.4. Suppose f satisfies $\langle \nabla f(x), x - x^* \rangle \geq \alpha \left(f(x) - f(x^*) \right)$. Derive a bound similar to Theorem 2.3.2 for gradient descent.

Exercise 2.4.5. Suppose f satisfies $\|\nabla f(x)\|_2^2 \geq \mu \left(f(x) - f(x^*) \right)$. Derive a bound similar to Theorem 2.4.3 for gradient descent.

Exercise 2.4.6. Give examples of nonconvex functions satisfying the above conditions. (Note: convex functions satisfy the first with $\alpha = 1$ and μ-strongly convex functions satisfy the second.)

■ 2.5 Line Search

In practice, we often stop the line search early because we can make larger progress via a new direction. One standard stopping condition is called Wolfe conditions.

Definition 2.5.1. A step size h satisfies the Wolfe conditions with respect to direction p if

1. $f(x + hp) \leq f(x) + c_1 h \cdot p^\top \nabla f(x)$,
2. $-p^\top \nabla f(x + hp) \leq -c_2 p^\top \nabla f(x)$.

with $0 < c_1 < c_2 < 1$.

Let the object progress $\phi(h) = f(x) - f(x+hp)$. The first condition requires the algorithm makes sufficient progress ($\phi(h) \geq c_1 h \cdot \phi'(0)$). The second condition requires the slop reduced significantly ($\phi'(h) \leq c_2\phi'(0)$). One can think the first condition gives a upper bound on h while the second condition gives a lower bound on h. In general, step size satisfying the Wolfe conditions will be larger than the step size $h = \frac{1}{L}$. In particular, one can show the following:

Exercise 2.5.2. Suppose f is μ-strongly convex and has L-Lipschitz gradient. If a step size h satisfies the Wolfe conditions with the direction $p = -\nabla f(x)$, then we have

$$\frac{2(1-c_1)}{\mu} \geq h \geq \frac{1-c_2}{L}.$$

As a corollary, we have that the function value progress given by such step is at least $\Omega(\|\nabla f(x)\|^2/L)$. Therefore, this gives the same guarantee Theorem 2.3.2 and Theorem 2.4.3. One common way to via backtracking line search. The algorithm starts with a large step size and decreases it by a constant factor when the Wolfe conditions are violated. For gradient descent, the next step involves exactly computing $\nabla f(x + hp)$ and hence if our line search accepts the step size immediately, the line search almost cost nothing. Therefore, if we maintain a step size throughout the algorithm and decreases it only when it violates the condition, the total cost of the line search will be only an additive logarithmic number of gradient calls throughout the algorithm and it is negligible.

Finally, we note that for problems of the form $\sum_i f_i(a_i^\top x)$, the bottleneck is often in computing Ax. In this case, exact line search is almost free because we can store the vectors Ax and Ah.

2.6 Generalizing Gradient Descent

Now, we study what properties gradient descent are using for the strongly convex case. There are many ways to generalize it. One way is to view gradient descent approximate the function f by splitting it into two terms, one term is the first-order approximation and the second term is just an ℓ_2 norm. More generally, we can split a function into two terms, one term is easy to optimize and the another term we need to approximate with some error. Precisely, we consider the following

Definition 2.6.1. We say $g + h$ is a α-approximation to f at x if

- g is convex, $g(x) = f(x)$,
- $h(x) = 0$ and $h((1-\alpha)x + \lambda\widehat{x}) \leq \lambda^2 h(\widehat{x})$ for all $\widehat{x}$,
- $g(y) + \alpha h(y) \leq f(y) \leq g(y) + h(y)$ for all y.

To understand this assumption, we note that if f is μ-strongly convex with L-Lipschitz gradient, then for any x, we can use $\alpha = \frac{\mu}{L}$ and

$$g(y) = f(x) + \langle \nabla f(x), y - x\rangle,$$
$$h(y) = \frac{L}{2}\|y - x\|^2.$$

The condition requires h converging to 0 quadratically when $y \to x$.

Now, we consider the following algorithm:

Algorithm 3: `GeneralizedGradientDescent`

Input: Initial point $x^{(0)} \in \mathbb{R}^n$, approximation factor $\alpha > 0$.
for $k = 0, 1, \cdots$ **do**
  Find a α-approximation of f at $x^{(k)}$ given by $g^{(k)}(x) + h^{(k)}(x)$.
  $x^{(k+1)} \leftarrow \arg\min_y g^{(k)}(y) + h^{(k)}(y)$.
end

Theorem 2.6.2. *Given a convex function f that we can find a α-approximation at any x. Let x^* be any minimizer of f. Then the sequence $x^{(k)}$ in* `GeneralizedGradientDescent` *satisfies*

$$f(x^{(k)}) - f(x^*) \leq (1-\alpha)^k (f(x^{(0)}) - f(x^*)).$$

Proof. Using that $g^{(k)} + h^{(k)}$ is an upper bound of f, we have that our progress on f is larger than the best possible progress on $g^{(k)} + h^{(k)}$:

$$f(x^{(k+1)}) \leq \min_y g^{(k)}(y) + h^{(k)}(y).$$

To bound the best possible progress, we consider $\widehat{x} = \arg\min_y g^{(k)}(y) + \alpha h^{(k)}(y)$ and $z = (1-\alpha)x^{(k)} + \alpha\widehat{x}$. We have that

$$\begin{aligned}
\min_y g^{(k)}(y) + h^{(k)}(y) &\leq g^{(k)}(z) + h^{(k)}(z) \\
&\leq (1-\alpha) g^{(k)}(x^{(k)}) + \alpha g^{(k)}(\widehat{x}) + \alpha^2 h^{(k)}(\widehat{x}) \\
&\leq (1-\alpha) g^{(k)}(x^{(k)}) + \alpha (g^{(k)}(x^*) + \alpha h^{(k)}(x^*))
\end{aligned}$$

where we used $g^{(k)}$ is convex and the assumption on h in the second inequality, we used $\widehat{x}$ minimizes $g^{(k)} + \alpha h^{(k)}$.

Combining both and using $g^{(k)} + \alpha h^{(k)}$ is a lower bound of f, we have

$$f(x^{(k+1)}) \leq (1-\alpha) f(x^{(k)}) + \alpha f(x^*).$$

This gives the result. □

Although Theorem 2.6.2 looks weird, it captures many theorems. Here we list some of them.

Projected Gradient Descent / Proximal Gradient Descent

Given a convex set K, a μ-strongly convex function f with L-Lipschitz gradient, we consider the problem

$$\min_{x \in K} f(x).$$

To apply Theorem 2.6.2 to $F(x) \stackrel{\text{def}}{=} f(x) + \delta_K(x)$, for any x, we consider the functions

$$\begin{aligned}
g(y) &= f(x) + \langle \nabla f(x), y - x \rangle + \delta_K(y), \\
h(y) &= \frac{L}{2} \|y - x\|_2^2.
\end{aligned}$$

Note that $g + h$ is a $\frac{\mu}{L}$-approximation to F. Theorem 2.6.2 shows the `GeneralizedGradientDescent` converges in $O(\frac{L}{\mu} \log(1/\epsilon))$ steps where each step involves solving the problem

$$\min_{y \in K} f(x) + \langle \nabla f(x), y - x \rangle + \frac{L}{2} \|y - x\|_2^2.$$

More generally, this works for problems of the form

$$\min_x f(x) + \phi(x)$$

for some convex function $\phi(x)$. Theorem 2.6.2 requires us to solve a sub-problem of the form

$$\min_y f(x) + \langle \nabla f(x), y - x \rangle + \frac{L}{2} \|y - x\|_2^2 + \phi(y).$$

ℓ^p Regression

One can apply this framework for optimizing ℓ^p regression $\min_x \|Ax - b\|_p^p$. For example, one can approximate the function $f(x) = x^p$ by $g(y) = x^p + px^{p-1}(y - x)$ and $h(y) = p2^{p-1}(x^{p-2}(y-x)^2 + (y-x)^p)$. Using this, one can show that one can solve the problem

$$\min_x \|Ax - b\|_p^p$$

using the problem

$$\min_y v^\top y + \|DA(y-x)\|_2^2 + \|A(y-x)\|_p^p$$

for some vector v and some diagonal matrix D. One can show that Theorem 2.6.2 only need to solve sub-problem approximately. Therefore, this shows that one can solve ℓ^p regression with $\log(1/\epsilon)$ converging by solving mixed $\ell_2 + \ell_p$ regression approximately.

Recently, [2] and [36] applied these to obtain the fastest algorithms for ℓ^p regression and the ℓ^p flow problem. [32] showed that the ℓ^p flow problem can be used as a subroutine to solve uncapacitied maximum flow problem in $m^{4/3+o(1)}$ time.

Other assumptions

Instead of assuming $h(x)$ converges to 0 quadratically, [49] proved Theorem 2.6.2 assuming h is given by some divergence and showed its applications in D-optimal design.

For some special case, it is possible to analyze this algorithm without convexity. One prominent application is compressive sensing:

$$\min_{\|x\|_0 \le k} \|Ax - b\|_2^2.$$

For matrices A satisfying restricted isometry property, one can apply `GeneralizedGradientDescent` to solve the problem with the splitting $g(x) = 2A^\top(Ax-b) + \delta_{\|x\|_0 \le k}$ and $h(x) = \|x\|^2$. In this case, the algorithm is called iterative hard-thresholding [9] and the sub-problem has a closed form expression.

Exercise 2.6.3. Give the close form solution for the sub-problem given by the splitting above.

2.7 Gradient Flow

In continuous time, gradient descent follows the ODE

$$\frac{dx_t}{dt} = -\nabla f(x_t).$$

This can be viewed as the canonical continuous algorithm. Finding the right discretization has lead to many fruitful research directions. One benefit of the continuous view is to simplify some calculations. For example, the Theorem 2.4.3 now becomes

$$\frac{d}{dt}(f(x_t) - f(x^*)) = \nabla f(x_t)^\top \frac{dx_t}{dt} = -\|\nabla f(x_t)\|_2^2 \le -2\mu(f(x_t) - f(x^*))$$

where we used (2.4.2) at the end. Solving this differential inequality, we have

$$f(x_t) - f(x^*) \le e^{-2\mu t}(f(x_0) - f(x^*)).$$

Without the strongly convexity assumption, the gradient flow can behave wildly. For example, the length of the gradient flow can be exponential in d on an unit ball [50]. Finally, we emphasize that this continuous view is mainly useful for understanding, indicative of but not necessarily an algorithmic result.

Chapter 3

Elimination

3.1 Cutting Plane Methods

The goal of this chapter is to present some polynomial-time algorithms for convex optimization. These algorithms use the knowledge of the function in a minimal way, essentially by querying the function value and knowing some (weak) bounds on its support/value.

Given a continuously differentiable convex function f, Theorem 1.2.6 shows that

$$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle \text{ for all } y. \tag{3.1.1}$$

Let x^* be any minimizer of f. Replacing y with x^*, we have that

$$f(x) \geq f(x^*) \geq f(x) + \langle \nabla f(x), x^* - x \rangle .$$

Therefore, we know that $\langle \nabla f(x), x^* - x \rangle \leq 0$. Namely, x^* lies in a halfspace H with normal vector $-\nabla f(x)$. Roughly speaking, this shows that each gradient computation cuts the set of possible solutions in half. In one dimension, this allows us to do a binary search to minimize convex functions.

It turns out that in $\mathbb{R}^n$, binary search still works. In this chapter, we will cover several ways to do this binary search. All of them follow the same framework, called the *cutting plane method*.

More generally, cutting plane methods solves the following class of problems:

Open Problem 3.1.1 (Finding a point in a convex set)**.** Given a convex set $K \subset \mathbb{R}^n$. Suppose for any $x \notin K$, we can find a vector $g(x)$ such that

$$g(x)^\top (y - x) \leq 0 \text{ for all } y \in K.$$

Our goal is to find some $y \in K$ or prove that $\mathrm{vol} K \leq \epsilon^n$ using as few calls to g as possible.

Remark. When apply to convex functions, we have $g(x) = \nabla f(x)$ and K be the set of minimizer of f. In chapter 3.3, we relate the problem of proving $\mathrm{vol} K$ is small and the problem of finding an approximate minimizer of f.

In this framework, we maintain a convex set $E^{(k)}$ that contains the set K. Each iteration, we compute $g(x^{(k)})$ with $x^{(k)}$ chosen based on $E^{(k)}$. The guarantee of g shows that K lies in the halfspace $H^{(k)} = \{y : g(x^{(k)})^\top (y - x^{(k)}) \leq 0\}$ and hence $K \subset H^{(k)} \cap E^{(k)}$. The algorithm continues by choosing $E^{(k+1)}$ which contains $H^{(k)} \cap E^{(k)}$.

Algorithm 4: `CuttingPlaneFramework`

Input: Initial set $E^{(0)} \subseteq \mathbb{R}^n$ containing K.
for $k = 0, \cdots$ **do**
  Find a point $x^{(k)} \in E^{(k)}$.
  if $E^{(k)}$ *is "small" enough* **then return** $x^{(k)}$.
  Find $E^{(k+1)} \supset E^{(k)} \cap H^{(k)}$ where

$$H^{(k)} \stackrel{\text{def}}{=} \{x \in \mathbb{R}^n \text{ such that } g(x^{(k)})^\top (x - x^{(k)}) \leq 0\}. \tag{3.1.2}$$

end

The main questions are

1. How do we choose $x^{(k)}$ and $E^{(k+1)}$?
2. How do we measure progress?
3. How quickly does the method converge?
4. How expensive is each step?

Progress on the cutting plane method is shown in the next table.

Year	$E^{(k)}$ and $x^{(k)}$	Iter	Cost/Iter
1965 [41, 52]	Center of gravity	n	n^n
1979 [65, 56, 34]	Center of ellipsoid	n^2	n^2
1988 [33]	Center of John ellipsoid	n	$n^{2.878}$
1989 [60]	Volumetric center	n	$n^{2.378}$
1995 [6]	Analytic center	n	$n^{2.378}$
2004 [8]	Center of gravity	n	n^6
2015 [39]	Hybrid center	n	n^2 (amortized)
2020 [25]	Volumetric center	n	n^2 (amortized)

Table 3.1: Different Cutting Plane Methods. Omitted polylogarithmic terms. The number of iterations follows from the rate.

3.2 Ellipsoid Method

We first start by explaining the simplest algorithm, the Ellipsoid method. We maintain an ellipsoid

$$E^{(k)} \stackrel{\text{def}}{=} \{x \in \mathbb{R}^n : \ (x - x^{(k)})^\top \left(A^{(k)}\right)^{-1} (x - x^{(k)}) \leq 1\}$$

that contains K. After we compute $g(x^{(k)})$ and $H^{(k)}$ via (3.1.2), we define $E^{(k+1)}$ to be the smallest volume ellipsoid containing $E^{(k)} \cap H^{(k)}$. The key observation is that the volume of the ellipsoid $E^{(k)}$ decreases by $1 - \Theta(\frac{1}{n})$ every iteration. This volume property holds for any halfspace through the center of the current ellipsoid (not only for the one whose normal is the gradient), a property we will exploit in the next chapter.

Algorithm 5: `Ellipsoid`

Input: Initial ellipsoid $E^{(0)} = \{x \in \mathbb{R}^n : \ (x - x^{(0)})^\top \left(A^{(0)}\right)^{-1} (x - x^{(0)}) \leq 1\}$.
for $k = 0, \cdots$ **do**
  if $E^{(k)}$ *is "small" enough* **then return** $x^{(k)}$.
  $x^{(k+1)} \leftarrow x^{(k)} - \frac{1}{n+1} \frac{A^{(k)} g(x^{(k)})}{\sqrt{g(x^{(k)})^\top A^{(k)} g(x^{(k)})}}$.
  $A^{(k+1)} \leftarrow \frac{n^2}{n^2-1} \left(A^{(k)} - \frac{2}{n+1} \frac{A^{(k)} g(x^{(k)}) g(x^{(k)})^\top A^{(k)}}{g(x^{(k)})^\top A^{(k)} (x^{(k)})}\right)$.
end

Lemma 3.2.1. *For the Ellipsoid method (Algorithm 5), we have* $\mathrm{vol} E^{(k+1)} < e^{-\frac{1}{2n+2}} \mathrm{vol} E^{(k)}$.

Remark 3.2.2. Note that the proof below also shows that $\mathrm{vol} E^{(k+1)} = e^{-\Theta(\frac{1}{n})} \mathrm{vol} E^{(k)}$. Therefore, the ellipsoid method does not run faster for nice functions, making it provably slow in practice.

Proof. Note that the ratio of $\mathrm{vol} E^{(k+1)} / \mathrm{vol} E^{(k)}$ does not change under any affine transformation. Therefore, we can do a transformation so that $A^{(k)} = I$, $x^{(k)} = 0$ and $g(x^{(k)}) = e_1$. This simplifies our calculation. We need to prove two statements: $\mathrm{vol} E^{(k+1)} < e^{-\frac{1}{2n+2}} \mathrm{vol} E^{(k)}$ and that $E^{(k)} \cap H^{(k)} \subset E^{(k+1)}$.

Claim 1: $\mathrm{vol} E^{(k+1)} < e^{-\frac{1}{2(n+1)}} \mathrm{vol} E^{(k)}$.

Note that since $g(x^{(k)}) = e_1$,we have $A^{(k+1)} = \frac{n^2}{n^2-1}\left(I - \frac{2}{n+1}e_1e_1^\top\right)$. Therefore, we have that

$$\begin{aligned}\left(\frac{\mathrm{vol}E^{(k+1)}}{\mathrm{vol}E^{(k)}}\right)^2 = \left|\frac{\det A^{(k+1)}}{\det A^{(k)}}\right| &= \left(\frac{n^2}{n^2-1}\right)^n \det\left(I - \frac{2}{n+1}e_1e_1^\top\right)\\ &= \left(\frac{n^2}{n^2-1}\right)^{n-1}\frac{n^2}{n^2-1}\left(\frac{n-1}{n+1}\right)\\ &= \left(1+\frac{1}{n^2-1}\right)^{n-1}\left(1-\frac{1}{n+1}\right)^2\\ &< \exp(\frac{n-1}{n^2-1} - \frac{2}{n+1}) = \exp(-\frac{1}{n+1})\end{aligned}$$

where we used $1 + x \le e^x$ for all x.

Claim 2. $E^{(k)} \cap H^{(k)} \subset E^{(k+1)}$.

By the definition of $E^{(k)}$and the assumption $A^{(k)} = I$, we have

$$x^{(k+1)} = -\frac{e_1}{n+1}$$

and, for any $x \in E^{(k)} \cap H^{(k)}$, we have that $\|x\|_2 \le 1$ and $x_1 \le 0$. By direct computation, we have

$$\begin{aligned}&(x + \frac{e_1}{n+1})^\top\left(1-\frac{1}{n^2}\right)\left(I + \frac{2}{n-1}e_1e_1^\top\right)(x+\frac{e_1}{n+1})\\ =&\left(1-\frac{1}{n^2}\right)\left(\|x\|^2 + \frac{2x_1(1+x_1)}{n-1} + \frac{1}{n^2-1}\right)\\ \le&\left(1-\frac{1}{n^2}\right)\left(1+0+\frac{1}{n^2-1}\right) = 1\end{aligned}$$

where we used that $\|x\|_2 \le 1$ and $x_1(1+x_1) \le 0$ (because $-1 \le x_1 \le 0$) at the end. This shows that $x \in E^{(k+1)}$. □

Exercise 3.2.3. Show that the ellipsoid $E^{(k+1)}$ computed above is the minimum volume ellipsoid containing $E^{(k)} \cap H^{(k)}$.

■ 3.3 From Volume to Function Value

Lemma 3.2.1 shows that the volume of a set containing all minimizers decreases by a constant factor every n steps. In general, knowing that the optimal x^* lies in a small volume set does not provide enough information to find a point with small function value. For example, if we only knew that x^* lies in the plane $\{x : x_1 = 0\}$, we would still need to search for x^* over an $n-1$ dimensional space. However, if the set is constructed by the cutting plane framework (Algorithm 4), then we can guarantee that small volume implies that any point in the set has close to optimal function value.

This in turns implies that we can minimize any convex function with ε additive error in $O(n^2\log(1/\varepsilon))$ iterations. To make the statement more general, we note that ellipsoid method can be used for non-differentiable functions.

Theorem 3.3.1. *Let $x^{(k)}$ be the sequence of points produced by the cutting plane framework (Algorithm 4) for a convex function f. Given a function $\mathcal{V}$ mapping subsets of $\mathbb{R}^n$ to non-negative numbers such that*

1. *(Linearity) For any set $E \subseteq \mathbb{R}^n$, any vector v and any scalar $\alpha \ge 0$, we have $\mathcal{V}(\alpha E + v) = \alpha\mathcal{V}(E)$ where $\alpha E + v = \{\alpha x + v : x \in E\}$.*
2. *(Monotonic) For any set $F \subset E$, we have that $\mathcal{V}(F) \le \mathcal{V}(E)$.*

Then, for any set $\Omega \subset E^{(0)}$, we have that

$$\min_{i=1,2,\cdots k} f(x^{(i)}) - \min_{y\in\Omega} f(y) \le \frac{\mathcal{V}(E^{(k)})}{\mathcal{V}(\Omega)} \cdot \left(\max_{z\in\Omega} f(z) - \min_{x\in\Omega} f(x)\right) .$$

Remark 3.3.2. We can think $\mathcal{V}(E)$ as some way to measure the size of E. It can be radius, mean-width or any other way to measure "size". For the ellipsoid method, we use $\mathcal{V}(E) = \mathrm{vol}(E)^{\frac{1}{n}}$ because we have proved the volume decreases in Lemma 3.2.1. We raise the volume to $\frac{1}{n}$-th power to satisfy linearity.

Proof. Let x^* be any minimizer of f over Ω. For any $\alpha > \frac{\mathcal{V}(E^{(k)})}{\mathcal{V}(\Omega)}$ and $S = (1-\alpha)x^* + \alpha\Omega$. By the linearity of $\mathcal{V}$, we have that

$$\mathcal{V}(S) = \alpha\mathcal{V}(\Omega) > \mathcal{V}(E^{(k)})$$

Therefore, S is not a subset of $E^{(k)}$ and hence there is a point $y \in S \backslash E^{(k)}$. y is not in $E^{(k)}$. This means it is separated by the gradient at some step $i \leq k$, namely

$$\nabla f(x^{(i)})^\top (y - x^{(i)}) > 0.$$

By the convexity of f, we have $f(x^{(i)}) \leq f(y)$. Since $y \in S$, we have $y = (1-\alpha)x^* + \alpha z$ for some $z \in \Omega$. Thus, the convexity of f implies that

$$f(x^{(i)}) \leq f(y) \leq (1-\alpha)f(x^*) + \alpha f(z).$$

Therefore, we have

$$\min_{i=1,2,\cdots k} f(x^{(i)}) - \min_{x\in\Omega} f(x) \leq \alpha \left(\max_{z\in\Omega} f(z) - \min_{x\in\Omega} f(x) \right).$$

Since this holds for any $\alpha > \frac{\mathcal{V}(E^{(k)})}{\mathcal{V}(\Omega)}$, we have the result. □

Combining Lemma 3.2.1 and Theorem 3.3.1, we have the following rate of convergence.

Theorem 3.3.3. *Let f be a convex function on $\mathbb{R}^n$, $E^{(0)}$ be any initial ellipsoid and $\Omega \subset E^{(0)}$ be any convex set. Suppose that for any $x \in E^{(0)}$, we can find, in time $\mathcal{T}$, a nonzero vector $g(x)$ such that*

$$f(y) \geq f(x) \text{ for any } y \text{ such that } g(x)^\top (y - x) \geq 0.$$

Then, we have

$$\min_{i=1,2,\cdots k} f(x^{(i)}) - \min_{y\in\Omega} f(y) \leq \left(\frac{\mathrm{vol}(E^{(0)})}{\mathrm{vol}(\Omega)} \right)^{\frac{1}{n}} \exp\left(-\frac{k}{2n(n+1)} \right) \left(\max_{z\in\Omega} f(z) - \min_{x\in\Omega} f(x) \right).$$

Furthermore, each iteration takes $O(n^2 + \mathcal{T})$ time.

Remark 3.3.4. We usually obtain $g(x)$ from a *separation oracle*.

Proof. Lemma 3.2.1 shows that the volume of the ellipsoid maintained decreases by $\exp(-\frac{1}{2n+2})$ every iteration. Hence, $\mathrm{vol}^{\frac{1}{n}}$ decreases by $\exp(-\frac{1}{2n(n+1)})$ every iteration. The bound follows by applying Theorem 3.3.1 with $\mathcal{V}(E) \stackrel{\text{def}}{=} \mathrm{vol}(E)^{\frac{1}{n}}$. Next, we note the proof of Theorem 3.3.1 only used the fact that one side of halfspace defined by the gradient has higher value. Therefore, we can replace the gradient with the vector $g(x)$. □

This theorem can be used to solve many problems in polynomial time. As an illustration, we show how to solve linear programs in polynomial time here.

Theorem 3.3.5. *Given a linear program $\min_{x\in\Omega} c^\top x$ where $\Omega = \{x \text{ such that } Ax \geq b\}$. Let the diameter $R \stackrel{\text{def}}{=} \max_{x\in\Omega} \|x\|_2$ and the volume radius $r = \mathrm{vol}(\Omega)^{\frac{1}{n}}$. Then, we can find x such that*

$$f(x) - \min_{x\in\Omega} f(x) \leq \varepsilon \cdot (\max_{x\in\Omega} c^\top x - \min_{x\in\Omega} c^\top x)$$

in $O(n^2(n^2 + \mathrm{nnz}(A)) \log(\frac{R}{r\varepsilon}))$ time where $\mathrm{nnz}(A)$ is the number of non-zero elements in A.

Remark 3.3.6. If the dimension n is constant, this algorithm is nearly linear time (linear to the number of constraints)!

Proof. For the linear program $\min_{Ax \geq b} c^\top x$, the function we want to minimize is

$$L(x) = c^\top x + \delta_{Ax \geq b} \quad \text{where} \quad \delta_{Ax \geq b}(x) = \begin{cases} 0 & \text{if } a_i^\top x \geq b_i \text{ for all } i \\ +\infty & \text{otherwise.} \end{cases}. \tag{3.3.1}$$

For this function L, we can use the separation oracle $g(x) = c$ if $Ax \geq b$ and $g(x) = -a_i$ if $a_i^\top x < b$. If there are many violated constraints, any one of the corresponding a_i's will do.

We can simply pick $E^{(0)}$ be a unit ball centered at 0 with radius R and apply Theorem 3.3.3 to find x such that

$$f(x) - \min_{x \in \Omega} f(x) \leq \varepsilon(\max_{x \in \Omega} c^\top x - \min_{x \in \Omega} c^\top x)$$

in time $O(n^2(n^2 + \text{nnz}(A)) \log(\frac{R}{r\varepsilon}))$. □

To get the solution exactly, i.e., $\varepsilon = 0$, we need to assume the linear program has integral (or rational) coefficients and the running time depends on the bound of the numbers in the matrix A and the vectors b and c. It is still open how to solve linear programs in time bounded by a polynomial in only the number of variables and constraints (and not the bit sizes of the coefficients). Such a running time is called *strongly polynomial*.

Open Problem. Can we solve linear programs in strongly polynomial time?

■ 3.4 Center of Gravity Method

In the Ellipsoid method, we used an ellipsoid as the current set, and its center as the next query point.

In the center of gravity method, we start with any bounded convex set containing a minimizer, e.g., a large enough cube, and the set maintained is simply the intersection of all halfspaces used so far. The query point will be the center of gravity (or barycenter) of the current set. The measure of progress will once again be the volume (or more precisely, volume radius) of the current set.

It is clear that the volume can only decrease in each iteration. But at what rate? The following classical theorem shows that the volume of the convex body decreases by at least a $1 - \frac{1}{e}$ factor when using the exact center of gravity.

Theorem 3.4.1. *Let K be a convex body in $\mathbb{R}^n$ with center of gravity z. Let H be any halfspace containing z. Then,*

$$\text{vol}(K \cap H) \geq \left(\frac{n}{n+1}\right)^n \text{vol}(K).$$

Note that the RHS is at least $1/e$. We prove the theorem later in this chapter. Unfortunately, computing the center of gravity even of a polytope is #P-hard [54]. For the purpose of efficient approximations, it is important to establish a stable version of the theorem that does not require an exact center of gravity.

Recall that a nonnegative function is logconcave if its logarithm is concave, i.e., for any $x, y \in \mathbb{R}^n$ and any $\lambda \in [0, 1]$,

$$f(\lambda x + (1 - \lambda)y) \geq f(x)^\lambda f(y)^{1-\lambda}.$$

We refer to Sections 1.6 for some background on logconcave functions. A distribution p is *isotropic* if the mean of a random variable drawn from the distribution is zero and the covariance is the identity matrix.

The randomized center of gravity method defined as follows: Let $y^1, \ldots, y^N$ be uniform random points from $E^{(k)}$ and define

$$x^{(k)} = \frac{1}{N} \sum_{i=1}^{N} y^i,$$
$$E^{(k+1)} = E^{(k)} \cap H^{(k)}$$

To prove its convergence, we will use a robust version of the following classical theorem of Grunbaum.

Theorem 3.4.2 (Robust Grunbaum)**.** *Let p be an isotropic logconcave distribution, namely $\mathbf{E}_{x\sim p}x = 0$ and $\mathbf{E}_{x\sim p}x^2 = 1$. For any $\theta \in \mathbb{R}^n$, $t \in \mathbb{R}$ we have*

$$\mathbf{P}_{x\sim p}(x^\top \theta \geq t) \geq \frac{1}{e} - t.$$

Proof. By taking the marginal with respect to the direction θ, we can assume the distribution is one-dimensional. Let $P(t) = \mathbf{P}_{x\sim p}(x^\top\theta \leq t)$. Note that $P(t)$ is the convolution of p and $1_{(-\infty,0]}$. Hence, it is logconcave (Lemma 1.6.3). By some limit arguments, we can assume $P(-M) = 0$ and $P(M) = 1$ for some large enough M (to be rigorous, we do the proof below for finite M and a RHS $\epsilon(M)$ instead of zero, then take the limit $M \to \infty$). Since $\mathbf{E}_{x\sim p}x = 0$, we have that

$$\int_{-M}^{M} tP'(t)\,dt = 0$$

Integration by parts gives that $\int_{-M}^{M} P(t)dt = M$. Note that $P(t)$ is increasing logconcave, if $P(0)$ is too small, it would make $\int_{-M}^{M} P(t)dt$ too small. To be precise, since P is logconcave, we have that

$$-\log P(t) \geq -\log P(0) - \frac{P'(0)}{P(0)}t.$$

Or we simply write $P(t) \leq P(0)e^{\alpha t}$ for some α. Hence,

$$M = \int_{-M}^{M} P(t)dt \leq \int_{-\infty}^{\frac{1}{\alpha}} P(0)e^{\alpha t}dt + \int_{1/\alpha}^{M} 1dt = \frac{eP(0)}{\alpha} + M - \frac{1}{\alpha}.$$

This shows that $P(0) \geq \frac{1}{e}$.

Next, Lemma 3.4.3 shows that $\max_x p(x) \leq 1$. Therefore, the cumulative distribution P is 1-Lipschitz and we have

$$\mathbf{P}_{x\sim p}(x^\top\theta \geq t) \geq \mathbf{P}_{x\sim p}(x^\top\theta \geq 0) - t \geq \frac{1}{e} - t.$$

□

Lemma 3.4.3. *Let p be a one-dimensional isotropic logconcave density. Then* $\max p(x) \leq 1$.

For a proof of this (and for other properties of logconcave functions), we refer the reader to [48].

Exercise 3.4.4. Give a short proof that $\max_x p(x) = O(1)$ for any one-dimensional isotropic logconcave density.

Using the robust Grunbaum theorem 3.4.2, we get the following algorithm, which uses uniform random points from the current set. Obtaining such a random sample algorithmically is an interesting problem that we will study in the second part of this book.

Lemma 3.4.5. *Suppose $y^1, \ldots, y^N$ are i.i.d. uniform random points from a convex body K and $y = \frac{1}{N}\sum_{i=1}^{N} y^i$. Then for any halfspace H not containing y,*

$$\mathbf{E}(\mathrm{vol}(K \cap H)) \leq \left(1 - \frac{1}{e} + \sqrt{\frac{n}{N}}\right)\mathrm{vol}(K).$$

Proof. Without loss of generality, we assume that K is in isotropic position, i.e., $\mathbf{E}_K(y^i) = 0$ and $\mathbf{E}_K(y^i(y^i)^\top) = I$. Then we have $\mathbf{E}(y) = 0$ and

$$\mathbf{E}\,\|y\|^2 = \frac{1}{N}\mathbf{E}\,\|y^i\|^2 = \frac{n}{N}.$$

Therefore,

$$\mathbf{E}\,\|y\| \leq \sqrt{\mathbf{E}\left(\|y\|^2\right)} = \sqrt{\frac{n}{N}}.$$

Thus, we can apply Theorem 3.4.2 with $t = \sqrt{\frac{n}{N}}$. □

Theorem 3.3.1 readily gives the following guarantee for convex optimization, again using volume radius as the measure of progress.

Theorem 3.4.6. *Let f be a convex function on $\mathbb{R}^n$, $E^{(0)}$ be any initial set and $\Omega \subset E^{(0)}$ be any convex set. Suppose that for any $x \in E^{(0)}$, we can find a nonzero vector $g(x)$ such that*

$$f(y) \geq f(x) \text{ for any } y \text{ such that } g(x)^\top (y - x) \geq 0.$$

Then, for the center of gravity method with $N = 10n$, we have

$$\mathbf{E} \min_{i=1,2,\cdots k} f(x^{(i)}) - \min_{y \in \Omega} f(y) \leq \left(\frac{\text{vol}(E^{(0)})}{\text{vol}(\Omega)} \right)^{\frac{1}{n}} (0.95)^{\frac{k}{n}} \left(\max_{z \in \Omega} f(z) - \min_{x \in \Omega} f(x) \right) .$$

Now, we give a geometric proof Theorem 3.4.1. Note that one can modify the proof of Theorem 3.4.2 to get another proof.

Proof. Without loss of generality, assume that the center of gravity of K is the origin and H is the halfspace $\{x : x_1 \leq 0\}$. For each point $t \in \mathbb{R}$, let $A(t) = K \cap \{x : x_1 = t\}$ be the $(n-1)$-dimensional slice of K with $x_1 = t$. Define $r(t)$ as the radius of the $(n-1)$-dimensional ball with the same $(n-1)$-dimensional volume as $A(t)$.

The goal of the proof is to show that the smallest possible halfspace volume is achieved for a cone by a cut perpendicular to its axis. In the first step, we will symmetrize K as follows: replace each cross-section $A(t)$ by a ball of the same volume, centered at $(t, 0, \ldots, 0)^T$. We claim that the resulting rotationally symmetric body is convex. To see this, note that all we have to show is that the radius function $r(t)$ is concave. For any $s, t \in \mathbb{R}$, and any $\lambda \in [0, 1]$, we have by convexity of K that

$$\lambda A(s) + (1 - \lambda) A(t) \subseteq A(\lambda s + (1 - \lambda) t)$$

and by the Brunn-Minkowski theorem (Theorem 1.6.5) applied to $A(s), A(t)$, we have

$$\text{vol}_{n-1}(A(\lambda s + (1 - \lambda) t))^{\frac{1}{n-1}} \geq \lambda \text{vol}_{n-1}(A(s))^{\frac{1}{n-1}} + (1 - \lambda) \text{vol}_{n-1}(A(t))^{\frac{1}{n-1}}.$$

From this and the definition of $r(t)$, it follows that

$$r(\lambda s + (1 - \lambda) t) \geq \lambda r(s) + (1 - \lambda) r(t)$$

as desired.

Next consider the subset $K_1 = K \cap \{x : x_1 \leq 0\}$. We replace this subset with a cone C having the same base $A(0)$ and apex at some point along the e_1 axis so that the volume of the cone is the same as $\text{vol}(K_1)$. Using the concavity of the radial function, this transformation can only decrease the center of gravity along e_1. Therefore, proving a lower bound on the transformed body K_1 will give a lower bound for K. So assume we do this and the center of gravity is the origin. Next, extend the cone to the right, so that it remains a rotational cone, and the volume in the positive halfspace along e_1 is the same as $\text{vol}(K \setminus K_1)$. Once again, the center of gravity can only move to the left, and so the volume of K_1 can only decrease by this transformation. At the end we have shown that the lower bound for any convex body follows by proving for a rotational cone with axis along the normal to the halfspace. Now we compute the volume ratio:

$$\frac{\text{vol}(K_1)}{\text{vol}(K)} = \left(\frac{n}{n+1} \right)^n .$$

□

Discussion

In later chapters we will see how to implement each iteration of the center of gravity method in polynomial time. Computing the exact center of gravity is #P-hard even for a polytope [54], but we can find an arbitrarily close approximation in randomized polynomial time via sampling. The method generalizes to certain noisy computation models, e.g., when the oracle reports a function value that is within a bounded additive error, i.e., a noisy function oracle.

We will also see that the iteration complexity of the center of gravity method, $O(n)$ iterations, is asymptotically optimal.

3.5 Sphere and Parabola Methods

In Section 2.2, we proved that gradient descent converges at the rate $(1-\frac{\mu}{L})^k$ in function value assuming the function satisfies $\mu \cdot I \preceq \nabla^2 f(x) \preceq L \cdot I$ for all x. Hence, if $\frac{L}{\mu}$ is not too large, this rate of decrease of the function value can be much better than the $\left(1-\frac{1}{n^2}\right)^k$ rate of the ellipsoid method or the $\left(1-\frac{1}{n}\right)^k$ rate of the center of gravity method (recall that the convergence rate in function value is n times slower than the convergence rate in volume). In this section, we show how to modify the ellipsoid method to get a faster convergence rate when $\frac{L}{\mu}$ is small. One can view this whole section as just an interpretation of accelerated gradient descent (which we haven't seen yet) in the cutting plane framework. In a later section, we will give another interpretation.

Sphere Method

We have been using convexity to find a halfspace containing the optimum with the current point on its boundary. For a strongly convex function, any optimal x^* is contained in a strictly smaller region than a halfspace. In particular, we have

$$f(x^*) \geq f(x) + \nabla f(x)^\top (x^* - x) + \frac{\mu}{2}\|x^* - x\|_2^2.$$

Completing the square, we have that

$$\frac{\mu}{2}\left\|(x^* - x) + \frac{1}{\mu}\nabla f(x)\right\|_2^2 - \frac{\|\nabla f(x)\|_2^2}{2\mu} \leq -\left(f(x) - f(x^*)\right)$$

Using $x^+ \stackrel{\text{def}}{=} x - \frac{1}{L}\nabla f(x)$ and $x^{++} \stackrel{\text{def}}{=} x - \frac{1}{\mu}\nabla f(x)$, we can write

$$\left\|x^* - x^{++}\right\|_2^2 \leq \frac{\|\nabla f(x)\|_2^2}{\mu^2} - \frac{2}{\mu}(f(x) - f(x^*)). \tag{3.5.1}$$

To use this formula in the cutting plane framework, we need a crude upper bound on $f(x^*)$. One can simply use $f(x^*) \leq f(x)$. Or, we can use Lemma 2.2.4 and get

$$f(x^*) \leq f(x^+) \leq f(x - \frac{1}{L}\nabla f(x)) \leq f(x) - \frac{1}{2L}\|\nabla f(x)\|^2.$$

Putting it into (3.5.1), we see that

$$\begin{aligned}
\left\|x^* - x^{++}\right\|_2^2 &\leq \frac{1}{\mu^2}\left(\|\nabla f(x)\|_2^2 - 2\mu \cdot (f(x) - f(x^*))\right) \\
&\leq \frac{1}{\mu^2}\left(\|\nabla f(x)\|_2^2 - 2\mu \cdot \left(f(x) - f(x^+)\right)\right) - \frac{2}{\mu}\left(f(x^+) - f(x^*)\right) \\
&\leq \frac{1}{\mu^2}\left(\|\nabla f(x)\|_2^2 - 2\mu \cdot \frac{1}{2L}\|\nabla f(x)\|_2^2\right) - \frac{2}{\mu}\left(f(x^+) - f(x^*)\right) \\
&\leq \frac{\left(1-\frac{\mu}{L}\right)}{\mu^2}\|\nabla f(x)\|_2^2 - \frac{2}{\mu}\left(f(x^+) - f(x^*)\right).
\end{aligned} \tag{3.5.2}$$

Therefore, using the trivial bound of zero for the second term on the RHS, x^* lies in a ball centered at x^{++} with radius at most

$$\sqrt{1-\frac{\mu}{L}} \cdot \frac{\|\nabla f(x)\|_2}{\mu}. \tag{3.5.3}$$

This suggests using balls instead of ellipsoids in a cutting plane algorithm; it would certainly be more efficient to maintain!

We arrive at the following algorithm.

Algorithm 6: `SphereMethod`

Input: Initial point $x^{(0)} \in \mathbb{R}^n$, strong convexity parameter μ, Lipschitz gradient parameter L.
$Q^{(0)} \leftarrow \mathbb{R}^n$.
for $k = 0, \cdots$ **do**
  Set $Q = \left\{ x \in \mathbb{R}^n : \left\| x - (x^{(k)} - \frac{1}{\mu}\nabla f(x^{(k)})) \right\|_2^2 \le \frac{1-\frac{\mu}{L}}{\mu^2} \cdot \left\| \nabla f(x^{(k)}) \right\|_2^2 \right\}$.
  $Q^{(k+1)} \leftarrow$ `minSphere`$(Q \cap Q^{(k)})$ where `minSphere`(K) is the smallest sphere covering K.
  $x^{(k+1)} \leftarrow$ center of $Q^{(k+1)}$.
end

To analyze `SphereMethod`, we need the following lemma, which is illustrated in Figure 3.5.1.

Lemma 3.5.1. *For any $g \in \mathbb{R}^n$ and $\epsilon \in (0,1)$, we have $B(0,1) \cap B(g, \|g\|_2\sqrt{1-\epsilon}) \subset B(x, \sqrt{1-\epsilon})$ for some x.*

Proof. It suffices to consider the two-dimensional case by symmetry, and to assume that $g = ae_1$. If $a \le 1$, we can simply pick $x = g$. Otherwise, let $(x, 0)$ be the center of the smallest ball containing the required intersection, and y be its radius. (See Figure 3.5.1). We have $x^2 + y^2 = 1$ and $(x-a)^2 + y^2 = (1-\epsilon)a^2$. This implies that

$$x = \frac{1 + \epsilon a^2}{2a}$$

and so

$$y^2 = 1 - \frac{\epsilon}{2} - \frac{1}{4a^2} - \frac{\epsilon^2 a^2}{4} \le 1 - \epsilon$$

as claimed. □

Lemma 3.5.2. *Let the measure of progress $\mathcal{V}(Q) = \text{radius}(Q)$. Then, we have that $x^* \in Q^{(k)}$ and $\mathcal{V}(Q^{(k+1)}) \le \sqrt{1-\frac{\mu}{L}} \cdot \mathcal{V}(Q^{(k)})$ for all k.*

Remark 3.5.3. The function value decrease follows from Theorem 3.3.1.

Proof Sketch. The fact $x^* \in Q^{(k)}$ follows directly from the definition of Q. For the decrease of radius, suppose that

$$Q^{(k)} = \left\{ x \in \mathbb{R}^n : \|x - x^{(k)}\| \le R^{(k)} \right\}.$$

Then, the new ball is given by

$$Q^{(k+1)} = \texttt{minSphere}\left(\left\{ \left\| x - (x^{(k)} - \frac{1}{\mu}\nabla f(x^{(k)})) \right\|_2^2 \le \frac{1-\frac{\mu}{L}}{\mu^2} \cdot \left\| \nabla f(x^{(k)}) \right\|_2^2 \right\} \cap \left\{ \|x - x^{(k)}\| \le R^{(k)} \right\} \right).$$

To compute $\frac{\text{radius}(Q^{(k+1)})^2}{\text{radius}(Q^{(k)})^2}$, we can assume $x^{(k)} = 0$ and $R^{(k)} = 1$ and let $g = \frac{\nabla f(0)}{\mu}$. Hence, we have

$$Q^{(k+1)} = \texttt{minSphere}\left(\left\{ \|x - g\|_2^2 \le (1 - \frac{\mu}{L}) \cdot \|g\|_2^2 \right\} \cap \{\|x\| \le 1\} \right).$$

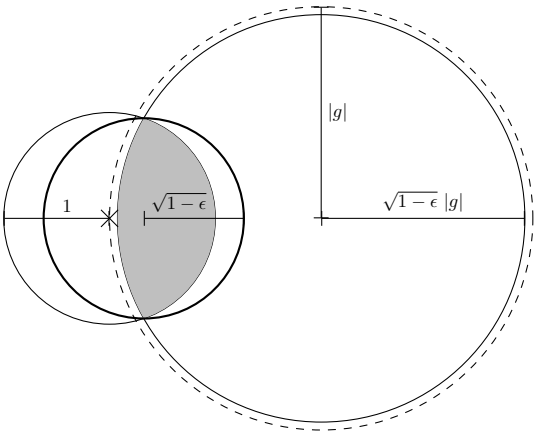
Now Lemma 3.5.1 (with $\epsilon = \frac{\mu}{L}$) shows that the $\text{radius}(Q^{(k+1)})^2 \le 1 - \frac{\mu}{L}$. □

Note that this gives the same convergence rate as gradient descent for strongly convex functions. Each iteration is much faster than the $O(n^2)$ time of the Ellipsoid method.

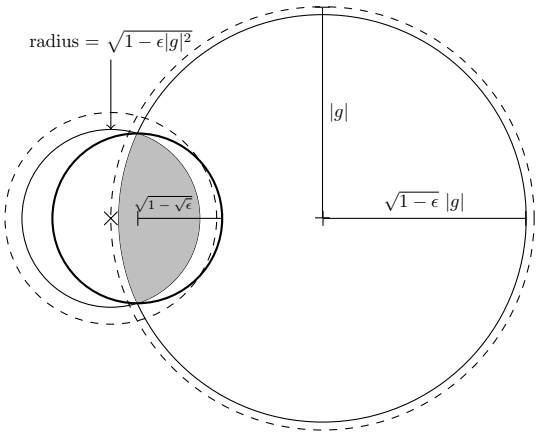
Parabola Method Via Epigraph

In previous sections, we have discussed cutting plane methods that maintain a region that contains the minimizer x^* of f. The proof of such a cutting plane method relies on the following inequality

$$f(x) > f(x^*) \ge f(x) + \langle \nabla f(x), x^* - x \rangle \tag{3.5.4}$$



$B(0,1) \cap B(g, |g|\sqrt{1-\epsilon}) \subset B(x, \sqrt{1-\epsilon})$



$B(0, \sqrt{1-\epsilon|g|^2}) \cap B(g, |g|\sqrt{1-\epsilon}) \subset B(x, \sqrt{1-\sqrt{\epsilon}})$

Figure 3.5.1: The left diagram shows the intersection shrinks at the same rate if only one of the ball shrinks; the right diagram shows the intersection shrinks much faster if two balls shrink at the same absolute amount.

Notice that the left-hand side is a strict inequality (unless we already solved the problem). We infer a halfspace containing the optimum, namely

$$\langle \nabla f(x), x^* - x \rangle \leq 0.$$

As the algorithm proceeds, we find a new point $x^{(\text{new})}$ such that $f(x^{(\text{new})}) < f(x)$. Therefore, the original inequality can be strengthened to

$$f(x^{(new)}) > f(x) + \langle \nabla f(x), x^* - x \rangle$$

or

$$\langle \nabla f(x), x^* - x \rangle < -(f(x) - f(x^{(new)})).$$

This suggests we should move earlier halfspaces and thereby reduce the measure of the next set. We expect merely updating the halfspaces can improve the convergence rate from $1 - \frac{\mu}{L}$ to $1 - \sqrt{\frac{\mu}{L}}$ because of the right diagram in Figure 3.5.1. An efficient way to manage all this information is to directly maintain a region that contains $(x^*, f(x^*))$. Now, we can view the inequality $f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle$ as a cutting plane of the epigraph of f and we do not need to update previous cutting planes anymore.

The next algorithm is an epigraph cutting plane method.

Algorithm 7: `ParabolaMethod`

Input: Initial point $x^{(0)} \in \mathbb{R}^n$ and the strong convexity parameter μ.
$q^{(0)}(y) \leftarrow -\infty$.
for $k = 0, \cdots$ **do**
  Set $q^{(k+\frac{1}{2})}(y) = f(x^{(k)}) + \nabla f(x^{(k)})^\top (y - x^{(k)}) + \frac{\mu}{2} \left\| y - x^{(k)} \right\|^2$.
  Let $q^{(k+1)} = \texttt{maxParabola}(\max(q^{(k+\frac{1}{2})}, q^{(k)}))$ where $\texttt{maxParabola}(q)$ outputs the parabolic function p such that $p(x) \leq q(x)$ for all x, and p maximizes $\min_x p(x)$.
  `// Alternatively, one can use` $x^{(k+\frac{1}{2})} \leftarrow x^{(k)} - \frac{1}{L}\nabla f(x^{(k)})$ `below.`
  Let $x^{(k+\frac{1}{2})} = \texttt{lineSearch}(x^{(k)}, -\nabla f(x^{(k)}))$ where

$$\texttt{lineSearch}(x, y) = \text{argmin}_{z = x + ty \text{ with } t \geq 0} f(z).$$

  Let $x^{(k+1)} = \texttt{lineSearch}(x^{(k+\frac{1}{2})}, c^{(k+1)} - x^{(k+\frac{1}{2})})$ where $c^{(k+1)}$ be the minimizer of $q^{(k+1)}(y)$.
end

As in gradient descent, to avoid using the parameter L, we do a line search from $x^{(k)}$ along the direction $\nabla f(x^{(k)})$.

The subroutine `maxParabola` has an explicit formula [18].

Algorithm 8: $q = \texttt{maxParabola}(\max(q_a, q_b))$

Input: $q_A(x) = v_A + \frac{\mu}{2}\|x - c_A\|_2^2$ and $q_B(x) = v_B + \frac{\mu}{2}\|x - c_B\|_2^2$.
Compute $\lambda = \text{proj}_{[0,1]}\left(\frac{1}{2} + \frac{v_A - v_B}{\mu\|c_A - c_B\|^2}\right)$.
$c_\lambda = \lambda c_A + (1-\lambda)c_B$.
$v_\lambda = \lambda v_A + (1-\lambda)v_B + \frac{\mu}{2}\lambda(1-\lambda)\|c_A - c_B\|^2$.
Output: $q(x) = v_\lambda + \frac{\mu}{2}\|x - c_\lambda\|_2^2$.

Exercise 3.5.4. Show that the formula for `maxParabola` computes the optimal parabola.

The key fact we will be using from the formula is that when $0 < \lambda < 1$, we have

$$v_\lambda = v_A + \frac{\mu}{8}\frac{(\|c_A - c_B\|^2 + \frac{2}{\mu}(v_B - v_A))^2}{\|c_A - c_B\|^2}.$$

This says that the quadratic lower bound improves a lot whenever $\frac{\mu}{2}\|c_A - c_B\|^2 \gg v_B - v_A$ or $\frac{\mu}{2}\|c_A - c_B\|^2 \ll v_B - v_A$. Using this, we can analyze the `ParabolaMethod`.

Theorem 3.5.5. *Assume that f is μ-strongly convex with L-Lipschitz gradient. Let $r_k = f(x^{(k)}) - \min_y q^{(k)}(y)$. Then, we have that*

$$r_{k+1}^2 \le (1 - \sqrt{\frac{\mu}{L}}) \cdot r_k^2.$$

In particular, we have that

$$f(x^{(k+1)}) - f^* \le \frac{1}{2\mu}(1 - \sqrt{\frac{\mu}{L}})^k \|\nabla f(x^{(0)})\|^2.$$

Remark 3.5.6. Note that the squared radius of $\{y : q^{(k)}(y) = f(x^{(k)})\}$ is $\frac{2}{\mu}(f(x^{(k)}) - \min_y q^{(k)}(y))$ because $q^{(k)}(y) = \min_y q^{(k)}(y) + \frac{\mu}{2}\|y - \arg\min_x q^{(k)}(x)\|^2$. Hence, r_k is measuring the squared radius of our quadratic lower bound. To relate to the cutting plane framework, we can view the set as $\{y : q^{(k)}(y) \le f(x^{(k)})\}$ and the measure $\mathcal{V} = f(x^{(k)}) - \min_y q^{(k)}(y)$.

Proof. Fix some k. We write $q^{(k)}(y) = v_A + \frac{\mu}{2}\|y - c_A\|^2$ and $q^{(k+\frac{1}{2})}(y) = v_B + \frac{\mu}{2}\|y - c_B\|^2$ with

$$c_A = c^{(k)}, \quad c_B = x^{(k)} - \frac{\nabla f(x^{(k)})}{\mu} \quad \text{and} \quad v_B = f(x^{(k)}) - \frac{\|\nabla f(x^{(k)})\|^2}{2\mu}.$$

Using the notation in `maxParabola`, we write $q^{(k+1)}(y) = v_\lambda + \frac{\mu}{2}\|y - c_\lambda\|^2$. Note that $r_{k+1}^2 = f(x^{(k+1)}) - v_\lambda$ and $r_k^2 = f(x^{(k)}) - v_A$. Therefore, we have

$$\frac{r_k^2 - r_{k+1}^2}{r_k^2} = \frac{f(x^{(k)}) - f(x^{(k+1)}) + v_\lambda - v_A}{r_k^2}. \tag{3.5.5}$$

To bound the right hand side, it suffices to bound v_A and v_λ. From the description of the algorithm `maxParabola`, we see that there are three cases $\lambda = 0$, $\lambda = 1$ and $0 < \lambda < 1$. We only focus on proving the nontrivial case $\lambda \in (0, 1)$. In this case, we have that

$$\begin{aligned} v_\lambda &= v_A + \frac{\mu}{8}\frac{(\|c_A - c_B\|^2 + \frac{2}{\mu}(v_B - v_A))^2}{\|c_A - c_B\|^2} \\ &= v_A + \frac{\mu}{8}\frac{(\|c_A - c_B\|^2 + \frac{2}{\mu}(f(x^{(k)}) - v_A) - \frac{\|\nabla f(x^{(k)})\|^2}{\mu^2})^2}{\|c_A - c_B\|^2}. \end{aligned}$$

Since $f \ge q^{(k)}$, we have $f(x^{(k)}) \ge q^{(k)}(x^{(k)}) \ge \min_x q^{(k)}(x) = v_A$. Next, we claim that $\|c_A - c_B\|^2 \ge \frac{\|\nabla f(x^{(k)})\|^2}{\mu^2}$. Using these two facts, we can prove that

$$v_\lambda \ge v_A + \frac{\mu}{8}\frac{(\frac{2}{\mu}(f(x^{(k)}) - v_A))^2}{\frac{\|\nabla f(x^{(k)})\|^2}{\mu^2}} = v_A + \frac{\mu \cdot r_k^4}{2\|\nabla f(x^{(k)})\|^2}.$$

Putting this into (3.5.5), we have

$$\begin{aligned}\frac{r_k^2 - r_{k+1}^2}{r_k^2} &\geq \frac{f(x^{(k)}) - f(x^{(k+1)})}{r_k^2} + \frac{\mu \cdot r_k^2}{2\|\nabla f(x^{(k)})\|^2} \\ &\geq \frac{f(x^{(k+\frac{1}{2})}) - f(x^{(k+1)})}{r_k^2} + \frac{\mu \cdot r_k^2}{2\|\nabla f(x^{(k)})\|^2} \\ &\geq \frac{\|\nabla f(x^{(k)})\|^2}{2Lr_k^2} + \frac{\mu \cdot r_k^2}{2\|\nabla f(x^{(k)})\|^2} \\ &\geq \sqrt{\frac{\mu}{L}}\end{aligned} \tag{3.5.6}$$

where the second inequality is due to the fact $x^{(k+\frac{1}{2})}$ is a line search from $x^{(k)}$, the third inequality is due to the assumption on L (Lemma 2.2.4), the last inequality follows from the Cauchy-Schwarz inequality.

For the final conclusion, we note that

$$\begin{aligned}q^{(1)}(y) &= f(x^{(0)}) + \nabla f(x^{(0)})^\top (y - x^{(0)}) + \frac{\mu}{2}\left\|y - x^{(0)}\right\|^2 \\ &= f(x^{(0)}) - \frac{1}{2\mu}\|\nabla f(x^{(0)})\|^2 + \frac{\mu}{2}\left\|y - (x^{(0)} - \frac{1}{\mu}\nabla f(x^{(0)}))\right\|^2.\end{aligned}$$

Hence, we have $v^{(1)} = f(x^{(0)}) - \frac{1}{2\mu}\|\nabla f(x^{(0)})\|^2$ and hence $r_1 \leq f(x^{(1)}) - v^{(1)} \leq \frac{1}{2\mu}\|\nabla f(x^{(0)})\|^2$.

To prove the claim, we note that $x^{(k)}$ is the result of a line search of f between $c^{(k)}$ and some point. Therefore, we have that $\nabla f(x^{(k)}) \perp (x^{(k)} - c^{(k)})$ and hence

$$\|c_A - c_B\|^2 = \|c^{(k)} - x^{(k)} + \frac{\nabla f(x^{(k)})}{\mu}\|^2 \geq \frac{\|\nabla f(x^{(k)})\|^2}{\mu^2}.$$

□

Remark 3.5.7. We can view (3.5.6) as the key equation of the proof above. It shows that the progress is roughly $\frac{\|\nabla f\|^2}{L} + \frac{\mu}{\|\nabla f\|^2}$ where the first term comes from the progress on the function value and the second term comes from "the curvature" of the cutting sphere.

Exercise 3.5.8. Prove the following extension of Lemma 3.5.1: There exists x s.t. $B(0, \sqrt{1-\epsilon|g|^2}) \cap B(g, |g|\sqrt{1-\epsilon}) \subset B(x, \sqrt{1-\sqrt{\epsilon}})$.

Discussion

This section was about the idea of managing cutting planes, and as a byproduct we get an accelerated rate of convergence. As we will see later, standard accelerated gradient descent does not use line search and achieves the rate $1 - \sqrt{\frac{\mu}{L}}$. However, it seems that the use of line search helps in practice and that with a careful implementation, line search can be as cheap as gradient computation. For more difficult problems, one may want to store multiple quadratic lower bounds (see [18]).

3.6 Lower Bounds

In this chapter, we have discussed a few cutting plane methods. In particular, we showed that $O(\min(n, \sqrt{\frac{L}{\mu}})\log(\frac{1}{\epsilon}))$ gradient computations suffice. We conclude this section by showing that $\min(n, \sqrt{\frac{L}{\mu}})$ is in fact optimal among "gradient-based" methods. For convenience the reader may consider $\mu = 1$ and $L = \kappa$.

Theorem 3.6.1. *Fix any $L \geq \mu > 0$. Consider the function*

$$f(x) = -\frac{L-\mu}{4}x_1 + \frac{L-\mu}{8}\sum_{i=1}^{n-1}(x_i - x_{i+1})^2 + \frac{L-\mu}{8}x_1^2 + \frac{\mu}{2}\sum_{i=1}^{n} x_i^2 + \frac{\sqrt{L\mu}-\mu}{4}x_n^2 \tag{3.6.1}$$

which satisfies $\mu \cdot I \preceq \nabla^2 f(x) \preceq L \cdot I$. Assume that our algorithm satisfies $x^{(k)} \in \text{span}(x^{(0)}, \nabla f(x^{(0)}), \cdots, \nabla f(x^{(k-1)}))$ with the initial point $x^{(0)} = 0$. Then, for $k < n$,

$$f(x^{(k)}) - \min_x f(x) \geq (\frac{\mu}{L})^{3/2} (\frac{\sqrt{L/\mu} - 1}{\sqrt{L/\mu} + 1})^{2k} (f(x^{(0)}) - \min_x f(x)).$$

Proof. First, we check the strong convexity and smoothness. Note that $f(x) = -x_1 + \frac{1}{2} x^\top A x$ for some matrix A. Hence, $\nabla^2 f(x) = A$. Hence, we have

$$\theta^\top \nabla^2 f(x) \theta = \frac{L-\mu}{4} \sum_{i=1}^{n-1} (\theta_i - \theta_{i+1})^2 + \frac{L-\mu}{4} \theta_1^2 + \mu \sum_{i=1}^{n} \theta_i^2 + \frac{\sqrt{L\mu} - \mu}{2} \theta_n^2$$

For the upper bound $\nabla^2 f(x) \preceq L \cdot I$, we note that

$$\begin{aligned}
\theta^\top \nabla^2 f(x) \theta &\leq \frac{L-\mu}{4} \sum_{i=1}^{n-1} (2\theta_i^2 + 2\theta_{i+1}^2) + \frac{L-\mu}{4} \theta_1^2 + \mu \sum_{i=1}^{n} \theta_i^2 + (\frac{\sqrt{L\mu} - \mu}{2}) \theta_n^2 \\
&\leq (L-\mu) \sum_{i=1}^{n} \theta_i^2 + \mu \sum_{i=1}^{n} \theta_i^2 \\
&= L \sum_{i=1}^{n} \theta_i^2.
\end{aligned}$$

For the lower bound $\nabla^2 f(x) \succeq \mu \cdot I$, we note that

$$\theta^\top \nabla^2 f(x) \theta \geq \mu \sum_{i=1}^{n} \theta_i^2.$$

To lower bound the error, we note that the gradient at $x^{(0)}$ is of the form $(?, 0, 0, 0, \cdots)$ and hence by the assumption $x^{(1)} = (?, 0, 0, 0, \cdots)$ and $\nabla f(x^{(1)}) = (?, ?, 0, 0, \cdots)$. By induction, only the first k coordinates of $x^{(k)}$ are non-zero.

Now, we compute the minimizer of $f(x)$. Let x^* be the minimizer of $f(x)$. By the optimality condition, we have that

$$\begin{aligned}
-\frac{L-\mu}{4} + \frac{L-\mu}{4}(x_1^* - x_2^*) + \frac{L-\mu}{4} x_1^* + \mu x_1^* &= 0, \\
\frac{L-\mu}{4}(x_i^* - x_{i-1}^*) + \frac{L-\mu}{4}(x_i^* - x_{i+1}^*) + \mu x_i^* &= 0, \text{ for } i \in \{2, 3, \cdots, n-1\} \\
\frac{L-\mu}{4}(x_n^* - x_{n-1}^*) + \frac{\sqrt{L\mu} + \mu}{2} x_n^* &= 0.
\end{aligned}$$

By a direct substitution, we have that $x_i^* = (\frac{\sqrt{L/\mu} - 1}{\sqrt{L/\mu} + 1})^i$ is a solution of the above equation. Now, we note that

$$\|x^{(k)} - x^*\|_2^2 \geq \sum_{i=k+1}^{n} (\frac{\sqrt{L/\mu} - 1}{\sqrt{L/\mu} + 1})^{2i} \geq (\frac{\sqrt{L/\mu} - 1}{\sqrt{L/\mu} + 1})^{2(k+1)}$$

and that

$$\|x^{(0)} - x^*\|_2^2 \leq \sum_{i=1}^{\infty} (\frac{\sqrt{L/\mu} - 1}{\sqrt{L/\mu} + 1})^{2i} = (\frac{\sqrt{L/\mu} - 1}{\sqrt{L/\mu} + 1})^2 \frac{\sqrt{L/\mu} + 1}{2}.$$

Now, by smoothness and by the strong convexity of f, we have

$$\frac{f(x^{(k)}) - f(x^*)}{f(x^{(0)}) - f(x^*)} \geq \frac{\mu}{L} \cdot \frac{\|x^{(k)} - x^*\|_2^2}{\|x^{(0)} - x^*\|_2^2} \geq (\frac{\mu}{L})^{3/2} (\frac{\sqrt{L/\mu} - 1}{\sqrt{L/\mu} + 1})^{2k}.$$

□

Note that this "worst" function naturally appears in many problems. So, it is a problem we need to address. In some sense, the proof points out a common issue of any algorithm which only uses gradient information. Given any

convex function, we construct the dependence graph G on the set of variables x_i by connecting x_i to x_j if $\nabla f(x)_i$ depends on x_j or $\nabla f(x)_j$ depends on x_i (given all other variables). Note that the dependence graph G of the worst function is simply a n vertex path, whose diameter is $n-1$. Also, note that gradient descent can only transmit information from one vertex to another in each iteration. Therefore, it takes at least Ω(diameter) time to solve the problem unless we know the solution is sparse (when L/μ is small). However, we note that this is not a lower bound for all algorithms.

The problem (3.6.1) belongs to a general class of functions called Laplacian systems and it can be solved in nearly linear time using spectral graph theory.

Chapter 4

Reduction

4.1 Equivalences between Oracles

Oracles for Convex Sets

Grötschel, Lovasz and Schrijver [24] introduced five different ways to access convex sets, showed they are equivalent and used them to get polynomial-time algorithms for a variety of combinatorial problems (including the first ones in many cases). Here are four basic oracles for a convex set $K \subseteq \mathbb{R}^n$. The original definitions are more general, allowing for error parameters in each oracle, i.e., approximate versions of all the oracles below. Here we will focus on exact oracles for simplicity.[1]

Definition 4.1.1 (Membership Oracle (MEM))**.** Queried with a vector $y \in \mathbb{R}^n$, the oracle either

- asserts that $y \in K$, or
- asserts that $y \notin K$.

Definition 4.1.2 (Separation Oracle (SEP))**.** Queried with a vector $y \in \mathbb{R}^n$, the oracle either

- asserts that $y \in K$, or
- finds a unit vector $c \in \mathbb{R}^n$ such that $c^T x \leq c^T y$ for all $x \in K$.

Definition 4.1.3 (Validity Oracle (VAL))**.** Queried with a unit vector $c \in \mathbb{R}^n$, the oracle either[2]

- outputs $\max_{x \in K} c^\top x$, or
- asserts that K is empty.

Definition 4.1.4 (Optimization Oracle (OPT))**.** Queried with a unit vector $c \in \mathbb{R}^n$, the oracle either

- finds a vector $y \in \mathbb{R}^n$ such that $y \in K$ and $c^T x \leq c^T y$ for all $x \in K$, or
- asserts that K is empty.

According to the definition, the separation oracle gives strictly more information than the membership oracle and the optimization oracle gives more information than the validity oracle. Depending on the problem, usually one of the oracles will be the preferred way to access the convex set. For example, for the polytope given by $\{Ax \geq b\}$, the separation oracle is the preferred way because the membership oracle takes as much time as the separation oracle, and both validity and optimization involve solving a linear program. On the contrary, the preferred oracle for the convex set $\text{conv}(\{a_i\})$ is the optimization oracle because $\max_{x \in \text{conv}(\{a_i\})} \theta^\top x$ can be solved by checking $x = a_i$ for each i. In combinatorial optimization, many polytopes have exponentially many vertices and constraints but one can use combinatorial structure to solve the optimization problem.

[1] We also omit the fifth oracle VIOL defined by [24], which checks whether the convex set satisfies a given inequality or gives a violating point in the convex set, since this is equivalent to the OPT oracle below up to a logarithmic factor.

[2] We use a slightly different definition than [24] for clarity.

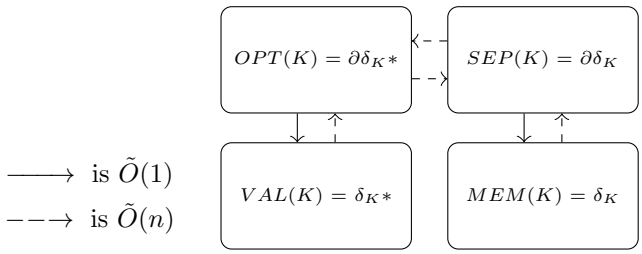


Figure 4.1.1: The relationships among the four oracles for convex sets.

Example 4.1.5. The spanning tree polytope of an undirected graph $G = (V, E)$ is given by

$$P = \{x \in \mathbb{R}_+^E | \sum_{e \in E} x_e = |V| - 1, \sum_{(u,v) \in E \cap (S \times S)} x_{(u,v)} \leq |S| - 1 \quad \forall S \subseteq V\}.$$

The extreme points of this polytope are exactly the indicator vectors of spanning trees of G. Thus, the optimization oracle in this case is to simply find a maximum cost spanning tree.

Exercise 4.1.6. Design a membership oracle for the spanning tree polytope.

Oracles for Convex Functions

Now, we generalize the oracles for convex sets to convex functions. The membership oracle and separation oracle can be generalized as follows

Definition 4.1.7 (Evaluation Oracle (EVAL))**.** Queried with a vector y, the oracle outputs $f(y)$.

The next oracle generalizes gradients to subgradients so that they can be computed for general convex functions. Any vector output by the oracle is a subgradient of the function at the query point.

Definition 4.1.8 (Subgradient Oracle (GRAD))**.** Queried with a vector y, the oracle outputs $f(y)$ and a vector $g \in \mathbb{R}^n$ such that

$$f(x) \geq f(y) + g^\top (x - y) \text{ for all } x \in \mathbb{R}^n \tag{4.1.1}$$

To generalize the validity oracle, we define the convex (Fenchel) conjugate of f:

Definition 4.1.9 (Convex Conjugate)**.** For any function f, we define the convex conjugate

$$f^*(\theta) \stackrel{\text{def}}{=} \sup_{x \in \mathbb{R}^n} \theta^\top x - f(x).$$

Note that f^* is convex because it is the maximum of linear functions. Also, we have $f^*(0) = -\inf_{x \in \mathbb{R}^n} f(x)$. Note that[3] $\delta_K^*(c) = \sup_{x \in K} c^\top x$. Therefore, the validity oracle is simply the evaluation oracle for δ_K^*. The following lemma shows that the optimization oracle is simply the (sub)gradient oracle for δ_K^*.

Lemma 4.1.10. *For any continuously differentiable function f with differentiable f^*, we have that $\nabla f^*(\theta) = \arg\max_x \theta^\top x - f(x)$.*

Proof. Let $x_\theta = \arg\sup_x \theta^\top x - f(x)$. By definition, we have that $f^*(\theta) = \theta^\top x_\theta - f(x_\theta)$ and that $f^*(\eta) \geq \eta^\top x_\theta - f(x_\theta)$ for all η. Therefore,

$$f^*(\eta) \geq f^*(\theta) + x_\theta^\top (\eta - \theta) \text{ for all } \eta.$$

Therefore, $\nabla f^*(\theta) = x_\theta$. □

[3] Recall that $\delta_C(x) = 0$ if $x \in C$ and $+\infty$ otherwise.

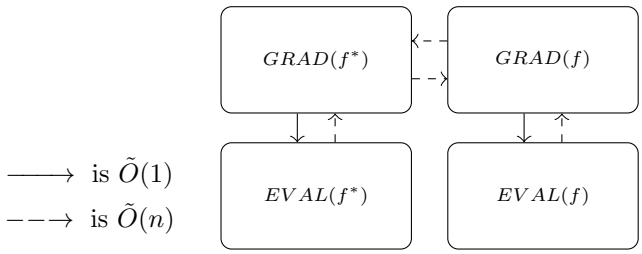


Figure 4.1.2: This illustrates the relationships of oracles for a convex function f and its convex conjugate f^*.

Note that this lemma shows that $\nabla \delta_K^*(\theta) = \arg\max_{x \in K} \theta^\top x$. So, the gradient oracle for δ_K^* is exactly the optimization oracle for K.

Figure 4.1.2 shows the current best reduction between these four function oracles. Note that according to our definition of the GRAD oracle, it also outputs the function value (EVAL). It is not hard to show that you can use $n+1$ calls to evaluation oracle to compute one gradient (using finite difference) and that you can use $\tilde{O}(n)$ calls to gradient oracle of f to compute one gradient of f^* (using the cutting plane method). In the next section (Sec. 4.2), we will show the rest of the reduction. However, it is still open whether all of these reductions are tight:

Open Problem. Prove that it takes $\Omega(n^2)$ calls to the evaluation oracle of f to compute the gradient of f^*.

Convex Conjugate and Equivalences

Here are some examples of conjugates.

Exercise 4.1.11. Show that

- The conjugate of $f(x) = \frac{1}{p}\|x\|_p^p$ is $f^*(\theta) = \frac{1}{q}\|\theta\|_q^q$ where $\frac{1}{p} + \frac{1}{q} = 1$ (with $p, q \geq 1$).
- The conjugate of $f(x) = \langle a, x \rangle - b$ is $f^*(\theta) = \begin{cases} b, & \theta = a \\ +\infty & \text{otherwise} \end{cases}$.
- The conjugate of $f(x) = \sum_i e^{x_i}$ is $f^*(\theta) = \begin{cases} \sum_i \theta_i \log \theta_i - \theta_i & \text{if } \theta_i \geq 0 \text{ for all } i \\ +\infty & \text{otherwise} \end{cases}$.

Exercise 4.1.12. For any x, θ, we have $\theta^\top x \leq f(x) + f^*(\theta)$.

Exercise 4.1.13. Prove that the gradient of f is L-Lipschitz if and only if f^* is $\frac{1}{L}$ strongly convex.

The following lemma shows that indeed one can recover f from f^*.

Lemma 4.1.14 (Involution property)**.** *For any convex function f with a closed epigraph, we have that $f^{**} = f$.*

Proof. Since epi$f$ is closed and convex, Corollary 1.2.5 shows that it is an intersection of halfspaces, i.e.,

$$f(x) = \sup_{\{\theta, b\} \in \mathcal{H}} \theta^\top x - b$$

where $\mathcal{H}$ is the set of supporting planes of epi$f$ and contains all affine lower bounds on f, namely, $f(x) \geq \theta^\top x - b$ for all x. For a fixed θ, any feasible b satisfies $b \geq \theta^\top x - f(x)$ for all x. So, the smallest feasible value satisfies

$$b^* = \sup_x \theta^\top x - f(x) = f^*(\theta).$$

Hence,

$$f(x) = \sup_{\{\theta, b\} \in \mathcal{H}} \theta^\top x - b^* = \sup_\theta \theta^\top x - f^*(\theta) = f^{**}(x).$$

□

Since we can use the gradient, ∇f, to compute the gradient of the dual, ∇f^* (via cutting plane method), the involution property shows that we can do the reverse – use ∇f^* to compute ∇f. Going back to the example about $\text{conv}(\{a_i\})$, since we know how to compute $\max_{x\in\text{conv}(\{a_i\})} \theta^\top x = \delta^*_{\text{conv}(\{a_i\})}(\theta)$, this reduction gives us a way to separate $\text{conv}(\{a_i\})$, or equivalently, to compute the (sub)gradient of $\delta^*_{\text{conv}(\{a_i\})}$. In combinatorial optimization, many convex sets are given by the convex hull for some discrete objects. In many cases, the only known way to do the separation is via such reductions.

Recall that for any linear space X, X^* denotes the dual space, i.e., the set of all linear functions on X and that under mild assumptions[4], we have $X^{**} = X$. Therefore, there are two natural "coordinate systems" to record a convex function, the primal space X and the dual space X^*. Under these "coordinate systems", we have the dual functions f and f^*.

Exercise 4.1.15 (Order reversal)**.** Show that $f \geq g \iff f^* \leq g^*$ (both for all $x \in \mathbb{R}^n$).

Interestingly, the convex conjugate is the unique transformation on convex functions that satisfies both involution and order reversal.

Theorem 4.1.16 ([5])**.** *Given a transformation $\mathcal{T}$ that maps the set of lower-semi-continuous convex functions onto itself such that $\mathcal{T}\mathcal{T}\phi = \phi$ and $\phi \leq \psi \implies \mathcal{T}\phi \geq \mathcal{T}\psi$ for all lower-semi-continuous convex functions ϕ and ψ. Then, $\mathcal{T}$ is essentially the convex conjugate, namely, there is an invertible symmetric linear transformation B, a vector v_0 and a constant C_0 such that*

$$(\mathcal{T}\phi)(x) = \phi^*(Bx + v_0) + v_0^\top x + C_0.$$

4.2 Gradient from Evaluation via Finite Difference

When the function f is in $\mathcal{C}^1$, we can approximate the gradient of a function by calculating a finite difference

$$\frac{\partial f}{\partial x_i} = \frac{f(x + he_i) - f(x)}{h} + o(1)$$

which only takes $n+1$ calls to the evaluation oracle (for computing $f(x), f(x+he_1), \cdots, f(x+he_n)$). The only issue is that the convex function may not be differentiable. However, any convex Lipschitz function is twice differentiable almost everywhere (see the proof below). Therefore, we can simply perturb x with random noise, then apply a finite difference. To see the idea more precisely, we first observe that the norm of the Hessian can be bounded in expectation for a Lipschitz function. Note that this is Lipschitzness of the function, not its gradient. The proof below uses the basic fact that the gradient is defined almost everywhere for Lipschitz functions.

Lemma 4.2.1. *For any L-Lipschitz convex function f defined in a unit ball, we have $\nabla^2 f(x)$ exists almost everywhere and that $\mathbf{E}_{x\in B(0,1)}\|\nabla^2 f(x)\|_F \leq nL$.*

Proof. We will only prove the part $\mathbf{E}_{x\in B(0,1)}\|\nabla^2 f(x)\|_F \leq nL$. Since $\nabla^2 f \succeq 0$ (where defined), we have $\|\nabla^2 f(x)\|_F \leq \text{tr}\nabla^2 f(x)$. Therefore,

$$\int_{B(0,1)} \|\nabla^2 f(x)\|_F dx \leq \int_{B(0,1)} \text{tr}\nabla^2 f(x) dx = \int_{B(0,1)} \Delta f(x) dx.$$

Using Stokes' Theorem, and letting $\nu(x)$ be the normal vector at x, we have

$$\int_{B(0,1)} \Delta f(x) dx = \int_{\partial B(0,1)} \langle \nabla f(x), \nu(x)\rangle \, dx \leq |\partial B(0,1)| \cdot L.$$

Hence, we have

$$\mathbf{E}_{x\in B(0,1)}\|\nabla^2 f(x)\|_F \leq \frac{|\partial B(0,1)|}{|B(0,1)|} L = nL.$$

□

To turn this into an algorithm, we need to develop it a bit further.

[4] The dual of the dual of a vector space X is isomorphic to X if and only if X is a finite-dimensional vector space.

Lemma 4.2.2 ([38]). *Let $B_\infty(x,r) = \{y : \|x-y\|_\infty \le r\}$. For any $0 < r_2 \le r_1$ and any convex function f defined on $B_\infty(x, r_1+r_2)$ with $\|\nabla f(z)\|_\infty \le L$ for any $z \in B_\infty(x, r_1+r_2)$ we have*

$$\mathbf{E}_{y\in B_\infty(x,r_1)}\mathbf{E}_{z\in B_\infty(y,r_2)} \|\nabla f(z) - g(y)\|_1 \le n^{3/2}\frac{r_2}{r_1}L$$

where $g(y)$ is the average of ∇f over $B_\infty(y, r_2)$.

Proof. Let $\omega_i(z) = \langle \nabla f(z) - g(y), e_i\rangle$ for all $i \in [n]$. Then, we have that

$$\int_{B_\infty(y,r_2)} \|\nabla f(z) - g(y)\|_1 \, dz \le \sum_i \int_{B_\infty(y,r_2)} |\omega_i(z)| \, dz.$$

Since $\int_{B_\infty(y,r_2)} \omega_i(z)dz = 0$, the Poincaré inequality for a box (Theorem 4.2.5 below) shows that

$$\begin{aligned}\int_{B_\infty(y,r_2)} |\omega_i(z)| \, dz &\le r_2 \int_{B_\infty(y,r_2)} \|\nabla \omega_i(z)\|_2 \, dz \\ &= r_2 \int_{B_\infty(y,r_2)} \left\|\nabla^2 f(z) e_i\right\|_2 dz \\ \sum_i \int_{B_\infty(y,r_2)} |\omega_i(z)| \, dz &\le \sqrt{n} r_2 \int_{B_\infty(y,r_2)} \left\|\nabla^2 f(z)\right\|_F dz\end{aligned}$$

Since f is convex, we have that $\left\|\nabla^2 f(z)\right\|_F \le \mathrm{tr}\nabla^2 f(z) = \Delta f(z)$. Therefore, we have

$$\begin{aligned}\mathbf{E}_{z\in B_\infty(y,r_2)} \|\nabla f(z) - g(y)\|_1 \, dz &\le \sqrt{n} r_2 \mathbf{E}_{z\in B_\infty(y,r_2)} \Delta f(z) dz \\ &= \sqrt{n} r_2 \Delta h(y)\end{aligned}$$

where $h = \frac{1}{(2r_2)^n} f * \chi_{B_\infty(0,r_2)}$ where $\chi_{B_\infty(0,r_2)}$ is 1 on the set $B_\infty(0,r_2)$ and 0 on outside.

Integrating by parts, we have that

$$\int_{B_\infty(x,r_1)} \Delta h(y) dy = \int_{\partial B_\infty(x,r_1)} \langle \nabla h(y), n(y)\rangle \, dy$$

where $\Delta h(y) = \sum_i \frac{d^2 h}{dx_i^2}(y)$ and $n(y)$ is the normal vector on $\partial B_\infty(x,r_1)$ the boundary of the box $B_\infty(x,r_1)$, i.e. standard basis vectors. Since f is L-Lipschitz with respect to $\|\cdot\|_\infty$ so is h, i.e. $\|\nabla h(z)\|_\infty \le L$. Hence, we have that

$$\mathbf{E}_{y\in B_\infty(x,r_1)} \Delta h(y) \le \frac{1}{(2r_1)^n} \int_{\partial B_\infty(x,r_1)} \|\nabla h(y)\|_\infty \|n(y)\|_1 \, dy \le \frac{1}{(2r_1)^n} \cdot 2n(2r_1)^{n-1} \cdot L = \frac{nL}{r_1}.$$

Therefore, we have that

$$\mathbf{E}_{y\in B_\infty(x,r_1)}\mathbf{E}_{z\in B_\infty(y,r_2)} \|\nabla f(z) - g(y)\|_1 \, dz \le n^{3/2}\frac{r_2}{r_1}L.$$

□

This lemma shows that we can implement an approximate gradient oracle (GRAD) using an evaluation oracle (EVAL) even for non-differentiable functions. By the involution property again, this completes all the reductions in Figure 4.1.2. This argument can also be used to construct an approximate separation oracle for a convex set from a membership oracle.

Theorem 4.2.3. *Let K be a convex body s.t. $B(0,1) \subseteq K \subseteq B(0,R)$. Then, for any $0 < \eta \le 1/2$, we can compute an η-approximate separation oracle for K using $O(n\log(nR/\eta))$ queries to a membership oracle for K.*

By an approximate separation oracle we mean that either the queried point x lies within distance η of K, or the oracle provides a halfspace $c^T y \le c^T x + \eta$ for all points $y \in K$ at distance at least η from the boundary of K. We sketch the proof here. To separate x from the set K, we consider the convex function $h_x : K \to \mathbb{R}$ defined as follows:

$$h_x(y) = -\max\{\alpha : d + \alpha x \in K\}.$$

We then use Lemma 4.2.2 to compute an approximate gradient of h at the origin. For more details, including dependence on approximation parameters, see [38].

Exercise 4.2.4. What is the best possible bound in Lemma 4.2.2?

In the proof of Lemma 4.2.2, we used the following fact applied to the case when Ω is a cube. In this case, the coefficient on the RHS is given by the Cheeger or KLS constant of the cube and is 1. This is an example of an isoperimetric inequality. Such inequalities will play an important role later in this book.

Theorem 4.2.5 (L^1-Poincaré inequality)**.** *Let Ω be connected, bounded and open. Then the following (best-possible) inequality holds for any smooth function $f : \Omega \to \mathbb{R}$:*

$$\left\| f - \frac{1}{|\Omega|} \int_\Omega f(x)\, dx \right\|_{L^1(\Omega)} \le \left(\sup_{S \subset \Omega} \frac{2|S||\Omega \setminus S|}{|\partial S||\Omega|} \right) \|\nabla f\|_{L^1(\Omega)}$$

where the supremum is over all subsets S s.t. S and $\Omega \setminus S$ are both connected.

Exercise 4.2.6. Prove the inequality in Theorem 4.2.5 using the classical coarea formula.

4.3 Gradient from Evaluation via Auto Differentiation

In this section, we give yet another way to compute gradients using evaluations.

Theorem 4.3.1. *Given a function $f : \mathbb{R}^n \to \mathbb{R}$ represented by a circuit. Suppose that $f(x)$ can be computed in time $\mathcal{T}$ (namely, the circuit has $\mathcal{T}$ edges). Then we can compute $\nabla f(x)$ in $O(\mathcal{T})$ time.*

Remark. In practice, the runtime is roughly like $2\mathcal{T}$ assuming we have enough memory. Check out google/jax for a modern implementation.

Before proving it formally, we first go through an example. Consider the function $f(x_1, x_2) = \sin(x_1/x_2) + x_1 x_2$. We use x_i to denote both the input and all intermediate variables. Then, we can write the program in $\mathcal{T} = 6$ steps:

- $x_1 = x_1$, $x_2 = x_2$, $x_3 = x_1/x_2$, $x_4 = \sin(x_3)$, $x_5 = x_1 x_2$, Output $x_6 = x_4 + x_5$.

Note that each step involves computing

$$x_i = f_i(\overbrace{x_1, \cdots, x_{i-1}}^{\text{only few } x_j})$$

with simple functions f_i whose derivatives we know how to compute. The key idea is compute $\frac{\partial f}{\partial x_1}$ not just for the inputs x_1 and x_2, but also for all intermediate variables. Here, we use $\frac{\partial f}{\partial x_i}$ to denote the derivative of f with respect to x_i while fixing $x_1, x_2, \cdots, x_{i-1}$ (and other inputs if x_i is an input). For the example above, suppose we want to compute $\nabla f(\pi, 2)$, we can simply compute first compute all x_i from $i = 1, 2, \cdots, 6$, then $\frac{\partial f}{\partial x_i}$ in the reverse order from $i = 6, 5, \cdots, 1$:

- $x_1 = \pi$, $x_2 = 2$, $x_3 = \pi/2$, $x_4 = \sin(x_3) = 1$, $x_5 = x_1 x_2 = 2\pi$, $x_6 = x_4 + x_5 = 2\pi + 1$.
- $\frac{\partial f}{\partial x_6} = 1$, $\frac{\partial f}{\partial x_5} = \frac{\partial (x_4 + x_5)}{\partial x_5} = 1$, $\frac{\partial f}{\partial x_4} = \frac{\partial (x_4 + x_5)}{\partial x_4} = 1$,
- $\frac{\partial f}{\partial x_3} = \frac{\partial f}{\partial x_4} \frac{\partial x_4}{\partial x_3} = 1 \cdot \cos(x_3) = 0$,
- $\frac{\partial f}{\partial x_2} = \frac{\partial f}{\partial x_3} \frac{\partial x_3}{\partial x_2} + \frac{\partial f}{\partial x_5} \frac{\partial x_5}{\partial x_2} = 0 \cdot (-\frac{x_1}{x_2^2}) + 1 \cdot x_1 = \pi$,
- $\frac{\partial f}{\partial x_1} = \frac{\partial f}{\partial x_3} \frac{\partial x_3}{\partial x_1} + \frac{\partial f}{\partial x_5} \frac{\partial x_5}{\partial x_1} = 0 \cdot (\frac{1}{x_2}) + 1 \cdot x_2 = 2$.

The general case is similar. See `AutoDifferentiation` for the algorithm.

Algorithm 9: `AutoDifferentiation`

Input: a function $f(x_1, x_2, \cdots, x_n)$ given by $f(x_1, x_2, \cdots, x_n) = x_m$ and

$$x_i = f_i(x_1, \cdots, x_{i-1}) \text{ for } i = n+1, n+2, \cdots, m$$

for $i = n+1, n+2, \cdots, m$ **do**
  Compute $x_i = f_i(x_1, \cdots, x_{i-1})$.
end
Let $\frac{\partial f}{\partial x_m} = 1$.
for $i = m-1, \cdots, 1$ **do**
  Let L_i be the set of j such that f_j depends on x_i (i.e. x_j directly depends on x_i).
  Compute $\frac{\partial f}{\partial x_i} = \sum_{j \in L_i} \frac{\partial f}{\partial x_j} \frac{\partial x_j}{\partial x_i}$.
end

Proof of Theorem 4.3.1. We prove by induction that the formula $\frac{\partial f}{\partial x_i} = \sum_{j \in L_i} \frac{\partial f}{\partial x_j} \frac{\partial x_j}{\partial x_i}$ is correct. For the base case $i = m$, we have $f = x_m$ and hence $\frac{\partial f}{\partial x_m} = 1$. For the induction, we let $L_i = \{x_{j_1}, x_{j_2}, \cdots, x_{j_k}\}$. If we fix variables $x_1, x_2, \cdots, x_{i-1}$, then f is a function of x_i (and of other inputs if x_i is an input). Since only $x_{j_1}, x_{j_2}, \cdots, x_{j_k}$ depend on x_i, we can also view f as a function of $x_{j_1}, x_{j_2}, \cdots, x_{j_k}$. More precisely, we have

$$f(x_i) = f(x_{j_1}(x_i, x_{j_{-1}}), x_{j_2}(x_i, x_{j_{-2}}), \cdots, x_{j_k}(x_i, x_{j_{-k}}))$$

where we use $x_{j_{-1}}$ to denote the variables $x_{j_2}, x_{j_3}, \cdots, x_{j_k}$. By chain rule, we have

$$\frac{\partial f}{\partial x_i} = \sum_{j \in L_i} \frac{\partial f}{\partial x_j} \frac{\partial x_j}{\partial x_i}.$$

To bound the runtime, we define the computation graph G be a graph on $x_1, x_2, \cdots, x_m$ such that $i \to j$ if f_j depends on x_i. Note that both the cost of computing f and the cost of our algorithm is $\Theta(m)$ where m is the number of edges in G. This completes the proof. □

We note that Theorem 4.3.1 is surprising even for some simple explicit functions.

Corollary 4.3.2. *If we can compute $f(A) = \det A$ exactly in time $\mathcal{T}$, then we can compute A^{-1} exactly in $O(\mathcal{T})$.*

Proof. Note that $\frac{\partial}{\partial A_{ij}} \det A = \operatorname{adj}(A)_{ji} = \det A \cdot (A^{-1})_{ji}$. Hence, $\nabla \log \det A = A^{-\top}$. Theorem 4.3.1 shows that computing $A^{-\top}$ can be done as fast as $\det A$. □

4.4 Gradient from Evaluation via Complex Step Differentiation

Auto differentiation is great if the function can be computed exactly. However, it does not work well if f can be only approximated or the computation of f involves too many variables. For example, if f is the energy usage of some aircraft design. Then, f can only be computed approximately via simulation and such computation is memory extensive (if we need to store all intermediate variables) and not exact. To introduce complex step differentiation, we recall the formula of finite difference:

- Forward/Backward difference: $\frac{f(x \pm h) - f(x)}{\pm h} = f'(x) + hf''(\zeta)$.
- Central difference: $\frac{f(x+h) - f(x-h)}{2h} = f'(x) + O(h^2) f'''(\zeta)$.

Note that the error analysis above assumes no numerical error involved. The formula involves subtracting two close numbers and dividing by a small number and this step creates lots of error. If we are computing f as floating point,

we have

$$\frac{(1\pm\epsilon)f(x\pm h)-(1\pm\epsilon)f(x)}{\pm h} = f'(x)+O(\frac{\epsilon}{h})f(\zeta_1)+O(h)f''(\zeta_2)$$
$$\frac{(1\pm\epsilon)f(x+h)-(1\pm\epsilon)f(x-h)}{2h} = f'(x)+O(\frac{\epsilon}{h})f(\zeta_1)+O(h^2)f'''(\zeta_2)$$

where ϵ is the floating point precision. Suppose that both f, f', f'', f''' are bounded by constants and suppose $\epsilon = (10)^{-8}$ (single precision floating point), then we should pick $h = \sqrt{\epsilon}$ for the first case and $h = \epsilon^{1/3}$ in the second case. Hence, forward/backward difference gives only accuracy $\epsilon^{1/2} \approx (10)^{-4}$ and the central difference gives only accuracy $\epsilon^{2/3} \approx (10)^{-5}$. In reality, the error will be larger because f'' and f''' usually are large.

The complex complex step differentiation uses the step

$$\frac{\mathrm{Im} f(x+ih)}{h} \approx f'(x)+O(h^2)f''(\zeta).$$

This formula works only for complex analytic functions, but it avoids subtracting numbers. In practice, this really allows us to compute $f'(x)$ close to machine accuracy. In general, ensuring algorithms give close to machine accuracy is important because algorithms often stack on top of each others.

■ 4.5 Optimization from Membership via Sampling

In this chapter, we assumed the oracle can be computed exactly. However, it is still open to determine the best runtime for reductions between *noisy* oracles, where the oracle provides answers to within some bounded error. In particular, the question about minimizing (approximately) convex functions under noisy oracles is an active research area, and the gaps between the lower bound and upper bound for many problems are still quite large ([7, 22, 10] based on [26]). We will see these methods in detail later. For now, to put them in the context of oracles, we just discuss the following theorem.

Theorem 4.5.1 (OPT via sampling)**.** *Let $F(x) = e^{-\alpha c^T x}1_K(x)$ for some convex set K in $\mathbb{R}^n$ and vector $c \in \mathbb{R}^n$. Suppose $\min_{x\in K} c^T x$ is bounded. Let x be a random sample from the distribution with density proportional to $F(x)$. Then,*

$$\mathbf{E}\left(c^T x\right) \le \min_K c^T x + \frac{n}{\alpha}.$$

Proof. We will show that the worst case is when the convex set K is an infinite cone. WLOG assume that the minimum is at $x = 0$. Replace every cross-section of K along c with an $(n-1)$-dimensional ball of the same volume as the cross-section. This does not affect the expectation. Suppose the expectation is $\mathbf{E}(c^T x) = a$. Next, replace the subset of K with $c^T x \le a$ with a cone whose base is the cross-section at a and apex is at zero. This only makes the expectation larger. Next, replace the subset of K on the right of a with the infinite conical extension of the cone to the left of a. Again, this expectation can only be higher.

Now for this infinite cone, we compute, using $y = c^T x$,

$$\begin{aligned}\mathbf{E}\left(c^T x\right) &= \frac{\int_0^\infty y e^{-\alpha y} y^{n-1}\,dy}{\int_0^\infty e^{-\alpha y} y^{n-1}\,dy}\\ &= \frac{1}{\alpha}\frac{\int_0^\infty e^{-y} y^{n}\,dy}{\int_0^\infty e^{-y} y^{n-1}\,dy}\\ &= \frac{1}{\alpha}\frac{n!}{(n-1)!} = \frac{n}{\alpha}.\end{aligned}$$

□

The theorem says that if we sample according to $e^{-\alpha c^T x}$ for $\alpha = n/\varepsilon$, we will get an ε-approximation to the optimum. However, sampling from such a density is not trivial. Instead, we will have to go through a sequence of overlapping distributions, starting with one that is easy to sample and ending with a distribution that is focused close to the minimum. The complexity of sampling is polynomial in the dimension and in a suitable notion of

probabilistic distance between the starting distribution and the target distribution. The sampling algorithm only uses a membership (EVAL) oracle.

Exercise 4.5.2. Extend Theorem 4.5.1 by replacing $c^T x$ with any convex function $f(x)$.

Open Problem. Given an approximately convex function F on unit ball such that $\max_{\|x\|_2 \le 1} |f(x) - F(x)| \le \varepsilon/n$ for some convex function f, how efficiently can we find x in the unit ball such that $F(x) \le \min_{\|x\|_2 \le 1} F(x) + O(\varepsilon)$? The current fastest algorithm takes $O(n^{4.5} \log^{O(1)}(n/\varepsilon))$ calls to the noisy EVAL oracle for F .

4.6 Composite Problem via Duality

Motivation

In this section, we will discuss a few algorithmic applications of duality. Suppose we have a "difficult" convex problem $\min_x f(x)$. Often, we can decompose the difficult problem as $f(x) = g(x) + h(Ax)$ such that $\nabla g^*(x)$ and $\nabla h^*(x)$ are easy to compute. We can compute its dual as follows:

$$\begin{aligned}
\min_x g(x) + h(Ax) &= \min_x \max_\theta g(x) + \theta^\top Ax - h^*(\theta) \\
&= \max_\theta \min_x g(x) + (A^\top \theta)^\top x - h^*(\theta) \\
&= \max_\theta -g^*(-A^\top \theta) - h^*(\theta) \\
&= -\min_\theta g^*(-A^\top \theta) + h^*(\theta)
\end{aligned}$$

where we used $h = h^{**}$ in the first line, the following minimax theorem on the second line, and the definition of g^* on the third line.

Theorem 4.6.1 (Sion's minimax theorem)**.** *Let $X \subset \mathbb{R}^n$ be a compact convex set and $Y \subset \mathbb{R}^m$ be a convex set. If $f : X \times Y \to \mathbb{R} \cup \{+\infty\}$ such that $f(x, \cdot)$ is upper semi-continuous and quasi-concave on Y for all $x \in X$ and $f(\cdot, y)$ is lower semi-continuous and quasi-convex on X for all $y \in Y$. Then, we have*

$$\min_{x \in X} \sup_{y \in Y} f(x, y) = \sup_{y \in Y} \min_{x \in X} f(x, y).$$

Remark. Compactness is necessary. Consider $f(x, y) = x + y$.

We call "$g(x) + h(Ax)$" the primal problem and "$g^*(-A^\top \theta) + h^*(\theta)$" the dual problem. Often, the dual problem gives us some insight on the primal problem. However, we note that there are many ways to split a problem into two and hence many dual problems.

Example: Semidefinite Programming

When you can solve the dual problem, the proof of the optimality of the dual problem often directly gives you the solution of the primal problem, or gives you some way to solve the primal problem efficiently. Here, we use the semidefinite programming as an concrete example. Define $\bullet$ as the inner product of matrices, i.e.,

$$A \bullet B = \sum_{i,j} A_{ij} B_{ij}.$$

Consider the semidefinite programming (SDP) problem:

$$\max_{X \succeq 0} C \bullet X \text{ s.t. } A_i \bullet X = b_i \text{ for } i = 1, 2, \cdots, m \tag{4.6.1}$$

and its dual

$$\min_y b^\top y \text{ s.t. } \sum_{i=1}^m y_i A_i \succeq C \tag{4.6.2}$$

where X, C, A_i are $n \times n$ symmetric matrices and $b, y \in \mathbb{R}^m$. SDP is a generalization of linear programming and is useful for various problems involving matrices. If we apply the current-best cutting plane method naively on the primal problem, we would get $\tilde{O}(n^2(Z+n^4))$ time algorithm for the primal (because there are n^2 variables) and $\tilde{O}(m(Z+n^\omega+m^2))$ for the dual where Z is the total number of non-zeros in A_i. (The term $Z+n^\omega$ is the cost of computing $\sum y_i A_i$ and finding its minimum eigenvalue.) Generally, $n^2 \gg m$ and hence it takes much less time to solve the dual.

We note that

$$\min_{\sum_{i=1}^m y_i A_i \succeq C} b^\top y = \min_{v^\top (\sum_{i=1}^m y_i A_i - C) v \geq 0 \ \forall \|v\|_2 = 1} b^\top y.$$

In each step of the cutting plane method, the (sub)gradient oracle either outputs b or outputs one of the cutting planes

$$v^\top (\sum_{i=1}^m y_i A_i - C) v \geq 0.$$

Let S be the set of all cutting planes used in the algorithm. Then, the proof of the cutting plane method shows that

$$\min_{\sum_{i=1}^m y_i A_i \succeq C} b^\top y = \min_{v^\top (\sum_{i=1}^m y_i A_i - C) v \geq 0 \ \forall v \in \mathcal{S}} b^\top y \pm \varepsilon. \tag{4.6.3}$$

The key idea to obtaining the primal solution is to take the dual of the right hand side (which is an approximate dual of the original problem). Now, we have

$$\begin{aligned}
\min_{v^\top (\sum_{i=1}^m y_i A_i - C) v \geq 0 \ \forall v \in \mathcal{S}} b^\top y &= \min_y \max_{\lambda_v \geq 0} b^\top y - \sum_{v \in \mathcal{S}} \lambda_v v^\top (\sum_{i=1}^m y_i A_i - C) v \\
&= \max_{\lambda_v \geq 0} \min_y C \bullet \sum_{v \in \mathcal{S}} \lambda_v v v^\top + b^\top y - \sum_{i=1}^m y_i \sum_{v \in \mathcal{S}} \lambda_v v v^\top \bullet A_i \\
&= \max_{X = \sum_{v \in \mathcal{S}} \lambda_v v v^\top, \lambda_v \geq 0} \min_y C \bullet X + \sum_{i=1}^m y_i (b_i - X \bullet A_i) \\
&= \max_{X = \sum_{v \in \mathcal{S}} \lambda_v v v^\top, \lambda_v \geq 0, X \bullet A_i = b_i} C \bullet X.
\end{aligned}$$

Note that this is exactly the primal SDP problem, except that we restrict the set of solutions to the form $\sum_{v \in \mathcal{S}} \lambda_v v v^\top$ with $\lambda_v \geq 0$. Also, we can write this problem as a linear program:

$$\max_{\sum_v \lambda_v v^\top A_i v = b_i \text{ for all } i, \lambda_v \geq 0} \sum_v \lambda_v v^\top C v. \tag{4.6.4}$$

Therefore, we can simply solve this linear program and recover an approximate solution for the SDP. By (4.6.3), we know that this is an approximate solution with the same guarantee as the dual SDP.

Now, we analyze the runtime of this algorithm. This algorithm contains two phases: solve the dual SDP via cutting plane method, and solve the primal linear program. Note that each step of the cutting plane method involves finding a separating hyperplane of $\sum_{i=1}^m y_i A_i \succeq C$.

Exercise 4.6.2. Let $\Omega = \{y \in \mathbb{R}^m : \sum_{i=1}^m y_i A_i \succeq C\}$. Show that one can implement the separation oracle in time $O^*(Z+n^\omega)$ via eigenvalue computation.

Therefore, the first phase takes $O^*(m(Z+n^\omega+m^2))$ time in total. Since the cutting plane method takes $O^*(m)$ steps, we have $|\mathcal{S}| = O^*(m)$. In the second phase, we need to solve a linear program (4.6.4) with $O^*(m)$ variables with $O(m)$ constraints. It is known how to solve such linear programs in time $O^*(m^{2.38})$ [15]. Hence, the total cost is dominated by the first phase

$$O^*\left(mZ + mn^\omega + m^3\right).$$

Open Problem 4.6.3. In the first phase, each step involves computing an eigenvector of similar matrices. So, can we use matrix update formulas to decrease the cost per step in the cutting plane to $O^*(Z+n^2)$? Namely, can we solve SDP in time $O^*\left(mZ+m^3\right)$?

Duality and Convex Hull

The problem $\min_x g(x) + h(Ax)$ in general can be solved by a similar trick. To make this geometric picture clearer, we consider its linear optimization version: $\min_{(x,t_1)\in \operatorname{epi} g,(Ax,t_2)\in \operatorname{epi} h} t_1 + t_2$. To simplify the notation, we consider the problem

$$\min_{x\in K_1, Mx\in K_2} c^\top x$$

where $M \in \mathbb{R}^{m\times n}$, and we have convex sets $K_1 \subset \mathbb{R}^n$ and $K_2 \subset \mathbb{R}^m$.

To be concrete, let us consider the following example. Let V_1 be a set of students, V_2 be a set of schools. Each edge $e \in E$ represents a possible choice of a student. Let w_e be the happiness of a school/student if the student is assigned to that school. Suppose that every student can only be assigned to one school and school b can accept c_b students. Then, the problem can be formulated as

$$\max_{x_e\geq 0} \sum_{e\in E} w_e x_e \text{ subject to } \sum_{(a,b)\in E} x_{(a,b)} \leq 1 \ \forall a \in V_1, \sum_{(a,b)\in E} x_{(a,b)} \leq c_b \ \forall b \in V_2.$$

This is the weighted b-matching problem. Typically, the number of students is much more than the number of schools. Therefore, an algorithm with running time linear in the number of students is preferable. To apply our framework, we let

$$K_1 = \{x \in \mathbb{R}^E, x_e \geq 0, \sum_{(a,b)\in E} x_{(a,b)} \leq 1 \ \forall a \in V_1\},$$
$$K_2 = \{y \in \mathbb{R}^{V_2}, y_b \leq c_b\},$$

and $M \in \mathbb{R}^E \to \mathbb{R}^{V_2}$ is the map $(Tx)_b = \sum_{a:(a,b)\in E} x_{(a,b)}$.

To further emphasize its importance, consider some general examples here:

- Linear programming: $\min_{Ax=b,x\geq 0} c^\top x$: $K_1 = \{x \geq 0\}$, $K_2 = \{b\}$ and $M = A$.
- Semidefinite programming: $\min_{A_i\bullet X=b_i, X\succeq 0} C \bullet X$: $K_1 = \{X \succeq 0\}$, $K_2 = \{b\}$ and $M : \mathbb{R}^{n\times n} \to \mathbb{R}^m$ defined by $(MX)_i = A_i \bullet X$.
- Matroid intersection: $\min_{x\in M_1\cap M_2} 1^\top x$: $K_1 = M_1$ and $K_2 = M_2$, $M = I$.
- Submodular minimization: $K_1 = \{y \in \mathbb{R}^n : \sum_{i\in S} y_i \leq f(S) \text{ for all } S \subset [n]\}$, $K_2 = \{y \leq 0\}$, $M = I$.
- Submodular flow: $K_1 = \{\varphi \in \mathbb{R}^E, \ell_e \leq \varphi_e \leq u_e \text{ for all } e \in E\}$, $K_2 = \{y \in \mathbb{R}^V : \sum_{i\in S} y_i \leq f(S) \text{ for all } S \subset [n]\}$, M is the incidence matrix.

In all of these examples, it is easy to compute the gradient of $\delta^*_{K_1}$ and $\delta^*_{K_2}$. For the last three examples, it is not clear how to compute gradient of δ_{K_1} and/or δ_{K_2} directly. Furthermore, in all examples, M maps from a larger space to the same or smaller space. Therefore, it is good to take advantage of the smaller space.

Before [39], the standard way was to use the equivalence of $\nabla\delta^*_{K_1}$ and $\nabla\delta_{K_1}$, and apply cutting plane methods. With the running time of cutting plane methods, such an algorithm usually had theoretical running time at least n^5 and was of little practical value.

Now we rewrite the problem as we did in the beginning:

$$\begin{aligned}
\min_{x\in K_1, Mx\in K_2} c^\top x &= \min_{x\in\mathbb{R}^n} c^\top x + \delta_{K_1}(x) + \delta_{K_2}(Mx) \\
&= \min_{x\in\mathbb{R}^n} \max_{\theta\in\mathbb{R}^m} c^\top x + \delta_{K_1}(x) + \theta^\top Mx - \delta^*_{K_2}(\theta) \\
&= \max_{\theta\in\mathbb{R}^m} \min_{x\in\mathbb{R}^n} c^\top x + \delta_{K_1}(x) + \theta^\top Mx - \delta^*_{K_2}(\theta) \\
&= \max_{\theta\in\mathbb{R}^m} -\delta^*_{K_1}(-c - M^\top\theta) - \delta^*_{K_2}(\theta).
\end{aligned}$$

Taking the dual has two benefits. First, the number of variables is smaller. Second, the gradient oracle is something we can compute efficiently. Hence, cutting plane methods can be used to solve it in $O^*(m\mathcal{T} + m^3)$ where $\mathcal{T}$ is the time to evaluate $\nabla\delta^*_{K_1}$ and $\nabla\delta^*_{K_2}$. The only problem left is to recover the primal x.

The key observation is the following lemma:

Lemma 4.6.4. *Let $x_i \in K_1$ be the set of points output by the oracle $\nabla\delta^*_{K_1}$ during the cutting plane method. Define $y_i \in K_2$ similarly. Suppose that the cutting plane method ends with the guarantee that the additive error is less than ε. Then, we have that*

$$\min_{x\in K_1, Tx\in K_2} c^\top x \le \min_{x\in \widetilde{K_1}, Tx\in \widetilde{K_2}} c^\top x \le \min_{x\in K_1, Tx\in K_2} c^\top x + \varepsilon$$

where $\widetilde{K_1} = \operatorname{conv}(x_i)$ and $\widetilde{K_2} = \operatorname{conv}(y_i)$.

Proof. Let θ_i be the set of directions queried by the oracle for $\nabla\delta^*_{K_1}$ and φ_i be the directions queried by the oracle for $\nabla\delta^*_{K_2}$. We claim that $x_i \in \nabla\delta^*_{\widetilde{K_1}}(\theta_i)$ and $y_i \in \nabla\delta^*_{\widetilde{K_2}}(\varphi_i)$. Having this, the algorithm cannot distinguish between $\widetilde{K_1}$ and K_1, and between $\widetilde{K_2}$ and K_2. Hence, the algorithm runs exactly the same, i.e., uses the same sequence of points. Therefore, we get the same value $c^\top x$. However, by the guarantee of cutting plane method, we have that

$$\min_{x\in \widetilde{K_1}, Tx\in \widetilde{K_2}} c^\top x \le c^\top x \le \min_{x\in K_1, Tx\in K_2} c^\top x + \varepsilon.$$

To prove the claim, we note that $x_i \in \nabla\delta^*_{\widetilde{K_1}}(\theta_i)$. Note that $\widetilde{K_1} \subset K_1$ and hence $\min_{x\in\widetilde{K_1}} \theta_i^\top x \ge \min_{x\in K_1} \theta_i^\top x$. Also, note that

$$x_i = \arg\min_{x\in K_1} \theta_i^\top x \in \widetilde{K_1}$$

and hence $\min_{x\in\widetilde{K_1}} \theta_i^\top x \le \min_{x\in K_1} \theta_i^\top x$. Therefore, we have that $\min_{x\in\widetilde{K_1}} \theta_i^\top x = \min_{x\in K_1} \theta_i^\top x$. Therefore, $x_i \in \arg\min_{x\in K_1} \theta_i^\top x$. This proves the claim for $\widetilde{K_1}$. The proof for $\widetilde{K_2}$ is the same. □

This reduces the problem into the form $\min_{x\in\widetilde{K_1}, Tx\in\widetilde{K_2}} c^\top x$. For the second phase, we let $z_i = Mx_i \in \mathbb{R}^m$. Then, we have

$$\begin{aligned}\min_{x\in\widetilde{K_1}, Mx\in\widetilde{K_2}} c^\top x &= \min_{t_i\ge 0, s_i\ge 0, M\sum_i t_i x_i = \sum_i s_i y_i} c^\top \Big(\sum_i t_i x_i\Big) \\ &= \min_{t_i\ge 0, s_i\ge 0, \sum_i t_i z_i = \sum_i s_i y_i} \sum t_i \cdot c^\top x_i.\end{aligned}$$

Note that it takes $O^*(mZ)$ time to write down this linear program where Z is the number of non-zeros in M. Next, we note that this linear program has $O^*(m)$ variables and m constraints. Therefore, we can solve it in $O^*(m^{2.38})$ time.

Therefore, the total running time is

$$O^*(m(\mathcal{T} + m^2) + (mZ + m^{2.38})).$$

To conclude, we have the following theorem.

Theorem 4.6.5. *Given convex sets $K_1 \subset \mathbb{R}^n$ and $K_2 \subset \mathbb{R}^m$ with $m \le n$ and a matrix $M : \mathbb{R}^n \to \mathbb{R}^m$ with Z non-zeros, let $\mathcal{T}$ be the cost to compute $\nabla\delta^*_{K_1}$ and $\nabla\delta^*_{K_2}$. Then, we can solve the problem*

$$\min_{x\in K_1, Mx\in K_2} c^\top x$$

in time $O^*(m\mathcal{T} + mZ + m^3)$.

Remark. We hid all sorts of terms in the log term hidden in O^* such as the diameter of the set. Also this is the number of arithmetic operations, not the bit complexity.

Going back to the school/student problem, this algorithm gives a running time of

$$O^*(|V_2||E| + |V_2|^3)$$

which is linear in the number of students!

In general, this statement says that if we can split a convex problem into two parts, with both being easy to solve and one part has fewer variables, then we can solve it in time proportional to the smaller dimension.

Exercise 4.6.6. How fast can we solve $\min_{x\in\cap_{i=1}^k K_i} c^\top x$ given the oracles $\nabla\delta^*_{K_i}$?

Chapter 5

Geometrization and Optimization

In this chapter, we study techniques that further exploit the geometry of convex functions and associated norms. Many of these techniques are effective in practice for large scale problems.

5.1 Mirror Descent

The cutting plane method is well-suited for minimizing non-smooth convex functions with high accuracy. However, its relatively large polynomial time complexity and quadratic space requirement are not favorable for large scale problems. Here we discuss a different approach to minimize a non-smooth function with low accuracy.

Subgradient method (a.k.a. projected GD)

In this section, we consider the constrained non-smooth minimization problem $\min_{x\in\mathcal{D}} f(x)$. Recall how to define gradient for non-smooth functions:

Definition 5.1.1. For any convex function f, we define $\partial f(x)$ be the set of vectors g such that

$$f(y) \geq f(x) + g^\top (y - x) \text{ for all } y \in \mathbb{R}^n.$$

Algorithm 10: `SubgradientMethod`

Input: Initial point $x^{(0)} \in \mathbb{R}^n$, step size $h > 0$.
for $k = 0, 1, \cdots, T-2$ **do**
  Pick any $g^{(k)} \in \partial f(x^{(k)})$.
  $y^{(k+1)} \leftarrow x^{(k)} - h \cdot g^{(k)}$.
  $x^{(k+1)} \leftarrow \pi_{\mathcal{D}}(y^{(k+1)})$ where $\pi_{\mathcal{D}}(y) = arg\min_{x\in\mathcal{D}} \|x - y\|_2$
end
return $\frac{1}{T}\sum_{k=0}^{T-1} x^{(k)}$.

Note that we return the average of all iterates, rather than the last iterate, as the average is better behaved in the worst case. To analyze the algorithm, we first need the following Pythagorean theorem. This shows that when we project a point to a convex set, we get closer to any point in the convex set, and how much closer depends on how much we move.

Lemma 5.1.2 (Pythagorean Theorem)**.** *Given a convex set $\mathcal{D}$ and a point y, let $\pi(y) = arg\min_{x\in\mathcal{D}} \|x - y\|_2$. For any $z \in \mathcal{D}$, we have that*

$$\|z - \pi(y)\|_2^2 + \|\pi(y) - y\|_2^2 \leq \|z - y\|_2^2.$$

Proof. Let $h(t) = \|(\pi(y) + t(z - \pi(y))) - y\|^2$. Since $\pi(y)$ is the closest point to y, $h(t)$ must be minimized at $t = 0$. So, we have that $h'(0) \geq 0$. i.e.,

$$(\pi(y) - y)^\top (z - \pi(y)) \geq 0.$$

(If y is in the set, then the statement is trivial since $\pi(y) = y$. Otherwise, since $\pi(y)$ is the closest point in $\mathcal{D}$ to y, the hyperplane normal to $\pi(y) - y$ through $\pi(y)$ separates y from z.). Hence,

$$\begin{aligned}\|z - y\|_2^2 &= \|z - \pi(y) + \pi(y) - y\|_2^2 \\ &= \|z - \pi(y)\|_2^2 + 2(z - \pi(y))^\top(\pi(y) - y) + \|\pi(y) - y\|_2^2 \\ &\geq \|z - \pi(y)\|_2^2 + \|\pi(y) - y\|_2^2.\end{aligned}$$

□

Now, we are ready to analyze the subgradient method. It basically involves tracking the squared distance to the optimum, $\|x^{(k+1)} - x^*\|_2^2$.

Theorem 5.1.3. *Let f be a convex function that is G-Lipschitz in ℓ_2 norm. After T steps, the subgradient method outputs a point x such that*

$$f(x) \leq f(x^*) + \frac{\|x^{(0)} - x^*\|_2^2}{2hT} + \frac{h}{2}G^2$$

where x^* is any point that minimizes f over $\mathcal{D}$.

Remark. If the distance $\|x^{(0)} - x^*\|$ and the Lipschitz constant G are known, we can pick $h = \frac{\|x^{(0)} - x^*\|_2}{G\sqrt{T}}$ and get

$$f(x) \leq f(x^*) + \frac{G \cdot \|x^{(0)} - x^*\|_2}{\sqrt{T}}.$$

Proof. Let x^* be any point that minimizes f. Then, we have

$$\begin{aligned}\|x^{(k+1)} - x^*\|_2^2 &= \|\pi_{\mathcal{D}}(y^{(k+1)}) - x^*\|_2^2 \\ &\leq \|y^{(k+1)} - x^*\|_2^2 \\ &= \|x^{(k)} - hg^{(k)} - x^*\|_2^2 \\ &= \|x^{(k)} - x^*\|_2^2 - 2h\left\langle g^{(k)}, x^{(k)} - x^*\right\rangle + h^2\|g^{(k)}\|_2^2.\end{aligned}$$

where we used Lemma 5.1.2 in the inequality. Using the definition of subgradient, we have that

$$f(x^*) \geq f(x^{(k)}) + \left\langle g^{(k)}, x^* - x^{(k)}\right\rangle.$$

Therefore, we have that

$$\begin{aligned}\|x^{(k+1)} - x^*\|_2^2 &\leq \|x^{(k)} - x^*\|_2^2 - 2h \cdot (f(x^{(k)}) - f(x^*)) + h^2\|g^{(k)}\|_2^2 \\ &\leq \|x^{(k)} - x^*\|_2^2 - 2h \cdot (f(x^{(k)}) - f(x^*)) + h^2G^2.\end{aligned}$$

Note that this equation shows that if the error $f(x^{(k)}) - f(x^*)$ is larger, then we move faster towards the optimum. Rearranging the terms, we have

$$f(x^{(k)}) - f(x^*) \leq \frac{1}{2h}\left(\|x^{(k)} - x^*\|_2^2 - \|x^{(k+1)} - x^*\|_2^2\right) + \frac{h}{2}G^2.$$

We sum over all iterations, to get

$$\begin{aligned}\frac{1}{T}\sum_{k=0}^{T-1}(f(x^{(k)}) - f(x^*)) &\leq \frac{1}{T} \cdot \frac{1}{2h}(\|x^{(0)} - x^*\|_2^2 - \|x^{(T)} - x^*\|_2^2) + \frac{h}{2}G^2 \\ &\leq \frac{\|x^{(0)} - x^*\|_2^2}{2hT} + \frac{h}{2}G^2.\end{aligned}$$

The result follows from observing that for a convex function,

$$f\left(\frac{1}{T}\sum_{k=0}^{T-1}x^{(k)}\right) - f(x^*) \leq \frac{1}{T}\sum_{k=0}^{T-1}(f(x^{(k)}) - f(x^*)).$$

□

Intuition of Mirror Descent

Consider using the subgradient method above to minimize $f(x) = \sum_i |x_i|$ over the unit ball $B(0, 1)$. The subgradient of f is given by $\text{sign}(x)$ (assuming $x_i \neq 0$ for all i). Therefore, we have that the Lipshitz constant is bounded by the Euclidean norm of a vector with ± 1 entries, i.e., $G = \sqrt{n}$ and the set is contained in a ball of radius $R = 1$. Hence, the convergence rate is $\sqrt{\frac{n}{k}}$ which grows with the dimension. Intuitively, the dimension dependence comes from the fact that we must take a tiny step size to avoid affecting all n variables. For example, if we take a constant step size h and start at the point $x = (1, \frac{1}{n}, \frac{1}{n}, \cdots, \frac{1}{n})$, then we will get a point with x_i constant in all directions $i \neq 1$, and this increases the function f dramatically from $\Theta(1)$ to $\Theta(nh)$. Therefore, to get constant error, we need to take step size $h \leq \frac{1}{n}$ and hence we need $\Omega(n)$ iterations.

Conceptually, the step $x = x - \eta g$ does not make sense either. Imagine the problem is infinitely dimensional. Note that $f(x) < +\infty$ for any

$$x \in \ell_1 = \{x \in \mathbb{R}^{\mathbb{N}} : \sum_i |x_i| < \infty\}.$$

On the other hand, its gradient g lives in ℓ_∞ space; the dual space of ℓ_1 is ℓ_∞ (in general, ℓ_p is dual to ℓ_q where $(1/p) + (1/q) = 1$). Since x and g are not in the same space, the term $x - \eta g$ does not make sense. More precisely, we have the following tautology (directly follows from the definition of dual space, namely the set of all linear maps in the original space).

Fact 5.1.4. *Given any Banach space $\mathcal{D}$ over the reals and a continuously differentiable function f from $\mathcal{D}$ to $\mathbb{R}$, its gradient $\nabla f(x) \in \mathcal{D}^*$ for any x.*

In general, if the function f is on the primal space $\mathcal{D}$, then the gradient g lives in the dual space $\mathcal{D}^*$. Therefore, we need to map x from the primal space $\mathcal{D}$ to the dual space $\mathcal{D}^*$, update its position, then map the point back to the original space $\mathcal{D}$.

In fact, Lemma 5.1.4 gives us one such map, ∇f. Consider the following algorithm: Starting at x. We use $\nabla f(x)$ to map x to the dual space $y = \nabla f(x)$. Then, we apply the gradient step on the dual $y^{(\text{new})} = y - \nabla f(x)$ and map it back to the primal space, namely finding $x^{(\text{new})}$ such that $\nabla f(x^{(\text{new})}) = y^{(\text{new})}$. Note that $y^{(\text{new})} = 0$ and hence $x^{(\text{new})}$ is exactly a minimizer of f. So, if we can map it back, this algorithm solves the problem in one step. Unfortunately, the task of mapping it back is exactly our original problem.

Instead of using the same f, mirror descent uses some other convex function Φ, called the *mirror map*. For constrained problems, the mirror map may not bring the point back to a point in $\mathcal{D}$. Naively, one may consider the algorithm

$$\nabla\Phi(y^{(t+1)}) = \nabla\Phi(x^{(t)}) - h \cdot \nabla f(x^{(t)}),$$
$$x^{(t+1)} = \arg\min_{x \in \mathcal{D}} \|x - y^{(t+1)}\|_2.$$

Note that the first step of finding $y^{(t+1)}$ involves solving an optimization problem. We will show how to do this (See 5.1.1) but not for this version. This algorithm does not consider the geometry of the function Φ. Instead we should measure the distance according to Φ:

Definition 5.1.5. For any strictly convex function Φ, we define the Bregman divergence as

$$D_\Phi(y, x) = \Phi(y) - \Phi(x) - \langle \nabla\Phi(x), y - x \rangle .$$

Note that $D_\Phi(y, x)$ is the error of the first order Taylor expansion of Φ at x. Due to the convexity of Φ, we have that $D_\Phi(y, x) \geq 0$. Also, we note that $D_\Phi(y, x)$ is convex in y, but not necessarily in x.

Example 5.1.6. $D_\Phi(y, x) = \|y - x\|^2$ for $\Phi(x) = \|x\|^2$. $D_\Phi(y, x) = \sum_i y_i \log \frac{y_i}{x_i} - \sum y_i + \sum x_i$ for $\Phi(x) = \sum x_i \log x_i$.

Algorithm 11: `MirrorDescent`

Input: Initial point $x^{(0)} = \arg\min_{x\in\mathcal{D}} \Phi(x)$, step size $h > 0$.
for $k = 0, 1, \cdots, T-2$ **do**
  Pick any $g^{(k)} \in \partial f(x^{(k)})$.
  `// As shown in (5.1.1), the next 2 steps can be implemented by an optimization over` D_Φ.
  Find $y^{(k+1)}$ such that $\nabla\Phi(y^{(k+1)}) = \nabla\Phi(x^{(k)}) - h \cdot g^{(k)}$.
  $x^{(k+1)} \in \pi_{\mathcal{D}}^{\Phi}(y^{(k+1)})$ where $\pi_{\mathcal{D}}^{\Phi}(y) = \arg\min_{x\in\mathcal{D}} D_\Phi(x, y)$.
end
return $\frac{1}{T}\sum_{k=0}^{T-1} x^{(k)}$.

The Mirror Descent step can be written as follows. In the second step below, we use the fact that the optimization is only over x (and hence all terms in y can be ignored):

$$\begin{aligned} x^{(k+1)} &= \arg\min_{x\in\mathcal{D}} D_\Phi(x, y^{(k+1)}) \\ &= \arg\min_{x\in\mathcal{D}} \Phi(x) - \Phi(y^{(k+1)}) - \nabla\Phi(y^{(k+1)})^\top (x - y^{(k+1)}) \\ &= \arg\min_{x\in\mathcal{D}} \Phi(x) - \nabla\Phi(y^{(k+1)})^\top x \\ &= \arg\min_{x\in\mathcal{D}} \Phi(x) - \nabla\Phi(x^{(k)})^\top x + hg^{(k)\top} x \\ &= \arg\min_{x\in\mathcal{D}} hg^{(k)\top} x + D_\Phi(x, x^{(k)}). \end{aligned} \tag{5.1.1}$$

Note that this is a natural generalization of $x^{(k+1)} = \arg\min_{x\in\mathcal{D}} hg^{(k)\top} x + \|x - x^{(k)}\|^2$.

Analysis of Mirror Descent

Lemma 5.1.7 (Pythagorean Theorem)**.** *Given a convex set $\mathcal{D}$ and a point y, let $\pi(y) = \arg\min D_\Phi(x, y)$. For any $z \in \mathcal{D}$, we have that*

$$D_\Phi(z, \pi(y)) + D_\Phi(\pi(y), y) \le D_\Phi(z, y).$$

Proof. Let $h(t) = D_\Phi(\pi(y) + t(z - \pi(y)), y)$. Since $h(t)$ is minimized at $t = 0$, we have that $h'(0) \ge 0$. Hence, we have

$$h'(0) = (\nabla\Phi(\pi(y)) - \nabla\Phi(y))^\top (z - \pi(y)) \ge 0.$$

Hence,

$$\begin{aligned} D_\Phi(z, y) &= D_\Phi(z, \pi(y)) + 2(z - \pi(y))^\top (\nabla\Phi(\pi(y)) - \nabla\Phi(y)) + D_\Phi(\pi(y), y) \\ &\ge D_\Phi(z, \pi(y)) + D_\Phi(\pi(y), y). \end{aligned}$$

□

Theorem 5.1.8. *Let f be a G-Lipschitz convex function on $\mathcal{D}$ with respect to some norm $\|\cdot\|$. Let Φ be a ρ-strongly convex function on $\mathcal{D}$ with respect to $\|\cdot\|$ with squared diameter $R^2 = \sup_{x\in\mathcal{D}} \Phi(x) - \Phi(x^{(0)})$. Then, mirror descent outputs x such that*

$$f(x) - \min_x f(x) \le \frac{R^2}{hT} + \frac{h}{2\rho} G^2.$$

Remark 5.1.9. We say a function f is ρ strongly convex with respect to the norm $\|\cdot\|$ if for any x, y, we have

$$f(y) \ge f(x) + \langle \nabla f(x), y - x\rangle + \frac{\rho}{2}\|y - x\|^2.$$

The usual strong convexity is with respect to the Euclidean norm.

Remark 5.1.10. Picking $h = \frac{R}{G}\sqrt{\frac{2\rho}{T}}$, we get the rate $f(x) \le \min_x f(x) + GR\sqrt{\frac{2}{\rho T}}$.

Proof. This proof is a complete "mirror" of the proof in Theorem 5.1.3.

$$\begin{aligned} D_\Phi(x^*, x^{(k+1)}) &\le D_\Phi(x^*, y^{(k+1)}) - D_\Phi(x^{(k+1)}, y^{(k+1)}) \\ &= D_\Phi(x^*, x^{(k)}) + (x^* - x^{(k)})^\top (\nabla\Phi(x^{(k)}) - \nabla\Phi(y^{(k+1)})) + D_\Phi(x^{(k)}, y^{(k+1)}) - D_\Phi(x^{(k+1)}, y^{(k+1)}) \\ &= D_\Phi(x^*, x^{(k)}) - h \cdot \left\langle g^{(k)}, x^{(k)} - x^* \right\rangle + D_\Phi(x^{(k)}, y^{(k+1)}) - D_\Phi(x^{(k+1)}, y^{(k+1)}) \end{aligned}$$

where we used Lemma 5.1.7 in the inequality. Using the definition of subgradient, we have that

$$f(x^*) \ge f(x^{(k)}) + \left\langle g^{(k)}, x^* - x^{(k)} \right\rangle.$$

Therefore, we have that

$$D_\Phi(x^*, x^{(k+1)}) \le D_\Phi(x^*, x^{(k)}) - h \cdot (f(x^{(k)}) - f(x^*)) + D_\Phi(x^{(k)}, y^{(k+1)}) - D_\Phi(x^{(k+1)}, y^{(k+1)}).$$

Next we note that

$$\begin{aligned} & D_\Phi(x^{(k)}, y^{(k+1)}) - D_\Phi(x^{(k+1)}, y^{(k+1)}) \\ =& \Phi(x^{(k)}) - \Phi(x^{(k+1)}) - \nabla\Phi(y^{(k+1)})^\top (x^{(k)} - x^{(k+1)}) \\ \le& \left\langle \nabla\Phi(x^{(k)}) - \nabla\Phi(y^{(k+1)}), x^{(k)} - x^{(k+1)} \right\rangle - \frac{\rho}{2}\|x^{(k)} - x^{(k+1)}\|^2 \\ =& h \cdot \left\langle g^{(k)}, x^{(k)} - x^{(k+1)} \right\rangle - \frac{\rho}{2}\|x^{(k)} - x^{(k+1)}\|^2 \\ \le& hG\|x^{(k)} - x^{(k+1)}\| - \frac{\rho}{2}\|x^{(k)} - x^{(k+1)}\|^2 \\ \le& \frac{(hG)^2}{2\rho}. \end{aligned}$$

Hence, we have

$$D_\Phi(x^*, x^{(k+1)}) \le D_\Phi(x^*, x^{(k)}) - h \cdot (f(x^{(k)}) - f(x^*)) + \frac{(hG)^2}{2\rho}.$$

Rearranging the terms, we have

$$f(x^{(k)}) - f(x^*) \le \frac{1}{h}\left(D_\Phi(x^*, x^{(k)}) - D_\Phi(x^*, x^{(k+1)})\right) + \frac{hG^2}{2\rho}.$$

Summing over all iterations, we have

$$\begin{aligned} \frac{1}{T}\sum_{k=0}^{T-1}(f(x^{(k)}) - f(x^*)) &\le \frac{1}{T} \cdot \frac{1}{h}(D_\Phi(x^*, x^{(0)}) - D_\Phi(x^*, x^{(T)})) + \frac{h}{2\rho}G^2 \\ &\le \frac{1}{hT}D_\Phi(x^*, x^{(0)}) + \frac{h}{2\rho}G^2 \\ &\le \frac{R^2}{hT} + \frac{h}{2\rho}G^2. \end{aligned}$$

Using the fact that $x^{(0)}$ was chosen as a minimum of $\Phi(x)$,

$$\begin{aligned} D_\Phi(x^*, x^{(0)}) =& (\Phi(x^*) - \Phi(x^{(0)}) - \left\langle \nabla\Phi(x^{(0)}), x^* - x^{(0)} \right\rangle) \\ =& \Phi(x^*) - \Phi(x^{(0)}) \\ \le& R^2 \end{aligned}$$

The result follows from the convexity of f. □

Multiplicative Weight Update

In this section, we discuss mirror descent under the map $\Phi(x) = \sum x_i \log x_i$ with the convex set being the simplex $\mathcal{D} = \{x_i \ge 0, \sum x_i = 1\}$.

Step Formula As we showed in (5.1.1), we have that

$$x^{(k+1)} = \arg\min_{x\in\mathcal{D}} hg^{(k)\top}x + D_\Phi(x, x^{(k)}).$$

Note that

$$\begin{aligned} D_\Phi(x, x^{(k)}) &= \sum x_i \log x_i - \sum x_i^{(k)} \log x_i^{(k)} - \sum_i (1 + \log x_i^{(k)})(x_i - x_i^{(k)}) \\ &= \sum x_i \log \frac{x_i}{x_i^{(k)}} \end{aligned}$$

where we used that $\sum_i x_i = \sum_i x_i^{(k)}$. Hence, the step is simply

$$x^{(k+1)} = \arg \min_{\sum x_i = 1, x_i \geq 0} hg^{(k)\top}x + \sum x_i \log \frac{x_i}{x_i^{(k)}}.$$

Note that the optimality condition is given by

$$hg_i^{(k)} + \log \frac{x_i^{(k+1)}}{x_i^{(k)}} + 1 - \lambda = 0$$

for some Lagrangian multiplier λ. Rewriting, we have

$$x_i^{(k+1)} = e^{-hg_i^{(k)}} x_i^{(k)} / Z$$

for some normalization constant Z. Note that this algorithm multiplies the current x with a multiplicative factor and hence it is also called multiplicative weight update.

Strong Convexity To bound the strong convexity parameter, we note that

$$\Phi(y) - \Phi(x) - \langle \nabla\Phi(x), y - x\rangle = \frac{1}{2}(y-x)^\top \nabla^2\Phi(\zeta)(y-x)$$

for some ζ between x and y. Since $\nabla^2\Phi(\zeta) = \frac{1}{\zeta}$, we have

$$(y-x)^\top \nabla^2\Phi(\zeta)(y-x) = \sum \frac{(y_i - x_i)^2}{\zeta_i} \geq \frac{(\sum_i |y_i - x_i|)^2}{\sum_i \zeta_i} = (\sum_i |y_i - x_i|)^2$$

where we used that $\sum_i x_i = \sum_i y_i = 1$ and so $\sum_i \zeta_i = 1$. Hence,

$$\Phi(y) - \Phi(x) - \langle \nabla\Phi(x), y - x\rangle \geq \frac{1}{2}\|y - x\|_1^2.$$

Therefore, Φ is 1-strongly convex in $\|\cdot\|_1$. Hence, $\rho = 1$.

Diameter Direct calculation shows that $-\log n \leq \Phi(x) \leq 0$. We start at $x^{(0)} = \frac{1}{n}(1, \ldots, 1)^T$. Hence, $R^2 = \log n$.

Result

Theorem 5.1.11. *Let f be a 1-Lipschitz function on $\|\cdot\|_1$. Then, mirror descent with the mirror map $\Phi(x) = \sum_i x_i \log x_i$.*

$$f(x^T) - \min_x f(x) \leq \sqrt{\frac{2\log n}{T}}.$$

In comparison, projected gradient descent had the bound of $\sqrt{\frac{n}{T}}$.

5.2 Frank–Wolfe

Mirror descent is not suitable for all spaces. The guarantee of mirror descent crucially depends on the fact that there is a 1-strongly convex mirror map Φ such that $\max_x \Phi(x) - \min_x \Phi(x)$ is small on the domain. For some domains such as $\{x : \|x\|_\infty \le 1\}$, the range $\max_x \Phi(x) - \min_x \Phi(x)$ can be large.

Lemma 5.2.1. *Let Φ be a 1-strongly convex function on $\mathbb{R}^n$ over $\|\cdot\|_\infty \le 1$. Then,*

$$\max_{\|x\|_\infty \le 1} \Phi(x) \ge \min_{\|x\|_\infty \le 1} \Phi(x) + \frac{n}{2}.$$

Proof. By the strong convexity of Φ, we have that

$$\begin{aligned}
\mathbf{E}_{s_k \in \{\pm 1\}\ \forall k \in [n]} \Phi(s_1, \cdots, s_n) &\ge \mathbf{E}_{s_k \in \{\pm 1\}\ \forall k \in [n-1]} \Phi(s_1, \cdots, s_{n-1}, 0) + \mathbf{E}_{s_n} \langle \nabla \Phi(s_1, \cdots, s_{n-1}, 0), s_n \rangle + \frac{1}{2} \\
&= \mathbf{E}_{s_k \in \{\pm 1\}\ \forall k \in [n-1]} \Phi(s_1, \cdots, s_{n-1}, 0) + \frac{1}{2} \\
&\vdots \\
&\ge \Phi(0) + \frac{n}{2}.
\end{aligned}$$

□

Remark. This inequality is tight because $\frac{1}{2}\|x\|^2$ is 1-strongly convex in $\|\cdot\|_\infty$ and its value is between 0 and $\frac{n}{2}$.

Algorithm

Now, we give another geometry dependent algorithm that relies on a different set of assumptions. The problem we study in this section is of the form

$$\min_{x \in \mathcal{D}} f(x)$$

for f such that ∇f is "Lipschitz" in a certain sense. The algorithm reduces the problem to a sequence of linear optimization problems.

Algorithm 12: `FrankWolfe`

Input: Initial point $x^{(0)} \in \mathbb{R}^n$, step size $h > 0$.
for $k = 0, 1, \cdots, T-1$ **do**
  Compute $y^{(k)} = \arg\min_{y \in \mathcal{D}} \left\langle y, \nabla f(x^{(k)}) \right\rangle$.
  $x^{(k+1)} \leftarrow (1 - h_k) x^{(k)} + h_k y^{(k)}$ with $h_k = \frac{2}{k+2}$.
end
return $x^{(T)}$.

Analysis

Theorem 5.2.2. *Let f be a convex function on a convex set $\mathcal{D}$ with a constant C_f such that*

$$f((1-h)x + hy)) \le f(x) + h \langle \nabla f(x), y - x \rangle + \frac{1}{2} C_f h^2.$$

Then, for any $x, y \in \mathcal{D}$ and $h \in [0, 1]$, we have

$$f(x^{(k)}) - f(x^*) \le \frac{2C_f}{k+2}.$$

Remark. If ∇f is L-Lipschitz with respect to the norm $\|\cdot\|$ over the domain $\mathcal{D}$, then $C_f \le L \cdot \operatorname{diam}_{\|\cdot\|}(\mathcal{D})^2$.

Proof. By the definition of C_f, we have that

$$f(x^{(k+1)}) \le f(x^{(k)}) + h_k \left\langle \nabla f(x^{(k)}), y^{(k)} - x^{(k)} \right\rangle + \frac{1}{2} C_f h_k^2.$$

Note that

$$f(x^*) \ge f(x^{(k)}) + \left\langle \nabla f(x^{(k)}), x^* - x^{(k)} \right\rangle \ge f(x^{(k)}) + \left\langle \nabla f(x^{(k)}), y^{(k)} - x^{(k)} \right\rangle$$

where we used the fact that $y^{(k)} = \arg\min_{y \in \mathcal{D}} \left\langle y, \nabla f(x^{(k)}) \right\rangle$ and that $x^* \in \mathcal{D}$. Hence, we have that

$$f(x^{(k+1)}) \le f(x^{(k)}) - h_k (f(x^{(k)}) - f(x^*)) + \frac{1}{2} C_f h_k^2.$$

Let $e_k = f(x^{(k)}) - f(x^*)$. Then,

$$e_{k+1} \le (1 - h_k) e_k + \frac{1}{2} C_f h_k^2.$$

Note that $e_0 = f(x^{(0)}) - f^* \le \frac{1}{2} C_f$. By induction, we have that $e_k \le \frac{2C_f}{k+2}$. □

Remark. Note that this proof is in fact the same as Theorem 2.3.2.

5.3 The Newton Method

We begin with the Newton-Raphson method for finding the zeros of a function g, i.e. finding x such that $g(x) = 0$. In optimization, the goal is to find a root of the gradient, $\nabla f(x) = 0$. The Newton-Raphson iteration approximates a function by its gradient/Jacobian, then guesses where the gradient intersects the zero line as a likely zero, repeating this process until a sufficient approximation has been found. In one dimension, we approximate the function g by its gradient

$$g(x) \sim g(x^{(k)}) + g'(x^{(k)})(x - x^{(k)}).$$

Finding the zeros of the right hand side, we have the Newton step

$$x^{(k+1)} = x^{(k)} - \frac{g(x^{(k)})}{g'(x^{(k)})}.$$

In high dimension, we can approximate the function by its Jacobian $g(x) \sim g(x^{(k)}) + Dg(x^{(k)})(x - x^{(k)})$ and this gives the step

$$x^{(k+1)} = x^{(k)} - \left(Dg(x^{(k)}) \right)^{-1} g(x^{(k)}).$$

When the function $g(x) = \nabla f(x)$, then the Newton step becomes

$$x^{(k+1)} = x^{(k)} - \left(\nabla^2 f(x^{(k)}) \right)^{-1} \nabla f(x^{(k)}).$$

Alternatively, we can derive the Newton step by

$$x^{(k+1)} \leftarrow \min_x f(x^{(k)}) + \left\langle \nabla f(x^{(k)}), x - x^{(k)} \right\rangle + \frac{1}{2} (y - x^{(k)})^\mathsf{T} (\nabla^2 f(x^{(k)}))(y - x^{(k)}),$$

namely the Newton step finds the minimum of the second order approximation of the function. We note that the Newton iteration is a natural idea, since

1. It is affine invariant (changing coordinate systems won't affect the convergence).
2. It is the fastest descent direction taking the Hessian into account.

For many classes of functions, gradient methods converge to the solution linearly (namely, it takes $c \cdot \log \frac{1}{\epsilon}$ iterations for some c depending on the problem) while the Newton method converges to the solution quadratically (namely, it takes $c' \cdot \log\log \frac{1}{\epsilon}$ for some c') if the starting point is sufficiently close to a root. However, each step of Newton method involves solving a linear system, which can be much more expensive.

Algorithm 13: `NewtonMethod`

Input: Initial point $x^{(0)} \in \mathbb{R}^n$
for $k = 0, 1, \cdots, T-1$ **do**
  $x^{(k+1)} = x^{(k)} - \left(Dg(x^{(k)})\right)^{-1} g(x^{(k)})$.
end
return $x^{(T)}$.

Theorem 5.3.1 (Quadratic convergence)**.** *Assume that $g : \mathbb{R}^n \to \mathbb{R}^n$ is twice continuously differentiable. Let $x^{(k)}$ be the sequence given by the Newton method. Suppose that $x^{(k)}$ converges to some x^* such that $g(x^*) = 0$ and $Dg(x^*)$ is invertible. Then, for k large enough, we have*

$$\|x^{(k+1)} - x^*\| \le \|(Dg(x^*))^{-1} D^2 g(x^*)\|_{\mathrm{op}} \cdot \|x^{(k)} - x^*\|^2.$$

Proof. Let $e^{(k)} = x^* - x^{(k)}$. Then, we have that

$$g(x^*) = g(x^{(k)}) + Dg(x^{(k)})[e^{(k)}] + \int_0^1 (1-s) D^2 g(x^{(k)} + s e^{(k)})[e^{(k)}, e^{(k)}] ds.$$

Hence, we have

$$0 = Dg(x^{(k)})^{-1} g(x^{(k)}) + x^* - x^{(k)} + \int_0^1 (1-s) Dg(x^{(k)})^{-1} D^2 g(x^{(k)} + s e^{(k)})[e^{(k)}, e^{(k)}] ds.$$

Since $x^{(k+1)} = x^{(k)} - Dg(x^{(k)})^{-1} g(x^{(k)})$, we have

$$x^{(k+1)} - x^* = \int_0^1 (1-s) Dg(x^{(k)})^{-1} D^2 g(x^{(k)} + s e^{(k)})[e^{(k)}, e^{(k)}] ds.$$

So, we have

$$\begin{aligned} \|x^{(k+1)} - x^*\| &\le \int_0^1 (1-s) \|Dg(x^{(k)})^{-1} D^2 g(x^{(k)} + s e^{(k)})[e^{(k)}, e^{(k)}]\| ds \\ &\le \int_0^1 (1-s) \|Dg(x^{(k)})^{-1} D^2 g(x^{(k)} + s e^{(k)})\|_{\mathrm{op}} ds \cdot \|x^{(k)} - x^*\|^2. \end{aligned}$$

Since $x^{(k)} \to x^*$, we have that

$$Dg(x^{(k)})^{-1} D^2 g(x^{(k)} + s e^{(k)}) \to Dg(x^*)^{-1} D^2 g(x^*)$$

and for large enough k, we can assume

$$\|Dg(x^{(k)})^{-1} D^2 g(x^{(k)} + s e^{(k)})\|_{\mathrm{op}} \le 2 \|Dg(x^*)^{-1} D^2 g(x^*)\|_{\mathrm{op}}.$$

The result follows. □

This argument above uses only that $g \in \mathcal{C}^2$ and does not require convexity. Without further global assumptions on g, the Newton method does not always converge to a root. The argument above only shows that if the algorithm converges, then it converges quadratically eventually. We call this "local" convergence since it only gives a bound when $x^{(k)}$ is close enough to x^*. In comparison, all earlier analyses in this book are about global convergence, bounding the total number of iterations. In practice, both analyses are important; global convergence makes sure the algorithm is robust and local quadratic convergence makes sure the algorithm converges to machine accuracy quickly enough. The local quadratic convergence is particularly important for simple problems such as computing $\sqrt{x}$.

Finding the largest root of a real-rooted polynomial

In general, the Newton method does not converge globally. Here, we give an application that does converge "globally".

Theorem 5.3.2. *Given a polynomial $g(x) = \sum_{i=1}^{n} a_i x^i$ with only real roots. Let λ_1 be its largest root. Suppose that $\lambda_1 \le x^{(0)}$. Then, after $O(n \log(\frac{1}{\epsilon}))$ iterations, the Newton method finds $x^{(k)}$ such that*

$$\lambda_1 \le x^{(k)} \le \lambda_1 + \varepsilon \cdot (x^{(0)} - \lambda_1).$$

Proof. Since the roots of g are real, we can write

$$g(x) = a_n \cdot \prod_{i=1}^{n} (x - \lambda_i)$$

with $\lambda_n \le \lambda_{n-1} \le \cdots \le \lambda_1$. Then, we have that $g'(x) = a_n \cdot \sum_i \prod_{j \ne i} (x - \lambda_i)$ and hence

$$\frac{g(x)}{g'(x)} = \frac{1}{\sum_i \frac{1}{x - \lambda_i}}.$$

For any $x \ge \lambda_1$, we have

$$\frac{g(x)}{g'(x)} \le \frac{1}{\frac{1}{x-\lambda_1}} = x - \lambda_1,$$
$$\frac{g(x)}{g'(x)} \ge \frac{1}{\sum_i \frac{1}{x-\lambda_1}} = \frac{x - \lambda_1}{n}.$$

Hence, by induction, we have that $x^{(k)} \ge \lambda_1$ every step and that

$$x^{(k+1)} - \lambda_1 \le (1 - \frac{1}{n})(x^{(k)} - \lambda_1).$$

□

Exercise 5.3.3. Consider the following iteration:

$$x^{t+1} = x^t - \frac{1}{n^{1/k}} \frac{g^{(k-1)}(x)}{g^{(k)}(x)}$$

where $g^{(k)}(x) = \sum_i \frac{1}{(x-\lambda_i)^k}$. For $k = 1$, this is exactly the Newton iteration as $g^{(0)}(x) = n$. Show that when applied to a degree n real-rooted polynomial, starting with $x^{(0)} > \lambda_1$, in each iteration the distance to the largest root decreases by a factor of $1 - \frac{1}{n^{1/k}}$.

Such a dependence of $\log \frac{1}{\epsilon}$ is called linear convergence. The convergence of the Newton method can be quadratic, when close enough to a root.

Theorem 5.3.4. *Assume that $|f'(x^*)| \ge \alpha$ at $f(x^*) = 0$ and f' is L-Lipschitz. Then, if $|x^0 - x^*| \le \frac{\alpha}{2L}$,*

$$|x^t - x^*| \le \frac{\alpha}{L} \left(\frac{L}{\alpha} |x^t - x^*| \right)^{2^t}.$$

Proof. The iteration says

$$x^{t+1} - x^* = x^t - x^* - \frac{f(x^t)}{f'(x^t)}.$$

$$f(x^*) = f(x^t) + \int_{x^t}^{x^*} f'(z) dz = f(x^t) + f'(x^t)(x^* - x^t) + \int_{x^t}^{x^*} (f'(z) - f'(x^t)) dz.$$

Therefore,

$$\begin{aligned} x^{t+1} - x^* &= \frac{1}{f'(x^t)} \int_{x^t}^{x^*} (f'(z) - f'(x^t))dz. \\ &\leq \frac{L}{|f'(x^t)|} \int_{x^t}^{x^*} |z - x^t| dz \\ &= \frac{L|x^* - x^t|^2}{2|f'(x^t)|}. \end{aligned}$$

Since $f'(x^t) \geq f'(x^*) - L|x^t - x^*|$,

$$\left|x^{t+1} - x^*\right| \leq \frac{L}{2(\alpha - L|x^t - x^*|)} |x^t - x^*|^2.$$

So, if $|x^t - x^*| \leq \frac{\alpha}{2L}$,

$$|x^{t+1} - x^*| \leq \frac{L}{\alpha}|x^t - x^*|^2 \leq \frac{1}{2}|x^t - x^*|$$

and

$$\frac{L}{\alpha}|x^{t+1} - x^*| \leq \left(\frac{L}{\alpha}|x^t - x^*|\right)^2.$$

After t steps,

$$\frac{L}{\alpha}|x^t - x^*| \leq \left(\frac{L}{\alpha}\epsilon_0\right)^{2^t}$$

implying $|x^t - x^*| < \epsilon$ after $\log\log(\frac{L\epsilon_0}{\alpha})$ steps. Note that f has not been assumed to be convex or polynomial. □

Moving back to optimization, we can view the goal as finding a root of $\nabla f(x) = 0$. Newton's iteration is the update

$$x^{t+1} = x^t - (\nabla^2 f(x^t))^{-1} \nabla f(x^t).$$

By the above proof, Newton's iteration has quadratic convergence from points close enough to the optimal.

■ 5.4 Interior Point Method for Linear Programs

In this section, we study interior point methods. In practice, this method reduces the problem of optimizing a convex function to solving a small number (often less than 30) linear systems. In theory, it reduces the problem to $\tilde{O}(\sqrt{n})$ linear systems.

We start by describing interior point method for linear programs. We establish a polynomial bound and then later discuss implementation details.

Basic Properties

We first consider the primal problem

$$(\mathrm{P}): \qquad \min_x c^\top x \text{ subject to } Ax = b, x \geq 0$$

where $A \in \mathbb{R}^{m \times n}$. The difficulty of linear programs is the constraint $x \geq 0$. Without this constraint, we can simply solve it as a linear system. One natural idea to solve linear programs is to replace the hard constraint $x \geq 0$ by some smooth function. So, let us consider the following "regularized" version of the linear program

$$(\mathrm{P}_t): \qquad \min_x c^\top x - t\sum_{i=1}^n \ln x_i \text{ subject to } Ax = b.$$

We will explain the reason of choosing $\ln x$ in more detail later. For now, we can think it as a nice function that blows up at $x = 0$.

One can think that $-\ln x$ gives a force from every constraint $x \geq 0$ to make sure $x \geq 0$ is true. Since the gradient of $-\ln x$ blows up when $x = 0$, when x is close enough, the force is large enough to counter the cost c. When $t \to 0$, then the problem (P_t) is closer to the original problem (P) and hence the minimizer of (P_t) is closer to a minimizer of (P).

First, we give a formula for the minimizer of (P_t).

Lemma 5.4.1 (Existence and Uniqueness of central path)**.** *If the polytope $\{Ax = b, x \geq 0\}$ has an interior, then the optimum of (P_t) is uniquely given by*

$$\begin{aligned} xs &= t, \\ Ax &= b, \\ A^\top y + s &= c, \\ (x, s) &\geq 0 \end{aligned}$$

where $xs = t$ is a shorthand of $x_i s_i = t$ for all i.

Proof. The optimality condition, using dual variables y for the Lagrangian of $Ax = b$ is given by

$$c - \frac{t}{x} = A^\top y.$$

Write $s_i = \frac{t}{x_i}$, to get the formula. The solution is unique because the function $-\ln x$ is strictly convex. □

Definition 5.4.2. We define the central path $\mathcal{C}_t = (x^{(t)}, y^{(t)}, s^{(t)})$ as the sequence of points satisfying

$$\begin{aligned} x^{(t)} s^{(t)} &= t, \\ Ax^{(t)} &= b, \\ A^\top y^{(t)} + s^{(t)} &= c, \\ (x^{(t)}, s^{(t)}) &\geq 0. \end{aligned}$$

To give another interpretation of the central path, note that the dual problem is

$$(\mathrm{D}): \qquad \max_{y,s} b^\top y \text{ subject to } A^\top y + s = c, s \geq 0.$$

Note that for any feasible x and y, we have that

$$0 \leq c^\top x - b^\top y = c^\top x - x^\top A^\top y = x^\top s.$$

Hence, (x, y, s) solves the linear program if it satisfies the central path equation with $t = 0$. Therefore, central path is a balanced way to decrease $x_i s_i$ uniformly to 0. We can formalize the intuition that for small t, $x^{(t)}$ is a good approximation of the primal solution.

Lemma 5.4.3 (Duality Gap)**.** *We have that*

$$\textit{Duality Gap} = c^\top x^{(t)} - b^\top y^{(t)} = c^\top x^{(t)} - \left(x^{(t)}\right)^\top A^\top y^{(t)} = \left(x^{(t)}\right)^\top s^{(t)} = tn.$$

The interior point method follows the following framework:

1. Find $\mathcal{C}_1$
2. Until $t < \frac{\varepsilon}{n}$,
   (a) Use $\mathcal{C}_t$ to find $\mathcal{C}_{(1-h)t}$ for $h = \frac{1}{10\sqrt{n}}$.

Note that this algorithm only finds a solution with ε error. If the linear program is integral, we can simply stop at small enough ε and round it off to the closest integral point.

Finding the initial point

The first question is to find $\mathcal{C}_1$. This can be handled by extending the problem to slightly higher dimension. To the reader familiar with the Simplex method, this might be reminiscent of the two phases of the simplex method, where the purpose of the first phase is to find a feasible initial solution.

Lemma 5.4.4. *Consider a linear program* $\min_{Ax=b,x\geq 0} c^\top x$ *with* n *variables and* d *constraints. Assume that*

1. *Diameter: For any* $x \geq 0$ *with* $Ax = b$*, we have that* $\|x\|_\infty \leq R$*.*
2. *Lipschitz constant of the objective:* $\|c\|_\infty \leq L$*.*

For any $0 < \delta \leq 1$*, the modified linear program* $\min_{\overline{A}\overline{x}=\overline{b},\overline{x}\geq 0} \overline{c}^\top \overline{x}$ *with*

$$\overline{A} = \begin{bmatrix} A & 0 & \frac{1}{R}b - A1_n \\ 1_n^\top & 1 & 0 \end{bmatrix}, \overline{b} = \begin{bmatrix} \frac{1}{R}b \\ n+1 \end{bmatrix}, \text{ and } \overline{c} = \begin{bmatrix} \delta/L \cdot c \\ 0 \\ 1 \end{bmatrix}$$

satisfies the following:

1. $\overline{x} = \begin{bmatrix} 1_n \\ 1 \\ 1 \end{bmatrix}$, $\overline{y} = \begin{bmatrix} 0_d \\ -1 \end{bmatrix}$ *and* $\overline{s} = \begin{bmatrix} 1_n + \frac{\delta}{L} \cdot c \\ 1 \\ 1 \end{bmatrix}$ *are feasible primal dual vectors.*
2. *For any feasible primal dual vectors* $(\overline{x}, \overline{y}, \overline{s})$ *with duality gap at most* δ^2*, the vector* $\hat{x} = R \cdot \overline{x}_{1:n}$ *(*$\overline{x}_{1:n}$ *are the first* n *coordinates of* $\overline{x}$*) is an approximate solution to the original linear program in the following sense*

$$\begin{aligned} c^\top \hat{x} &\leq \min_{Ax=b,x\geq 0} c^\top x + LR \cdot \delta, \\ \|A\hat{x} - b\|_1 &\leq 4n\delta \cdot \left(R \sum_{i,j} |A_{i,j}| + \|b\|_1 \right), \\ \hat{x} &\geq 0. \end{aligned}$$

Part 1. For the first result, straightforward calculations show that $(\overline{x}, \overline{y}, \overline{s}) \in \mathbb{R}^{(n+2)\times(d+1)\times(n+2)}$ are feasible, i.e.,

$$\overline{A}\overline{x} = \begin{bmatrix} A & 0 & \frac{1}{R}b - A1_n \\ 1_n^\top & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1_n \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{R}b \\ n+1 \end{bmatrix} = \overline{b}$$

and

$$\begin{aligned} \overline{A}^\top \overline{y} + \overline{s} &= \begin{bmatrix} A^\top & 1_n \\ 0 & 1 \\ \frac{1}{R}b^\top - 1_n^\top A^\top & 0 \end{bmatrix} \cdot \begin{bmatrix} 0_d \\ -1 \end{bmatrix} + \begin{bmatrix} 1_n + \frac{\delta}{L} \cdot c \\ 1 \\ 1 \end{bmatrix} \\ &= \begin{bmatrix} -1_n \\ -1 \\ 0 \end{bmatrix} + \begin{bmatrix} 1_n + \frac{\delta}{L} \cdot c \\ 1 \\ 1 \end{bmatrix} \\ &= \begin{bmatrix} \frac{\delta}{L} \cdot c \\ 0 \\ 1 \end{bmatrix} = \overline{c} \end{aligned}$$

Part 2. For the second result, we let

$$\mathrm{OPT} = \min_{Ax=b,x\geq 0} c^\top x, \quad \text{and,} \quad \overline{\mathrm{OPT}} = \min_{\overline{A}\overline{x}=\overline{b},\overline{x}\geq 0} \overline{c}^\top \overline{x}$$

For any optimal $x \in \mathbb{R}^n$ in the original LP, we consider the following $\overline{x} \in \mathbb{R}^{n+2}$

$$\overline{x} = \begin{bmatrix} \frac{1}{R}x \\ n + 1 - \frac{1}{R}\sum_{i=1}^n x_i \\ 0 \end{bmatrix} \tag{5.4.1}$$

and $\overline{c} \in \mathbb{R}^{n+2}$

$$\overline{c} = \begin{bmatrix} \frac{\delta}{L} \cdot c^\top \\ 0 \\ 1 \end{bmatrix} \tag{5.4.2}$$

We want to argue that $\overline{x} \in \mathbb{R}^{n+2}$ is feasible in the modified LP. It is obvious that $\overline{x} \geq 0$, it remains to show $\overline{A}\overline{x} = \overline{b} \in \mathbb{R}^{d+1}$. We have

$$\overline{A}\overline{x} = \begin{bmatrix} A & 0 & \frac{1}{R}b - A1_n \\ 1_n^\top & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} \frac{1}{R}x \\ n+1-\frac{1}{R}\sum_{i=1}^n x_i \\ 0 \end{bmatrix} = \begin{bmatrix} \frac{1}{R}Ax \\ n+1 \end{bmatrix} = \begin{bmatrix} \frac{1}{R}b \\ n+1 \end{bmatrix} = \overline{b},$$

where the third step follows from $Ax = b$, and the last step follows from definition of $\overline{b}$.

Therefore, using the definition of $\overline{x}$ in (5.4.1) we have that

$$\overline{\mathrm{OPT}} \leq \overline{c}^\top \overline{x} = \begin{bmatrix} \frac{\delta}{L} \cdot c^\top & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \frac{1}{R}x \\ n+1-\frac{1}{R}\sum_{i=1}^n x_i \\ 0 \end{bmatrix} = \frac{\delta}{LR} \cdot c^\top x = \frac{\delta}{LR} \cdot \mathrm{OPT}. \tag{5.4.3}$$

where the first step follows from modified program is solving a minimization problem, the second step follows from definition of $\overline{x} \in \mathbb{R}^{n+2}$ (5.4.1) and $\overline{c} \in \mathbb{R}^{n+2}$ (5.4.2), the last step follows from $x \in \mathbb{R}^n$ is an optimal solution in the original linear program.

Given a feasible $(\overline{x}, \overline{y}, \overline{s}) \in \mathbb{R}^{(n+2)\times(d+1)\times(n+2)}$ with duality gap δ^2, we can write $\overline{x} = \begin{bmatrix} \overline{x}_{1:n} \\ \tau \\ \theta \end{bmatrix} \in \mathbb{R}^{n+2}$ for some $\tau \geq 0$, $\theta \geq 0$. We can compute $\overline{c}^\top \overline{x}$ which is $\frac{\delta}{L} \cdot c^\top \overline{x}_{1:n} + \theta$. Then, we have

$$\frac{\delta}{L} \cdot c^\top \overline{x}_{1:n} + \theta \leq \overline{\mathrm{OPT}} + \delta^2 \leq \frac{\delta}{LR} \cdot \overline{\mathrm{OPT}} + \delta^2, \tag{5.4.4}$$

where the first step follows from definition of duality gap, the last step follows from (5.4.3).

Hence, we can upper bound the $\overline{\mathrm{OPT}}$ of the transformed program as follows:

$$c^\top \hat{x} = R \cdot c^\top \overline{x}_{1:n} = \frac{LR}{\delta} \cdot \frac{\delta}{L} c^\top \overline{x}_{1:n} \leq \frac{RL}{\delta}(\frac{\delta}{LR} \cdot \mathrm{OPT} + \delta^2) = \mathrm{OPT} + LR \cdot \delta,$$

where the first step follows by $\hat{x} = R \cdot \overline{x}_{1:n}$, the third step follows by (5.4.4).

Note that

$$\frac{\delta}{L} c^\top \overline{x}_{1:n} \geq -\frac{\delta}{L}\|c\|_\infty \|\overline{x}_{1:n}\|_1 = -\frac{\delta}{L}\|c\|_\infty \|\frac{1}{R}x\|_1 \geq -\frac{\delta}{L}\|c\|_\infty \frac{n}{R}\|x\|_\infty \geq -\delta n, \tag{5.4.5}$$

where the second step follows from definition $\overline{x} \in \mathbb{R}^{n+2}$, and the last step follows from $\|c\|_\infty \leq L$ and $\|x\|_\infty \leq R$.

We can upper bound the θ in the following sense,

$$\theta \leq \frac{\delta}{LR} \cdot \overline{\mathrm{OPT}} + \delta^2 + \delta n \leq 2n\delta + \delta^2 \leq 4n\delta \tag{5.4.6}$$

where the first step follows from (5.4.4) and (5.4.5), the second step follows by $\mathrm{OPT} = \min_{Ax=b, x\geq 0} c^\top x \leq nLR$ (because $\|c\|_\infty \leq L$ and $\|x\|_\infty \leq R$), and the last step follows from $\delta \leq 1 \leq n$.

The constraint in the new polytope shows that

$$A\overline{x}_{1:n} + (\frac{1}{R}b - A1_n)\theta = \frac{1}{R}b.$$

Using $\hat{x} = Rx_{1:n} \in \mathbb{R}^n$, we have

$$A\frac{1}{R}\hat{x} + (\frac{1}{R}b - A1_n)\theta = \frac{1}{R}b.$$

Rewriting it, we have $A\hat{x} - b = (RA1_n - b)\theta \in \mathbb{R}^d$ and hence

$$\|A\hat{x} - b\|_1 = \|(RA1_n - b)\theta\|_1 \leq \theta(\|RA1_n\|_1 + \|b\|_1) \leq \theta \cdot (R\|A\|_1 + \|b\|_1) \leq 4n\delta \cdot (R\|A\|_1 + \|b\|_1),$$

where the second step follows from triangle inequality, the third step follows from $\|A1_n\|_1 \le \|A\|_1$ (because the definition of entry-wise ℓ_1 norm), and the last step follows from (5.4.6).

Thus, we complete the proof.

Following the central path

During the algorithm, we maintain a point (x, y, s) such that $Ax = b$, $A^\top y + s = c$ and $x_i s_i$ is close to t for all i. We show how to find a feasible $(x + \delta_x, y + \delta_y, s + \delta_s)$ such that it is even closer to t. We can write the equation as follows:

$$\begin{aligned}(x+\delta_x)(s+\delta_s) &\approx t,\\ A(x+\delta_x) &= b,\\ A^\top(y+\delta_y)+(s+\delta_s) &= c.\end{aligned}$$

(Omitted the non-negative conditions.) Using our assumption on (x, y, s) and noting that $\delta_x \cdot \delta_s$ is small, the equation can simplified as follows. We use the notation $X = \mathrm{Diag}(x)$, $S = \mathrm{Diag}(s)$.

$$\begin{bmatrix} 0 & A^\top & I \\ A & 0 & 0 \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \delta_x \\ \delta_y \\ \delta_s \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ t - xs \end{bmatrix}.$$

This is a linear system and hence we can solve it exactly.

Exercise 5.4.5. Let $r = t - xs$. Prove that $S\delta_x = (I - \overline{P})r$ and $X\delta_s = \overline{P}r$ where $\overline{P} = XA^\top(AS^{-1}XA^\top)^{-1}AS^{-1}$.

First, we show that $x^{(\mathrm{new})} = x + \delta_x$ and $s^{(\mathrm{new})} = s + \delta_s$ are feasible.

Lemma 5.4.6. *Suppose $\sum_i (x_i s_i - t)^2 \le \varepsilon^2 t^2$ with $\varepsilon < \frac{1}{2}$. Then,$x_i^{(\mathrm{new})} > 0$ and $s_i^{(\mathrm{new})} > 0$ for all i.*

Proof. Consider the orthogonal projection matrix $P = S^{-\frac{1}{2}}X^{\frac{1}{2}}A^\top(AS^{-1}XA^\top)^{-1}AS^{-\frac{1}{2}}X^{\frac{1}{2}}$. Note that

$$X^{-1}\delta_x = S^{-\frac{1}{2}}X^{-\frac{1}{2}}(I - P)S^{-\frac{1}{2}}X^{-\frac{1}{2}}r.$$

By the assumption for each i, $x_i s_i \ge (1 - \varepsilon)t$. Therefore, we have

$$\begin{aligned}\|X^{-1}\delta_x\|_2 &\le \frac{1}{\sqrt{(1-\epsilon)t}}\|(I-P)S^{-\frac{1}{2}}X^{-\frac{1}{2}}r\|_2\\ &\le \frac{1}{\sqrt{(1-\epsilon)t}}\|S^{-\frac{1}{2}}X^{-\frac{1}{2}}r\|_2\\ &\le \frac{1}{(1-\epsilon)t}\|r\|_2 \le \frac{\epsilon}{1-\epsilon}.\end{aligned}$$

Similarly, we have $S^{-1}\delta_s = S^{-\frac{1}{2}}X^{-\frac{1}{2}}PS^{-\frac{1}{2}}X^{-\frac{1}{2}}r$. Hence, we have $\|S^{-1}\delta_s\|_2 \le \frac{\epsilon}{1-\epsilon}$. Therefore, when $\epsilon < \frac{1}{2}$, we have both $\|X^{-1}\delta_x\|_\infty$ and $\|S^{-1}\delta_s\|_\infty$ less than 1, which shows that both $x^{(\mathrm{new})}$ and $s^{(\mathrm{new})}$ are positive.

Next, we show that xs is closer to t after one Newton step. □

Lemma 5.4.7. *If $\sum_i (x_i s_i - t)^2 \le \varepsilon^2 t^2$ with $\epsilon < \frac{1}{4}$, we have that*

$$\sum_i \left(x_i^{(\mathrm{new})} s_i^{(\mathrm{new})} - t\right)^2 \le \left(\epsilon^4 + 16\epsilon^5\right) t^2.$$

Proof. We have that $x_i\delta_{s,i} + s_i\delta_{x,i} = t - x_i s_i$. Using this,

$$\mathrm{LHS} = \sum_i \left(x_i^{(\mathrm{new})} s_i^{(\mathrm{new})} - t\right)^2 = \sum_i (x_i s_i + x_i\delta_{s,i} + s_i\delta_{x,i} + \delta_{x,i}\delta_{s,i} - t)^2 = \sum_i \delta_{x,i}^2\delta_{s,i}^2 \le ((1+\epsilon)t)^2 \cdot \sum_i \left(\frac{\delta_{x,i}}{x_i}\right)^2 \left(\frac{\delta_{s,i}}{s_i}\right)^2$$

where in the last step we used $x_i^2 s_i^2 \le (1+\varepsilon)^2 t^2$. Using the previous lemma, we have that

$$\begin{aligned}\text{LHS} &\le ((1+\epsilon)t)^2 \cdot \|X^{-1}\delta_x\|_4^2 \|S^{-1}\delta_s\|_4^2 \\ &\le ((1+\epsilon)t)^2 \cdot \|X^{-1}\delta_x\|_2^2 \|S^{-1}\delta_s\|_2^2 \\ &\le ((1+\epsilon)t)^2 \left(\frac{\epsilon}{1-\epsilon}\right)^4 \\ &\le \left(\epsilon^4 + 16\epsilon^5\right) t^2\end{aligned}$$

□

Using this, we have the main theorem.

Theorem 5.4.8. *We can solve a linear program to within δ "error" (see Lemma 5.4.4) in $O(\sqrt{n}\log(\frac{1}{\delta}))$ iterations and each iteration only needs to solve a linear system.*

Proof. Let $\Phi = \sum_i (x_i s_i - t)^2$ be the error of the current iteration. We always maintain $\Phi \le \frac{t^2}{16}$ for the current (x, y, s) and t. At each step, we use Lemma 5.4.7 which makes $\Phi \le \frac{t^2}{50}$. Then, we decrease t by $t(1 - \frac{1}{10\sqrt{n}})$. Note that

$$\sum_i (x_i s_i - t(1-h))^2 \le 2\Phi + 2t^2 h^2 n \le \frac{2t^2}{50} + \frac{2t^2}{100} \le \frac{(t(1-h))^2}{16}.$$

Therefore, the invariant is preserved after each step. Since t is decreased by a $(1 - \frac{1}{10\sqrt{n}})$ factor each step, it takes $O(\sqrt{n}\log(\frac{1}{\delta}))$ to decrease t from 1 to δ^2. □

Why $\sqrt{n}$?

The central path is the solution to the following ODE

$$\begin{aligned} S_t \frac{d}{dt} x_t + X_t \frac{d}{dt} s_t &= 1, \\ A \frac{d}{dt} x_t &= 0, \\ A^\top \frac{d}{dt} y_t + \frac{d}{dt} s_t &= 0. \end{aligned}$$

Solving this linear system, we have that $S_t \frac{dx_t}{dt} = (I - P_t)1$ and $X_t \frac{ds_t}{dt} = P_t 1$ where $P_t = X_t A^\top (A S_t^{-1} X_t A^\top)^{-1} A S_t^{-1}$. Using that $x_t s_t = t$, we have that

$$P_t = X_t A^\top (A X_t^2 A^T)^{-1} A X_t = S_t^{-1} A^\top (A S_t^{-2} A)^{-1} A S_t^{-1}$$

and that $X_t^{-1} \frac{dx_t}{dt} = \frac{1}{t}(I - P_t)1$ and $S_t^{-1} \frac{ds_t}{dt} = \frac{1}{t} P_t 1$. Equivalently, we have

$$\frac{d \ln x_t}{d \ln t} = (I - P_t)1 \text{ and } \frac{d \ln s_t}{d \ln t} = P_t 1.$$

Note that

$$\|P_t 1\|_\infty \le \|P_t 1\|_2 = \sqrt{n}.$$

Hence, x_t and s_t can change by at most a constant factor when we change t by a $1 \pm \frac{1}{\sqrt{n}}$ factor.

Exercise 5.4.9. If we are given x such that $\|\ln x - \ln x_t\|_\infty = O(1)$, then we can find x_t by solving $\tilde{O}(1)$ linear systems.

5.5 Newton Method and Self-concordance

In this section, we give a general analysis for the Newton method. In the next section, we will use this to show that interior point method can be generalized to convex optimization. A key property of the Newton method is that

it is invariant under linear transformation. In general, whenever a method uses k^{th} order information, we need to assume the k^{th} derivative is continuous. Otherwise, the k^{th} derivative is not useful for algorithmic purposes. For the Newton method, it is convenient to assume that the Hessian is Lipschitz. Since the method is invariant under linear transformation, it only makes sense to impose an assumption that is invariant under linear transformation.

Definition 5.5.1. Given a convex function $f : \mathbb{R}^n \to \mathbb{R}$, and any point $x \in \mathbb{R}^n$, define the norm $\|.\|_x$ as

$$\|v\|_x^2 = v^\top \nabla^2 f(x) v.$$

We call a function f self-concordant if for any $h \in \mathbb{R}^n$ and any x in $\mathrm{dom} f$, we have

$$D^3 f(x)[h,h,h] \le 2\|h\|_x^3$$

where $D^k f(x)[h_1, h_2, \cdots, h_k]$ is the directional k^{th} derivative of f along the directions $h_1, h_2, \cdots, h_k$.

Remark. The constant 2 is chosen so that $-\ln(x)$ exactly satisfies the assumption and it is not very important, in that by scaling f, we can change any constant to any other constant.

Exercise 5.5.2. Show that the following property is equivalent fo self-concordance as defined above: restricted on any straight line $g(t) = f(x + th)$, we have $g'''(t) \le g''(t)^{3/2}$.

Exercise 5.5.3. Show that the functions $x^\top Ax$, $-\ln x$, $-\ln(1-\sum x_i^2)$, $-\ln\det X$ are self-concordant under suitable nonnegativity conditions.

The self-concordance condition says that locally, the Hessian does not change too fast, i.e., the change in the Hessian is bounded by its magnitude (to the power 1.5). We will skip the proof of the lemma below.

Lemma 5.5.4. *Given a self-concordant function f, for any $h_1, h_2, h_3 \in \mathbb{R}^n$, we have*

$$D^3 f(x)[h_1,h_2,h_3] \le 2\|h_1\|_x \|h_2\|_x \|h_3\|_x .$$

From the self-concordance condition, we have the following more directly usable property.

Lemma 5.5.5. *For a self-concordant function f and any $x \in \mathrm{dom} f$ and any $\|y-x\|_x < 1$, we have that*

$$(1-\|y-x\|_x)^2 \nabla^2 f(x) \preceq \nabla^2 f(y) \preceq \frac{1}{(1-\|y-x\|_x)^2}\nabla^2 f(x).$$

Proof. Let $\alpha(t) = \left\langle \nabla^2 f(x+t(y-x))u, u\right\rangle$. Then, we have that

$$\alpha'(t) = D^3 f(x+t(y-x))[y-x,u,u].$$

By self-concordance, we have

$$|\alpha'(t)| \le 2\|y-x\|_{x+t(y-x)} \|u\|_{x+t(y-x)}^2 . \tag{5.5.1}$$

For $u = y - x$, we have $|\alpha'(t)| \le 2\alpha(t)^{\frac{3}{2}}$. Hence, we have $\frac{d}{dt}\frac{1}{\sqrt{\alpha(t)}} \ge -1$. Integrating both on t, we have

$$\frac{1}{\sqrt{\alpha(t)}} \ge \frac{1}{\sqrt{\alpha(0)}} - t = \frac{1}{\|x-y\|_x} - t.$$

Rearranging it gives

$$\|y-x\|_{x+t(y-x)}^2 = \alpha(t) \le \frac{1}{(\frac{1}{\|x-y\|_x} - t)^2} = \frac{\|x-y\|_x^2}{(1-t\|x-y\|_x)^2}.$$

For general u, (5.5.1) gives

$$|\alpha'(t)| \le 2\frac{\|x-y\|_x}{1-t\|x-y\|_x}\alpha(t).$$

Rearranging,

$$\left|\frac{d}{dt}\ln\alpha(t)\right| \le 2\frac{\|x-y\|_x}{1-t\|x-y\|_x} = -2\frac{d}{dt}\ln(1-t\|x-y\|_x)$$

Integrating both from $t = 0$ to 1 gives the result. □

Now we are ready to study the convergence of Newton method:

Lemma 5.5.6. *Given a self-concordant convex function f, consider the iteration*

$$x' = x - (\nabla^2 f(x))^{-1}\nabla f(x).$$

Suppose that $r = \|\nabla f(x)\|_{\nabla^2 f(x)^{-1}} < 1$, then we have

$$\|\nabla f(x')\|_{\nabla^2 f(x')^{-1}} \le \frac{r^2}{1-r}.$$

Remark 5.5.7. Note that $\|\nabla f(x)\|_{\nabla^2 f(x)^{-1}} = \left\|\nabla^2 f(x)^{-1}\nabla f(x)\right\|_x$ is the step size of the Newton method. This is a measurement of the error, since the goal is to find x with $\nabla f(x) = 0$.

Proof. Lemma 5.5.5 shows that

$$\nabla^2 f(x') \succeq (1-r)^2 \nabla^2 f(x).$$

and hence

$$\|\nabla f(x')\|_{\nabla^2 f(x')^{-1}} \le \frac{\|\nabla f(x')\|_{\nabla^2 f(x)^{-1}}}{1-r}.$$

To bound $\nabla f(x')$, we calculate that

$$\begin{aligned}
\nabla f(x') &= \nabla f(x) + \int_0^1 \nabla^2 f(x + t(x'-x))(x'-x)dt \\
&= \nabla f(x) - \int_0^1 \nabla^2 f(x + t(x'-x))(\nabla^2 f(x))^{-1}\nabla f(x)dt \\
&= \left(\nabla^2 f(x) - \int_0^1 \nabla^2 f(x + t(x'-x))dt\right)(\nabla^2 f(x))^{-1}\nabla f(x). \qquad (5.5.2)
\end{aligned}$$

For the first term in the bracket, we use Lemma 5.5.5 to get that

$$(1 - r + \frac{1}{3}r^2)\nabla^2 f(x) \preceq \int_0^1 \nabla^2 f(x + t(x'-x))dt \preceq \frac{1}{1-r}\nabla^2 f(x).$$

Therefore, we have

$$\left\|(\nabla^2 f(x))^{-\frac{1}{2}}\left(\nabla^2 f(x) - \int_0^1 \nabla^2 f(x + t(x'-x))dt\right)(\nabla^2 f(x))^{-\frac{1}{2}}\right\|_{\text{op}} \le \max(\frac{r}{1-r}, r - \frac{1}{3}r^2) = \frac{r}{1-r}.$$

Putting it into (5.5.2) gives

$$\begin{aligned}
\|\nabla f(x')\|_{\nabla^2 f(x)^{-1}} &= \left\|\nabla^2 f(x)^{-\frac{1}{2}}\nabla f(x')\right\|_2 \\
&\le \left\|(\nabla^2 f(x))^{-\frac{1}{2}}\left(\nabla^2 f(x) - \int_0^1 \nabla^2 f(x + t(x'-x))dt\right)(\nabla^2 f(x))^{-\frac{1}{2}}\right\|_{\text{op}} \left\|(\nabla^2 f(x))^{-\frac{1}{2}}\nabla f(x)\right\|_2 \\
&\le \frac{r}{1-r}\cdot r \\
&= \frac{r^2}{1-r}.
\end{aligned}$$

□

Finally, we bound the error of the current iterate in terms of $\|\nabla f(x)\|_{\nabla^2 f(x)^{-1}}$.

Lemma 5.5.8. *Given x such that $\|\nabla f(x)\|_{\nabla^2 f(x)^{-1}} \le \frac{1}{6}$, we have that*

- $\|x - x^*\|_{x^*} \le 2\|\nabla f(x)\|_{\nabla^2 f(x)^{-1}}$,
- $\|x - x^*\|_x \le 2\|\nabla f(x)\|_{\nabla^2 f(x)^{-1}}$,
- $f(x) \le f(x^*) + 2\|\nabla f(x)\|^2_{\nabla^2 f(x)^{-1}}$.

Proof. Let $r = \|x - x^*\|_x$. Suppose that $r \le \frac{1}{4}$. Note that

$$\nabla f(x) = \nabla f(x) - \nabla f(x^*) = \int_0^1 \nabla^2 f(x^* + t(x - x^*))(x - x^*)dt.$$

Using that $\nabla^2 f(x^* + t(x - x^*)) \succeq (1 - (1 - t)r)^2 \nabla^2 f(x)$ (Lemma 5.5.5), we have

$$\begin{aligned}\|\nabla f(x)\|_{\nabla^2 f(x)^{-1}} &= \int_0^1 \|\nabla^2 f(x^* + t(x - x^*))(x - x^*)\|_{\nabla^2 f(x)^{-1}} dt \\ &\ge \int_0^1 (1 - (1 - t)r)^2 \|x - x^*\|_x dt \\ &= \left(1 - r + \frac{r^2}{3}\right) r \ge \frac{3r}{4}.\end{aligned}$$

Using $\|\nabla f(x)\|_{\nabla^2 f(x)^{-1}} \le \frac{1}{6}$, we have indeed $r \le \frac{1}{4}$ (our lower bound above is a non-decreasing function) Using Lemma 5.5.5 again, we have

$$\|x - x^*\|_{x^*} \le \frac{\|x - x^*\|_x}{1 - r} \le \frac{4}{3} \frac{1}{1 - \frac{1}{4}} \|\nabla f(x)\|_{\nabla^2 f(x)^{-1}} \le 2\|\nabla f(x)\|_{\nabla^2 f(x)^{-1}}.$$

For the bound for $f(x)$, we have that

$$f(x) = f(x^*) + \langle \nabla f(x^*), x - x^* \rangle + \int_0^1 (1 - t)(x - x^*)^\top \nabla^2 f(x^* + t(x - x^*))(x - x^*)dt.$$

Using that $\nabla^2 f(x^* + t(x - x^*)) \preceq \frac{1}{(1-(1-t)r)^2} \nabla^2 f(x)$, we have

$$\begin{aligned}f(x) &\le f(x^*) + \int_0^1 \frac{1 - t}{(1 - (1 - t)r)^2} dt \cdot \|x - x^*\|_x^2 \\ &= f(x^*) + \frac{1}{r^2}\left(\frac{r}{1 - r} + \log(1 - r)\right)\|x - x^*\|_x^2 \\ &\le f(x^*) + \left(\frac{1}{2} + r\right)\|x - x^*\|_x^2\end{aligned}$$

where we used $r \le \frac{1}{4}$ at the end. □

5.6 Interior Point Method for Convex Programs

The interior point method can be used to optimize any convex function. For more in-depth treatment, please see the structural programming section in [51].

Recall that any convex optimization problem

$$\min_x f(x)$$

can be rewritten in the epigraph form as

$$\min_{\{(x,t):\ f(x) \le t\}} t.$$

Hence, it suffices to study the problem $\min_{x \in K} c^\top x$. Similar to the case of linear programs, we replace the hard constraint $x \in K$ by a soft constraint as follows:

$$\min_x \phi_t(x) \text{ where } \phi_t(x) = tc^\top x + \phi(x).$$

where $\phi(x)$ is a convex function such that $\phi(x) \to +\infty$ as $x \to \partial K$. Note that we put the parameter t in front of the cost $c^\top x$ instead of ϕ as in the last lecture, it is slightly more convenient here. We say ϕ is a *barrier* for K. To be concrete, we can always keep in mind $\phi(x) = -\sum_{i=1}^n \ln x_i$. As before, we define the central path.

Definition 5.6.1. The central path $x_t = \arg\min_x \phi_t(x)$.

The interior point method follows the following framework:

1. Find x close to x_1.
2. While t is not tiny,
   (a) Move x closer to x_t
   (b) $t \to (1+h) \cdot t$.

Self-concordant barrier function

To analyze the algorithm above, we need to assume that ϕ is well-behaved. We measure the quality of ϕ by $\|\nabla\phi\|_{\nabla^2\phi(x)^{-1}}$. One can think this as the Lipschitz constant of ϕ but measured in the local norm.

Definition 5.6.2. We call ϕ is a ν-self-concordant barrier for K if ϕ is self-concordant, $\phi(x) \to +\infty$ as $x \to \partial K$ and that $\|\nabla\phi(x)\|^2_{\nabla^2\phi(x)^{-1}} \leq \nu$ for all x.

Not all convex functions are self-concordant. However, for our purpose, it suffices to show that we can construct a self-concordant barrier for any convex set.

Theorem 5.6.3. *Any convex set has an n-self concordant barrier.*

Unfortunately, this is an existence result and the barrier function is expensive to compute.In practice, we construct self-concordant barriers out of simpler ones:

Lemma 5.6.4. *We have the following self-concordant barriers. We use ν-sc as a short form for ν-self-concordant barrier.*

- $-\ln x$ *is* 1*-sc for* $\{x \geq 0\}$.
- $-\ln\cos(x)$ *is* 1*-sc for* $\{|x| \leq \frac{\pi}{2}\}$.
- $-\ln(t^2 - \|x\|^2)$ *is* 2*-sc for* $\{t \geq \|x\|_2\}$.
- $-\ln\det X$ *is* n*-sc for* $\{X \in \mathbb{R}^{n\times n}, X \succeq 0\}$.
- $-\ln x - \ln(\ln x + t)$ *is* 2*-sc for* $\{x \geq 0, t \geq -\ln x\}$.
- $-\ln t - \ln(\ln t - x)$ *is* 2*-sc for* $\{t \geq e^x\}$.
- $-\ln x - \ln(t - x\ln x)$ *is* 2*-sc for* $\{x \geq 0, t \geq x\ln x\}$.
- $-2\ln t - \ln(t^{2/p} - x^2)$ *is* 4*-sc for* $\{t \geq |x|^p\}$ *for* $p \geq 1$.
- $-\ln x - \ln(t^p - x)$ *is* 2*-sc for* $\{t^p \geq x \geq 0\}$ *for* $0 < p \leq 1$.
- $-\ln t - \ln(x - t^{-1/p})$ *is* 2*-sc for* $\{x > 0, t \geq x^{-p}\}$ *for* $p \geq 1$.
- $-\ln x - \ln(t - x^{-p})$ *is* 2*-sc for* $\{x > 0, t \geq x^{-p}\}$ *for* $p \geq 1$.

The following lemma shows how we can combine barriers:

Lemma 5.6.5. *If ϕ_1 and ϕ_2 are ν_1 and ν_2-self concordant barriers on K_1 and K_2 with respectively, then $\phi_1 + \phi_2$ is a $\nu_1 + \nu_2$ self concordant barrier on $K_1 \cap K_2$.*

Lemma 5.6.6. *If ϕ is a ν-self concordant barrier on K, then $\phi(Ax)$ is a ν-self concordant on $\{y : Ay \in K\}$.*

Exercise 5.6.7. Using the lemmas above, prove that $-\sum_{i=1}^m \ln(a_i^\top x - b_i)$ is an m-self concordant barrier on $\{Ax \geq b\}$.

Main Algorithm and Analysis

Algorithm 14: `InteriorPointMethod`

Input: A ν-self-concordant barrier ϕ for K, the minimizer x of ϕ.
Define $f_t(x) = tc^\top x + \phi(x)$. $t = \frac{1}{6}\|c\|^{-1}_{\nabla^2\phi(x)^{-1}}$.
while $t \le \frac{\nu+1}{\epsilon}$ **do**
  $x \leftarrow x - \nabla^2 f_t(x)^{-1}\nabla f_t(x)$.
  $t \leftarrow (1+h)t$ with $h = \frac{1}{9\sqrt{\nu}}$.
end
return x.

We first explain the termination condition. Intuitively, we should think $\min f_t(x_t)$ tends to optimality as $t \to \infty$. We first need a lemma showing that the gradient of ϕ is small.

Lemma 5.6.8 (Duality Gap)**.** *Suppose that ϕ is a ν-self concordant barrier. For any $x, y \in K$, we have that*

$$\langle \nabla\phi(x), y - x\rangle \le \nu.$$

Proof. Let $\alpha(t) = \langle \nabla\phi(z_t), y - x\rangle$ where $z_t = x + t(y-x)$. Then, we have

$$\alpha'(t) = \left\langle \nabla^2\phi(z_t)(y-x), y-x\right\rangle.$$

Note that

$$\alpha(t) \le \|\nabla\phi(z_t)\|_{\nabla^2\phi(z_t)^{-1}} \|y-x\|_{\nabla^2\phi(z_t)} \le \sqrt{v}\,\|y-x\|_{\nabla^2\phi(z_t)}.$$

Hence, we have $\alpha'(t) \ge \frac{1}{v}\alpha(t)^2$. If $\alpha(0) \le 0$, then we are done. Otherwise, α is increasing and hence $\alpha(1) > 0$. Since $\frac{1}{\alpha(1)} \le \frac{1}{\alpha(0)} - \frac{1}{v}$. So, $\alpha(0) \le v$. □

Lemma 5.6.9 (Duality Gap)**.** *Suppose that ϕ is a ν-self concordant barrier, we have that*

$$\langle c, x_t\rangle \le \langle c, x^*\rangle + \frac{\nu}{t}.$$

More generally, for any x such that $\|tc + \nabla\phi(x)\|_{(\nabla^2\phi(x))^{-1}} \le \frac{1}{6}$, we have that

$$\langle c, x\rangle \le \langle c, x^*\rangle + \frac{\nu + \sqrt{\nu}}{t}.$$

Proof. Let x^* be a minimizer of $c^\top x$ on K. By optimality, we have $tc + \nabla\phi(x) = 0$. Therefore, we have

$$\langle c, x_t\rangle - \langle c, x^*\rangle = \frac{1}{t}\langle \nabla\phi(x_t), x^* - x_t\rangle \le \frac{\nu}{t}.$$

For the second result, let $f(x) = tc^T x + \phi(x)$. Lemma 5.5.8 shows that

$$\|x - x_t\|_x \le 2\|\nabla f(x)\|_{\nabla^2 f(x)^{-1}} \le \frac{1}{3}.$$

Hence,

$$\langle c, x - x_t\rangle \le \|c\|_{\nabla^2\phi(x)^{-1}}\|x - x_t\|_x \le \frac{1}{3}\|c\|_{\nabla^2\phi(x)^{-1}}$$

Using $c = \frac{tc+\nabla\phi(x)}{t} - \frac{\nabla\phi(x)}{t}$, we have

$$\begin{aligned}\langle c, x - x_t\rangle &\le \frac{1}{3t}\left(\|tc + \nabla\phi(x)\|_{\nabla^2\phi(x)^{-1}} + \|\nabla\phi(x)\|_{\nabla^2\phi(x)^{-1}}\right)\\ &\le \frac{1}{3t}\left(\frac{1}{6} + \sqrt{\nu}\right) \le \frac{\sqrt{\nu}}{t}.\end{aligned}$$

This gives the result. □

Hence, it suffices to end with $t = (\nu + 1)/\varepsilon$, which is exactly same as the previous lecture.

Theorem 5.6.10. *Given a ν-self concordant barrier ϕ and its minimizer. We can find $x \in K$ such that $c^\top x \leq c^\top x^* + \epsilon$ in*

$$O(\sqrt{\nu}\log(\frac{\nu}{\epsilon}\|c\|_{\nabla^2\phi(x)^{-1}}))$$

many iterations.

Proof. We prove by induction that $\|\nabla f_t(x)\|_{\nabla^2 f_t(x))^{-1}} \leq \frac{1}{6}$ at the beginning of each iteration. This is true at the beginning by the definition of initial t. By Lemma 5.5.6, after the Newton step, we have

$$\|\nabla f_t(x)\|_{\nabla^2 f_t(x)^{-1}} \leq (\frac{1/6}{1-1/6})^2 = \frac{1}{25}.$$

Let $t' = (1+h)t$ with $h = \frac{1}{9\sqrt{\nu}}$. Note that $\nabla f_{t'}(x) = (1+h)tc + \nabla\phi(x)$ and hence $\nabla f_{t'}(x) = (1+h)\nabla f_t(x) - h\nabla\phi(x)$. Therefore, we have that

$$\begin{aligned}
\|\nabla f_{t'}(x)\|_{\nabla^2 f_{t'}(x)^{-1}} &= \|(1+h)\nabla f_t(x) - h\nabla\phi(x)\|_{\nabla^2 f_t(x)^{-1}} \\
&\leq (1+h)\|\nabla f_t(x)\|_{\nabla^2 f_t(x)^{-1}} + h\|\nabla\phi(x)\|_{\nabla^2\phi(x)^{-1}} \\
&\leq \frac{1+h}{25} + h\sqrt{\nu} \leq \frac{1}{6}.
\end{aligned}$$

This complete the induction. □

Chapter 6

Sparsification

In this chapter, we study some randomization techniques for faster convex optimization.

6.1 Subspace embedding

In this section and next section, we consider the least squares regression problem

$$\min_x \|Ax - b\|_2^2$$

where $A \in \mathbb{R}^{n \times d}$ with $n \gg d$. The gradient of the function is $2A^\top Ax - 2A^\top b$. Therefore, the solution is given by

$$x = (A^\top A)^{-1} A^\top b.$$

If the matrix $A^\top A \in \mathbb{R}^{d \times d}$ is given, then, we can solve the equation above in time d^ω. If $n > d^\omega$, then the bottleneck is simply to compute $A^\top A$. The following lemma shows that it suffices to approximate $A^\top A$.

The simplest iteration is the Richardson iteration:

$$x^{(k+1)} = A^T b + (I - A^\top A)x^{(k)} = x^{(k)} + A^T b - A^\top A x^{(k)}.$$

To ensure this converges, we scale down $A^T A$ by its largest eigenvalue so that $A^T A \prec I$. This gives a bound of $O(\kappa(A^\top A)\log(1/\epsilon))$ on the number of iterations to get ϵ error where $\kappa(A^\top A) = \lambda_{\max}(A^\top A)/\lambda_{\min}(A^\top A)$. More generally, one can use *pre-conditioning.*

Lemma 6.1.1. *Given a matrix M such that $A^\top A \preceq M \preceq \kappa \cdot A^\top A$ for some $\kappa \geq 1$. Consider the algorithm $x^{(k+1)} = x^{(k)} - M^{-1}(A^\top A x^{(k)} - A^\top b)$. Then, we have that*

$$\|x^{(k)} - x^*\|_M \leq \left(1 - \frac{1}{\kappa}\right)^k \|x^{(0)} - x^*\|_M.$$

where $r^{(k)} = A^\top A x^{(k)} - A^\top b$.

Remark 6.1.2. The proof also shows why the choice of norm above is the natural one. In this norm, the residual drops geometrically.

Proof. Using $x^* = (A^\top A)^{-1} A^\top b$ and the formula of $x^{(k+1)}$, we have

$$\begin{aligned} x^{(k+1)} - x^* &= x^{(k)} - M^{-1}(A^\top A x^{(k)} - A^\top b) - x^* \\ &= x^{(k)} - M^{-1}(A^\top A x^{(k)} - A^\top A x^*) - x^* \\ &= \left(I - M^{-1} A^\top A\right)\left(x^{(k)} - x^*\right). \end{aligned}$$

Therefore, the norm of error is

$$\begin{aligned} \|x^{(k+1)} - x^*\|_M^2 &= r^{(k)\top}(I - A^\top A M^{-1}) M (I - M^{-1} A^\top A) r^{(k)} \\ &= r^{(k)} M^{\frac{1}{2}} (I - H)^2 M^{\frac{1}{2}} r^{(k)} \end{aligned}$$

where $H = M^{-\frac{1}{2}} A^\top A M^{-\frac{1}{2}}$. Note that the eigenvalues of H lie between $1/\kappa$ and 1 and hence

$$\lambda_{\max}(I-H)^2 \leq \left(1 - \frac{1}{\kappa}\right)^2.$$

This gives the conclusion. □

Note that $x^\top A^\top A x = \|Ax\|^2$. Alternatively, we can think that our goal is to approximate A by a smaller matrix B s.t. $\|Ax\|^2$ is close to $\|Bx\|^2$ for all x. In this section, we show that we can simply take $B = \Pi A$ for a random matrix Π with relatively few rows. With this choice of M, we can run the Richardson iteration. We need to see if this will make the entire procedure more efficient.

Definition 6.1.3. A matrix $\Pi \in \mathbb{R}^{m \times n}$ is an embedding for a set $S \subset \mathbb{R}^n$ with distortion ϵ if

$$(1-\epsilon)\|y\|^2 \leq \|\Pi y\|^2 \leq (1+\epsilon)\|y\|^2$$

for all $y \in S$.

In this section, we focus on the case that S is a d-dimensional subspace in $\mathbb{R}^n$, namely $S = \{Ax : x \in \mathbb{R}^d\}$. Consider the SVD $A = U\Sigma V^\top$. For any $y \in S$, we have that

$$\|U^\top y\|_2^2 = \|U^\top U \Sigma V^\top x\|_2^2 = x^\top V \Sigma^2 V^\top x = \|Ax\|^2.$$

Therefore, any d-dimensional subspace has a zero distortion embedding of size $d \times n$.

Exercise 6.1.4. For any d-dimensional subspace S, any embedding with distortion $\epsilon < 1$ must have at least d rows.

This embedding is not useful for solving the least squares problem because the solution of the least square problem is simply a closed form of the SVD decomposition $x = V\Sigma^{-1}U^\top b$ and that finding the SVD is usually more expensive.

Oblivious Subspace Embedding via Johnson-Lindenstrauss

Surprisingly, there are random embeddings that have a small distortion without knowledge of the subspace.

Definition 6.1.5. A random matrix $\Pi \in \mathbb{R}^{m \times n}$ is a (d, ϵ, δ)-oblivious subspace embedding (OSE) if for any fixed d-dimensional subspace $S \subset \mathbb{R}^n$, Π is an embedding for S with distortion ϵ with probability at least $1 - \delta$.

The next lemma provides an equivalent definition.

Lemma 6.1.6. *We call Π is a (d, ϵ, δ) OSE if for any matrix $U \in \mathbb{R}^{n \times d}$ with orthonormal columns, we have that*

$$\mathbf{P}(\|U^\top \Pi^\top \Pi U - I_d\|_{\text{op}} \leq \epsilon) \geq \delta.$$

Proof. Let S be the subspace with an orthonormal basis $U \in \mathbb{R}^{n \times d}$, namely $S = \{y : Uz\}$ Then, the condition

$$(1-\epsilon)\|y\|^2 \leq \|\Pi y\|^2 \leq (1+\epsilon)\|y\|^2$$

can be rewritten as

$$(1-\epsilon)U^\top U \preceq U^\top \Pi^\top \Pi U \preceq (1+\epsilon)U^\top U.$$

Using $U^\top U = I_d$, we have

$$(1-\epsilon)I_d \preceq U^\top \Pi^\top \Pi U \preceq (1+\epsilon)I_d.$$

□

For the special case $d = 1$, the definition becomes

$$\mathbf{P}\left[|\|\Pi a\|_2^2 - 1| \leq \epsilon\right] \geq 1 - \delta$$

for all unit vectors a. An OSE for $d = 1$ is given by the Johnson-Lindenstrauss Lemma. The original version was for a uniform random subspace of dimension d, but later versions extended this extended this to Gaussian, Bermoulli and more general random matrices [62].

Lemma 6.1.7 (Johnson-Lindenstrauss Lemma)**.** *Let $\Pi \in \mathbb{R}^{m\times n}$ be a random matrix with i.i.d entries from $N(0, \frac{1}{\sqrt{m}})$ or uniformly sampled from $\pm\frac{1}{\sqrt{m}}$ with $m = \Theta(\frac{1}{\epsilon^2}\log(\frac{1}{\delta}))$. Then, Π is a $(1, \epsilon, \delta)$ OSE.*

We will skip the proof for this as we will prove a more general result later. Next, we show that any OSE for $d = 1$ is a OSE for general d. Therefore, it suffices to focus on the case $d = 1$. First, we need a lemma about ϵ-net on S^{n-1}.

Lemma 6.1.8. *For any $\epsilon > 0$ and any $n \in \mathbb{N}$, there are $(1 + \frac{2}{\epsilon})^n$ unit vectors $x_i \in \mathbb{R}^n$ such that for any unit vector $x \in \mathbb{R}^n$, there is an i such that $\|x - x_i\|_2 \le \epsilon$.*

Remark. We call the points $\{x_i\}_i$ the ϵ-net on S^{n-1}.

Proof. We consider the following algorithm:

1. $V = S^{n-1}$. $i \leftarrow 1$.
2. While there is $x_i \in V$
   (a) $V \leftarrow V\backslash B(x_i, \epsilon)$.
   (b) $i \leftarrow i + 1$.

Let $\{x_1, x_2, \cdots, x_N\}$ be the points it found. By the construction, it is clear that $\|x - x_i\| \le \epsilon$ for some i (else the procedure would continue). Now consider balls of radius $\epsilon/2$ centered at each x_i. To bound the number of points, we note that all $B(x_i, \frac{\epsilon}{2})$ are disjoint and that all balls lie in $B(0, 1 + \frac{\epsilon}{2})$. Therefore,

$$N \cdot \mathrm{vol}(B(0, \frac{\epsilon}{2})) \le \mathrm{vol}(B(0, 1 + \frac{\epsilon}{2})).$$

and hence $N \le (1 + \frac{2}{\epsilon})^n$. □

Lemma 6.1.9. *Suppose that Π is a $(1, \epsilon, \delta)$ OSE. Then, Π is a $(d, 4\epsilon, 5^d\delta)$ OSE.*

Proof. Let $N = \{x_i\}_{i=1}$ be a $\frac{1}{2}$-net for S^{d-1}. Then, for any x, we have $x_1 \in N$ such that $\|x - x_1\|_2 \le \frac{1}{2}$. Using the $\frac{1}{2}$-net guarantee on the vector $x - x_1$, we can find $x_2 \in N$ and $0 \le t_2 \le \frac{1}{2}$ such that

$$\|x - x_1 - t_2x_2\| \le \frac{1}{4}.$$

Continuing similarly, we have $x = \sum_{i=1}^{\infty} t_ix_i$ with $0 \le t_i \le \frac{1}{2^{i-1}}$. Hence, we have that

$$\begin{aligned} x^\top(U^\top\Pi^\top\Pi U - I_d)x &= \sum_{i,j} t_it_jx_i^\top(U^\top\Pi^\top\Pi U - I_d)x_j \\ &\le \sum_{i,j} t_it_j \cdot \max_{x\in N} \left|x^\top \left|U^\top\Pi^\top\Pi U - I_d\right| x\right| \\ &\le 4\max_{x\in N} \left|x^\top \left|U^\top\Pi^\top\Pi U - I_d\right| x\right| \\ &= 4\max_{x\in UN} \left|\|\Pi x\|^2 - 1\right| \end{aligned}$$

Now, using that Π is $(1, \epsilon, \delta)$ OSE, we have that

$$\mathbf{P}(\left|\|\Pi x\|^2 - 1\right| \le \epsilon) \ge 1 - \delta.$$

Take a union bound over all $x \in UN$, we have

$$\mathbf{P}(x^\top(U^\top\Pi^\top\Pi U - I_d)x \le 4\epsilon) \ge 1 - 5^d\delta.$$

□

This reduction and the Johnson-Lindenstrauss Lemma shows that a random $\pm\frac{1}{\sqrt{m}}$ is a (d,ϵ,δ) OSE with

$$m = \Theta(\frac{1}{\epsilon^2}(d + \log(\frac{1}{\delta}))).$$

As we discussed before that any (d,ϵ,δ) OSE should have at least d rows. Therefore, the number of rows of this OSE is tight for the regime $\epsilon = \Theta(1)$. We only need an OSE for $\epsilon = \Theta(1)$ because of Lemma 6.1.1; by iterating we can get any ϵ with an overhead of $\log(1/\epsilon)$. Unfortunately, computing ΠA is in fact more expensive than $A^\top A$. The first involves multiplying $\Theta(d) \times n$ and $n \times d$ matrix, the second one involves multiplying $d \times n$ and $n \times d$ matrix.

Sparse JL

We consider the sparse matrix $\Pi \in \mathbb{R}^{m\times n}$ where each entry is $\pm\frac{1}{\sqrt{s}}$ with probability $\frac{s}{m}$ and 0 otherwise. We will show that this random matrix is a (d,ϵ,δ) OSE for $s = \Theta(\log^2(d/\delta)/\epsilon^2)$ and $m = \Theta(d\log(\frac{d}{\delta})/\epsilon^2)$. When $n = d = 1$, $\|\Pi x\|^2$ is the number of non-zeros in the only column. Therefore, we indeed need that s scales like $1/\epsilon^2$.

Remark 6.1.10. It turns out that one can select exactly s non-zeros for each column. This allows us to use $s = \Theta(\frac{\log(d/\delta)}{\epsilon})$. The proof of this is a slightly more complicated due to the lack of independence [13].

To analyze $U^\top\Pi^\top\Pi U$, we note that

$$U^\top\Pi^\top\Pi U = \sum_{r=1}^{m}(\Pi U)_r^\top(\Pi U)_r.$$

Since each row of Π is independent, we can use two matrix concentration bounds to analyze the sum above. See [59] for a survey on matrix concentration bounds.

Theorem 6.1.11 (Matrix Chernoff)**.** *Suppose we have a sequence of independent random self-adjoint matrices $M_j \in \mathbb{R}^{n\times n}$ such that $\mathbf{E}M_j = I$ and $0 \preceq M_j \preceq R\cdot I$. Then for $T = \frac{R}{\varepsilon^2}\log\frac{n}{\delta}$,*

$$(1 - O(\varepsilon))I \preceq \frac{1}{T}\sum_{j=1}^{T} M_j \preceq (1 + O(\varepsilon))I$$

with probability $1-\delta$.

Lemma 6.1.12 (Hanson-Wright Inequality)**.** *For $\sigma_1,\cdots,\sigma_n$ are independent random variables with $\mathbf{E}\sigma_i = 0$, $|\sigma_i| \le 1$. Let $A \in \mathbb{R}^{n\times n}$, we have that*

$$\left|\sigma^\top A\sigma - \mathbf{E}\sigma^\top A\sigma\right| \lesssim \|A\|_F\sqrt{\log(\frac{1}{\delta})} + \|A\|_{\mathrm{op}}\log(\frac{1}{\delta})$$

with probability $1-\delta$. (Here we use $A \lesssim B$ to denote $A = O(B)$.)

For our problem, we set $M_r = m\cdot U^\top\pi_r\pi_r^\top U$ where π_r is the r-th row of Π. One can check that $M_r \succeq 0$ and that $\mathbf{E}M_r = I$. Next, we note that

$$M_r \preceq m\cdot\pi_r^\top UU^\top\pi_r\cdot I.$$

With small probability, $\pi_r^\top UU^\top\pi_r$ can be huge. However, as long as $\pi_r^\top UU^\top\pi_r$ is bounded by R with probability $1-\delta$, then we can still use the matrix Chernoff bound above. To bound $\pi_r^\top UU^\top\pi_r$, we will use the following large deviation inequality.

Lemma 6.1.13. *Assume that $s \gg \frac{m}{n}\log(\frac{1}{\delta})$,$\log^2(\frac{1}{\delta})/\epsilon^2$ and $m \gg d\log(\frac{1}{\delta})/\epsilon^2$. Then,*

$$\pi_r^\top UU^\top\pi_r \le \frac{\epsilon^2}{\log(d/\delta)}$$

with probability $1-\delta$. (Here $\gg$ means greater by a sufficiently large constant factor).

Proof. Note that $\pi_r \in \mathbb{R}^n$ with each entry non-zero with probability $\frac{s}{m}$. By the Chernoff bound, one can show that π_r has at most $2sn/m$ non-zeros with probability $1-\delta$ using that $sn/m \gg \log(\frac{1}{\delta})$. Let I be the indices that π_r

are non-zero. Conditional on the set I and that $|I| \leq 2sn/m$, note that $\sigma \stackrel{\text{def}}{=} \pi_r|_I$ is a random $\pm\frac{1}{\sqrt{s}}$ vector. Let $P = (UU^\top)|_{I\times I}$. Then, we have that

$$\pi_r^\top UU^\top \pi_r = \sigma^\top P\sigma.$$

The Hanson-Wright inequality shows that

$$\left|\sigma^\top P\sigma - \mathbf{E}\sigma^\top P\sigma\right| \lesssim \frac{1}{s}\|P\|_F\sqrt{\log(\frac{1}{\delta})} + \frac{1}{s}\|P\|_{\text{op}}\log(\frac{1}{\delta}).$$

Note that $UU^\top$ is a projection matrix and hence $\|P\|_{\text{op}} \leq 1$. Since $P \succeq 0$, we have that

$$\|P\|_F = \sqrt{\text{tr}P^2} \leq \sqrt{\|P\|_{\text{op}} \cdot \text{tr}P} \leq \sqrt{\text{tr}P}.$$

Also, we have that $\mathbf{E}\sigma^\top P\sigma = \frac{1}{s}\text{tr}P$. Hence, we have

$$\sigma^\top P\sigma \leq \frac{1}{s}\left(\text{tr}P + O\left(\sqrt{\text{tr}P \cdot \log\frac{1}{\delta}} + \log\frac{1}{\delta}\right)\right)$$

with probability $1-\delta$. Note that $\text{tr}UU^\top = \text{tr}U^\top U = d$ and that P is a random diagonal block of $UU^\top$of size at most $2sn/m$. By the Chernoff bound, one can show that

$$\text{tr}P \leq \frac{4sd}{m}.$$

Hence, we have

$$\sigma^\top P\sigma \leq \frac{4d}{m} + \frac{1}{s}O\left(\sqrt{\frac{sd}{m} \cdot \log\frac{1}{\delta}} + \log\frac{1}{\delta}\right)$$

with probability $1-\delta$. Using $s \gg \log^2(\frac{d}{\delta})/\epsilon^2$ and $m \gg d\log(\frac{d}{\delta})/\epsilon^2$, we have $\sigma^\top P\sigma \leq \frac{\epsilon^2}{\log(d/\delta)}$ with probability $1-\delta$. □

Now we can prove the main theorem.

Theorem 6.1.14. *Consider a random sparse matrix $\Pi \in \mathbb{R}^{m\times n}$ where each entry is $\pm\frac{1}{\sqrt{s}}$ with probability $\frac{s}{m}$ and 0 otherwise. There exist constants c_1, c_2 such that for $m = c_1\frac{d\log(\frac{d}{\delta})}{\epsilon^2}$ and $s = c_2\frac{\log^2(\frac{d}{\delta})}{\epsilon^2}$ Π is an $O(d,\epsilon,\delta)$-OSE.*

Proof. By the previous lemma, we have that

$$\pi_r^\top UU^\top \pi_r \leq \frac{\epsilon^2}{\log(d/\delta)}$$

with high probability. Under this event, we have that

$$(1-O(\varepsilon))I \preceq \sum_{r=1}^m \pi_r^\top UU^\top \pi_r \preceq (1+O(\varepsilon))I.$$

□

Back to Regression

In the last subsection, we presented a proof sketch of a suboptimal OSE. For completeness, we state a tighter version here:

Algorithm 15: `RegressionUsingOSE`

Input: a matrix $A \in \mathbb{R}^{n \times d}$, a vector $b \in \mathbb{R}^d$, success probability δ and target accuracy ϵ.
Let Π be a $O(d, \frac{1}{4}, \delta)$ OSE constructed by Theorem 6.1.14.
Compute $A' = 2\Pi A$, $M = A'^\top A'$ and M^{-1}.
$x^{(1)} = 0$.
for $k = 1, 2, \cdots, 4\log(\frac{1}{\epsilon})$ **do**
  $x^{(k+1)} = x^{(k)} - M^{-1}(A^\top A x^{(k)} - A^\top b)$
end
return $x^{(\text{last})}$.

Theorem 6.1.15. *Given any matrix $A \in \mathbb{R}^{n \times d}$, any vector $b \in \mathbb{R}^d$, the algorithm* `RegressionUsingOSE` *returns a vector x such that*

$$\|A^\top A x - A^\top b\|_{(A^\top A)^{-1}} \leq \epsilon \|A^\top b\|_{(A^\top A)^{-1}}.$$

with probability $1 - \delta$ in $\widetilde{O}(\mathrm{nnz}(A) + d^\omega)$ time.

Proof. The guarantee of x follows from the definition of OSE and Lemma 6.1.1. We note that ΠA simply involves duplicating each row of A into roughly s many rows in ΠA. Hence, the cost of computing ΠA is $O(s \cdot \mathrm{nnz}(A)) = \widetilde{O}(\mathrm{nnz}(A))$. The cost of computing M takes $\widetilde{O}(d^\omega)$ time. Computing M^{-1} takes $\widetilde{O}(d^\omega)$ time. The loop takes $\widetilde{O}(\mathrm{nnz}(A))$ time. This explain the total time. □

Linear regression can also be solved in time $O\left(\mathrm{nnz}(A) + d^{O(1)}\right)$ via a very sparse embedding: each column of Π picks exactly one nonzero entry in a random location. This was analyzed by Clarkson and Woodruff [64]. See also [30] for a survey.

6.2 Leverage Score Sampling

Here we give a way to reduce solving a tall linear regression system into a sequence of submatrices:

Theorem 6.2.1 ([14])**.** *Given a matrix $A \in \mathbb{R}^{n \times d}$. Let $T(m)$ be the cost of solving $B^\top B x = b$ among all m distinct subrows B of A. Then, we have that*

$$T(n) = O^*(\mathrm{nnz}(A) + T(O(d \log d))).$$

Remark. Here we use O^* to emphasize there is some dependence on $\log(\frac{1}{\epsilon})$ suppressed for notational simplicity. Lemma 6.1.1 shows that once we can solve the system with constant approximation, we can repeat $\log(\frac{1}{\epsilon})$ times to get an ϵ-accurate solution.

Leverage scores

The key concept in this reduction is leverage score.

Definition 6.2.2. Given a matrix $A \in \mathbb{R}^{n \times d}$, let $a_i^\top$ be the i^{th} row of A. The leverage score of the i^{th} row of A is

$$\sigma_i(A) \stackrel{\text{def}}{=} a_i^\top (A^\top A)^+ a_i.$$

Note that $\sigma(A)$ is the diagonal of the projection matrix $A(A^\top A)^+ A^\top$. Since $0 \preceq A(A^\top A)^+ A^\top \preceq I$, we have that $0 \leq \sigma_i(A) \leq 1$. Moreover, since $A(A^\top A)^+ A^\top$ is a projection matrix, the sum of A's leverage scores (its trace) is

equal to the rank of A:

$$\sum_{i=1}^{n} \sigma_i(A) = \mathrm{tr}(A(A^\top A)^+ A^\top) = \mathrm{rank}(A(A^\top A)^+ A^\top) = \mathrm{rank}(A) \leq d. \tag{6.2.1}$$

The leverage score measures the "importance" of a row in forming the row space of A. If a row has a component orthogonal to all other rows, its leverage score is 1. Removing it would decrease the rank of A, completely changing its row space. The *coherence* of A is $\|\sigma(A)\|_\infty$. If A has low coherence, no particular row is especially important. If A has high coherence, it contains at least one row whose removal would significantly affect the composition of A's row space. The following two characterizations that helps with this intuition.

Lemma 6.2.3. *For all $A \in \mathbb{R}^{n\times d}$ and $i \in [n]$ we have that*

$$\sigma_i(A) = \min_{A^\top x = a_i} \|x\|_2^2.$$

where a_i is the i^{th} row of A.

Lemma 6.2.4. *For all $A \in \mathbb{R}^{n\times d}$ and $i \in [n]$, we have that $\sigma_i(A)$ is the smallest t such that*

$$a_i a_i^\top \preceq t \cdot A^\top A. \tag{6.2.2}$$

Sampling rows from A according to their exact leverage scores gives a spectral approximation for A with high probability. Sampling by leverage score overestimates also suffices.

Lemma 6.2.5. *Given a vector u of leverage score overestimates, i.e., $\sigma_i(A) \leq u_i$ for all i, define*

$$X = \frac{1}{p_i} a_i a_i^\top \quad \text{with probability } p_i = \frac{u_i}{\|u\|_1}.$$

For $T = \Omega(\frac{\|u\|_1 \log n}{\varepsilon^2})$, with probability $1 - \frac{1}{n^{O(1)}}$, we have that

$$(1-\varepsilon)A^\top A \preceq \frac{1}{T}\sum_{i=1}^{T} X_i \preceq (1+\varepsilon)A^\top A$$

where X_i are independent copies of X.

Proof. Note that $\mathbf{E}X = A^\top A$ and that

$$0 \preceq X = \frac{1}{p_i} a_i a_i^\top \preceq \frac{\|u\|_1}{\sigma_i} a_i a_i^\top \preceq \|u\|_1 \cdot A^\top A.$$

Now, the statement simply follows from the matrix Chernoff bound with $M_k = (A^\top A)^{-\frac{1}{2}} X_k (A^\top A)^{-\frac{1}{2}}$ and $R = \|u\|_1$. □

Combining Lemma 6.2.5 and Lemma 6.1.1, we have that

$$T(n) = \text{cost of computing } \sigma_i + O^*(\mathrm{nnz}(A) + T(d \log d)) \tag{6.2.3}$$

where we used that $\|\sigma\|_1 = O(d)$. However, computing σ exactly is too expensive for many purposes. In [**?**], they showed that we can compute leverage scores approximately by solving only polylogarithmically many regression problems. This result uses the fact that

$$\sigma_i(A) = \left\|A(A^\top A)^+ A^\top e_i\right\|_2^2$$

and that by the Johnson-Lindenstrauss Lemma these lengths are preserved up to a multiplicative error if we project these vectors to a random low-dimensional subspace.

In particular, this lemma shows that we can approximate $\sigma_i(A)$ via $\left\|\Pi A(A^\top A)^+ A^\top e_i\right\|_2^2$. The benefit of this is that we can compute $\Pi A(A^\top A)^+$ by solving $\frac{\log n}{\varepsilon^2}$ many linear systems. In other words, we have that

$$\text{cost of approximating } \sigma_i = O^*(\mathrm{nnz}(A)) + O^*(T(n)).$$

Putting it into (6.2.3), we have this useless formula

$$T(n) = O^*(\mathrm{nnz}(A) + T(n) + T(d \log d)).$$

This is a chicken and egg problem. To solve an overdetermined system faster, we want to use leverage score to sample the rows. And to approximate the leverage score, we need to solve the original overdetermined system. (Even worse, we need to solve it a few times.)

Uniform Sampling

The key idea to break this chicken and egg problem is to use uniform sampling of the rows of A. We define

$$\sigma_{i,S} = a_i^\top (A_S^\top A_S)^{-1} a_i$$

where A_S is A restricted to rows in S. The set S will be a random sample of k rows of A. Note that $A_S^\top A_S$ is an overestimate of σ_i. Hence, it suffices to bound $\|\sigma_{i,S}\|_1$. The key lemma is the following:

Lemma 6.2.6. *We have that*

$$\mathbf{E}_{|S|=k} \sum_{i=1}^n \sigma_{i,S\cup\{i\}} \leq \frac{nd}{k}.$$

Proof. Note that

$$\mathbf{E}_{|S|=k} \sum_{i=1}^n \sigma_{i,S\cup\{i\}} = \mathbf{E}_{|S|=k} \sum_{i\notin S} \sigma_{i,S\cup\{i\}} + \mathbf{E}_{|S|=k} \sum_{i\in S} \sigma_{i,S\cup\{i\}}.$$

Note that $\sum_{i\in S} \sigma_{i,S\cup\{i\}} = \sum_{i\in S} \sigma_{i,S} \leq d$. Hence, the second term is bounded by n.

For the first term, we note that "sample a set S of size k, then sample $i \notin S$" is same as "sample a set T of size $k+1$, then sample $i \in T$". Hence, we have

$$\begin{aligned}\mathbf{E}_{|S|=k} \mathbf{E}_{i\notin S} \sigma_{i,S\cup\{i\}} &= \mathbf{E}_{|T|=k+1} \mathbf{E}_{i\in T} \sigma_{i,T} \\ &\leq \mathbf{E}_{|T|=k+1} \frac{d}{k+1} = \frac{d}{k+1}\end{aligned}$$

Hence, we have that

$$\mathbf{E}_{|S|=k} \sum_{i=1}^n \sigma_{i,S\cup\{i\}} \leq \frac{d}{k+1}(n-k) + d = d \cdot \frac{n+1}{k+1}.$$

□

Next, using Sherman-Morrison formula, we have that

$$\sigma_{i,S\cup\{i\}} = \begin{cases} \sigma_{i,S} & \text{if } i \in S \\ \frac{1}{1+\frac{1}{\sigma_{i,S}}} & \text{otherwise} \end{cases}.$$

Namely, we can compute $\sigma_{i,S\cup\{i\}}$ using $\sigma_{i,S}$. Therefore, we have that

$$\text{cost of approximating } \sigma_{i,S\cup\{i\}} = O^*(\mathrm{nnz}(A)) + O^*(T(k)).$$

Using Lemma 6.2.6, we have now

$$T(n) = O^*(\mathrm{nnz}(A) + T(k) + T(\frac{nd}{k} \log d)).$$

Picking $k = \sqrt{nd \log d}$, we have that

$$T(n) = O^*(\mathrm{nnz}(A) + T(\sqrt{nd \log d})).$$

Repeating this process $\tilde{O}(1)$ times, we have

$$T(n) = O^*(\mathrm{nnz}(A) + T(O(d \log d))).$$

6.3 Stochastic Gradient Descent

In this section, we study problems of the form $\sum f_i$ where each f_i are convex and the sum can be either infinite or finite.

Stochastic Problem

In this part, we discuss the problem

$$\min_x F(x) \stackrel{\text{def}}{=} \mathbf{E}_{f\sim\mathcal{D}} f(x)$$

where $\mathcal{D}$ is a distribution of convex functions in $\mathbb{R}^d$. The goal is to find the minimizer $x^* = \operatorname{argmin}_x F(x)$. Suppose we observed $f_1, f_2, \cdots, f_T$ samples from $\mathcal{D}$. Ideally, we wish to approximate x^* by the empirical risk minimizer

$$x_{\text{ERM}}^{(T)} = \arg\min_x F_T(x) \stackrel{\text{def}}{=} \frac{1}{T}\sum_{i=1}^{T} f_i(x).$$

It is known that $x_{\text{ERM}}^{(T)}$ is optimal in a certain sense in spite of its computational cost. Therefore, to discuss the efficiency of an optimization algorithm for $F(x)$, it is helpful to consider the ratio

$$\frac{\mathbf{E}F(x^{(T)}) - F(x^*)}{\mathbf{E}F(x_{\text{ERM}}^{(T)}) - F(x^*)}.$$

We will first discuss the term $\mathbf{E}F(x_{\text{ERM}}^{(T)}) - F(x^*)$. As an example, consider the simplest one-dimensional problem $F(x) = \mathbf{E}_{b\sim N(0,I)}(x-b)^2$. Note that $x^{(T)}$ is simply average of T standard normal variables and hence

$$\begin{aligned}
\mathbf{E}F(x_{\text{ERM}}^{(T)}) - F(x^*) &= \mathbf{E}_{b_1,b_2,\cdots,b_T}\mathbf{E}_b(x_{\text{ERM}}^{(T)} - b)^2 - b^2 \\
&= \mathbf{E}_{b_1,b_2,\cdots,b_T}(x_{\text{ERM}}^{(T)})^2 \\
&= \mathbf{E}_{b_1,b_2,\cdots,b_T}(\frac{1}{T}\sum_{i=1}^{T} b_i)^2 = \frac{1}{T}.
\end{aligned}$$

In general, the following lemma shows that

$$\mathbf{E}F(x_{\text{ERM}}^{(T)}) - F(x^*) \to \frac{\sigma^2}{T} \quad \text{where } \sigma^2 = \frac{1}{2}\mathbf{E}\|\nabla f(x^*)\|^2_{(\nabla^2 F(x^*))^{-1}}.$$

where $\frac{\sigma^2}{T}$ is called the Cramer-Rao bound.

Lemma 6.3.1. *Suppose that f is μ-strongly convex with Lipschitz Hessian for all $f \sim \mathcal{D}$ for some $\mu > 0$. Suppose that $\mathbf{E}_{f\sim\mathcal{D}}\|\nabla f(x^*)\|^2 < +\infty$. Then, we have that*

$$\lim_{N\to+\infty} \frac{\mathbf{E}F(x_{\text{ERM}}^{(N)}) - F(x^*)}{\sigma^2/N} = 1.$$

Remark. The statement holds with much weaker assumptions and the rate of convergence can be made quantitative.

Proof. We first prove $x_{\text{ERM}}^{(N)} \to x^*$ as $N \to +\infty$. By the optimality condition of $x_{\text{ERM}}^{(N)}$, we have that

$$0 = \nabla F_N(x_{\text{ERM}}^{(N)}) = \nabla F_N(x^*) + \nabla^2 F_N(\widetilde{x})(x_{\text{ERM}}^{(N)} - x^*). \tag{6.3.1}$$

By the μ-strongly convexity, we have

$$\|x_{\text{ERM}}^{(N)} - x^*\|_2^2 \le \frac{1}{\mu}\|\nabla F_N(x^*)\|^2.$$

Since $\mathbf{E}_{f\sim\mathcal{D}}\nabla f(x^*) = \nabla F(x^*) = 0$, we have

$$\begin{aligned}
\mathbf{E}\|x_{\text{ERM}}^{(N)} - x^*\|_2^2 &\le \frac{1}{\mu}\mathbf{E}\frac{1}{N^2}\sum_i \|\nabla f_i(x^*)\|^2 \\
&= \frac{1}{\mu N}\mathbf{E}_{f\sim\mathcal{D}}\|\nabla f(x^*)\|^2
\end{aligned}$$

Therefore, $x_{\mathrm{ERM}}^{(N)} \to x^*$ as $N \to +\infty$.

Now, to compute the error, Taylor expansion of F at x^* shows that

$$F(x_{\mathrm{ERM}}^{(N)}) - F(x^*) = \frac{1}{2}(x_{\mathrm{ERM}}^{(N)} - x^*)^\top \nabla^2 F(\overline{x})(x_{\mathrm{ERM}}^{(N)} - x^*)$$

for some $\overline{x}$ between x^* and $x_{\mathrm{ERM}}^{(N)}$. Using this and (6.3.1) gives

$$F(x_{\mathrm{ERM}}^{(N)}) - F(x^*) = \frac{1}{2}\nabla F_N(x^*)^\top (\nabla^2 F_N(\widetilde{x}))^{-1} \nabla^2 F(\overline{x}) (\nabla^2 F_N(\widetilde{x}))^{-1} \nabla F_N(x^*).$$

Since $x_{\mathrm{ERM}}^{(N)} \to x^*$ and $\nabla^2 F_N \succeq \mu I$, we have $(\nabla^2 F_N(\widetilde{x}))^{-1} \nabla^2 F(\overline{x}) (\nabla^2 F_N(\widetilde{x}))^{-1} \to (\nabla^2 F(x^*))^{-1}$. Hence, we have

$$\begin{aligned}\lim_{N\to\infty} \mathbf{E} N \cdot (F(x_{\mathrm{ERM}}^{(N)}) - F(x^*)) &= \lim_{N\to\infty} \frac{N}{2} \mathbf{E} \nabla F_N(x^*)^\top (\nabla^2 F(x^*))^{-1} \nabla F_N(x^*) \\ &= \frac{1}{2}\mathbf{E}\|\nabla f(x^*)\|^2_{(\nabla^2 F(x^*))^{-1}}.\end{aligned}$$

□

Now, we discuss how to achieve a bound similar to σ^2/T using stochastic gradient descent. Since gradient descent is a first order method, we can only achieve a bound related to the ℓ_2 norm, $\mathbf{E}\|\nabla f(x^*)\|_2^2$, instead of the inverse Hessian norm. We need a lemma relating $\nabla f(x^*)$ and $\nabla f(x)$.

Exercise 6.3.2. Suppose f has L-Lipschitz gradient. Then, $\|\nabla f(x) - \nabla f(y)\|^2 \le 2L \cdot D_f(y, x)$ for any x, y.

Lemma 6.3.3. *Suppose f has L-Lipschitz gradient for all $f \in \mathcal{D}$. Let x^* be the minimizer of F. Then, we have*

$$\mathbf{E}_{f\sim\mathcal{D}}\|\nabla f(x) - \nabla f(x^*)\|_2^2 \le 2L \cdot (F(x) - F(x^*)).$$

Proof. By the exercise above, we have

$$\|\nabla f(x) - \nabla f(x^*)\|^2 \le 2L \cdot (f(x) - f(x^*) - \nabla f(x^*)^\top (x - x^*)).$$

Taking expectation, we have the result. □

Algorithm 16: `StochasticGradientDescent` (SGD)

Input: Initial point $x^{(0)} \in \mathbb{R}^d$, step size $h > 0$.
for $k = 0, 1, \cdots, T$ **do**
  Sample $f \sim \mathcal{D}$.
  $x^{(k+1)} \leftarrow x^{(k)} - h \cdot \nabla f(x^{(k)})$.
end

Theorem 6.3.4. *Suppose f has L-Lipschitz gradient for all $f \in \mathcal{D}$ and F is μ strongly convex. Let x^* be the minimizer of F and $\sigma^2 = \frac{1}{2}\mathbf{E}\|\nabla f(x^*)\|^2$. For step size $h \le \frac{1}{4L}$, the sequence $x^{(k)}$ in* `StochasticGradientDescent` *satisfies*

$$\mathbf{E}\|x^{(k)} - x^*\|^2 \le \frac{8h\sigma^2}{\mu} + (1 - \frac{h\mu}{2})^k \cdot \|x^{(0)} - x^*\|^2$$

and

$$\frac{1}{T}\sum_{k=0}^{T-1} \mathbf{E}F(x^{(k)}) - F(x^*) \le 4h\sigma^2 + \frac{\|x^{(0)} - x^*\|^2}{hT}.$$

Proof. Note that

$$\|x^{(k+1)} - x^*\|^2 = \|x^{(k)} - x^*\|^2 - 2h\left\langle x^{(k)} - x^*, \nabla f(x^{(k)})\right\rangle + h^2\|\nabla f(x^{(k)})\|^2.$$

Taking expectation on f and using that

$$\begin{aligned}\mathbf{E}\|\nabla f(x^{(k)})\|^2 &\le 2\mathbf{E}\|\nabla f(x^{(k)}) - \nabla f(x^*)\|^2 + 2\mathbf{E}\|\nabla f(x^*)\|^2 \\ &\le 4L \cdot (F(x^{(k)}) - F(x^*)) + 4\sigma^2,\end{aligned}$$

we have

$$\begin{aligned}\mathbf{E}\|x^{(k+1)} - x^*\|^2 &= \|x^{(k)} - x^*\|^2 - 2h\left\langle x^{(k)} - x^*, \nabla F(x^{(k)})\right\rangle + 4h^2\sigma^2 + 4Lh^2 \cdot (F(x^{(k)}) - F(x^*)) \\ &\le \|x^{(k)} - x^*\|^2 - (2h - 4Lh^2)(F(x^{(k)}) - F(x^*)) + 4h^2\sigma^2. \qquad (6.3.2)\end{aligned}$$

Using $h \le \frac{1}{4L}$ and $F(x^{(k)}) - F(x^*) \le \frac{1}{2\mu}\|\nabla F(x^{(k)})\|_2^2$, we have

$$\mathbf{E}\|x^{(k+1)} - x^*\|^2 \le 4h^2\sigma^2 + (1 - \frac{h\mu}{2}) \cdot \|x^{(k)} - x^*\|^2.$$

The first conclusion follows.

For the second conclusion, (6.3.2) shows that

$$\mathbf{E}F(x^{(k)}) - F(x^*) \le \frac{\mathbf{E}\|x^{(k)} - x^*\|^2 - \mathbf{E}\|x^{(k+1)} - x^*\|^2}{h} + 4h\sigma^2.$$

Hence, we have

$$\frac{1}{T}\sum_{k=0}^{T-1} \mathbf{E}F(x^{(k)}) - F(x^*) \le 4h\sigma^2 + \frac{\|x^{(0)} - x^*\|^2}{hT}$$

□

Exercise 6.3.5. Applying the theorem above twice, show that one can achieve error $\tilde{O}(\frac{\sigma^2}{\mu T})$, which is roughly same as the Cramer-Rao bound.

We note that for many algorithms in this book, such as mirror descent, the stochastic version is obtained by replacing the gradient ∇F by the gradient of a sample, ∇f.

Finite Sum Problem

When $\nabla f(x^*) = 0$ for all $f \sim \mathcal{D}$, stochastic gradient descent converges exponentially. When the function is given by a finite sum $F = \frac{1}{n}\sum f_i$, we can reduce the variance at x^* by replacing

$$\nabla \widetilde{f}_i(x) = \nabla f_i(x) - \nabla f_i(x^{(0)}) + \nabla F(x^{(0)}).$$

Note that if $x^{(0)}$ is close to x^*, then $\nabla \widetilde{f}_i(x^*)$ is small because both $\nabla f_i(x^*) - \nabla f_i(x^{(0)})$ and $\nabla F(x^{(0)})$ are small. Formally, the variance for $\widetilde{f}$ is bounded as follows:

Lemma 6.3.6. *Suppose f has L-Lipschitz gradient for all $f \in \mathcal{D}$. For any $f \in \mathcal{D}$, let $\nabla \widetilde{f}(x) = \nabla f(x) - \nabla f(x^{(0)}) + \nabla F(x^{(0)})$ for some fixed $x^{(0)}$. Then, we have*

$$\mathbf{E}\|\nabla \widetilde{f}(x^*)\|^2 \le 8L \cdot (F(x^{(0)}) - F(x^*)).$$

Proof. Note that

$$\begin{aligned}\mathbf{E}\|\nabla \widetilde{f}(x^*)\|^2 &\le 2\mathbf{E}\|\nabla f(x^*) - \nabla f(x^{(0)})\|^2 + 2\|\nabla F(x^{(0)}) - \nabla F(x^*)\|^2 \\ &= 2\mathbf{E}\|\nabla f(x^*) - \nabla f(x^{(0)})\|^2 + 2\mathbf{E}\|\nabla f(x^{(0)}) - \nabla f(x^*)\|^2 \\ &\le 4L(F(x^{(0)}) - F(x^*)) + 4L(F(x^{(0)}) - F(x^*))\end{aligned}$$

where we used Lemma 6.3.3 at the end. □

Algorithm 17: `StochasticVarianceReducedGradient` (SVRG)

Input: Initial point $x^{(0)} \in \mathbb{R}^d$, step size $h > 0$, restart time m.
$\widetilde{x} = x^{(0)}$.
for $k = 0, 1, \cdots, T$ **do**
  if *k is a multiple of m and $k \neq 0$* **then**
    $x^{(k)} = \frac{1}{m}\sum_{j=1}^{m} x^{(k-j)}$.
    $\widetilde{x} = x^{(k)}$.
  end
  Sample f_i for $i \in [n]$
  $x^{(k+1)} \leftarrow x^{(k)} - h \cdot (\nabla f(x^{(k)}) - \nabla f(\widetilde{x}) + \nabla F(\widetilde{x}))$.
end

Theorem 6.3.7. *Suppose f has L-Lipschitz gradient for all f_i and F is μ strongly convex. Let x^* be the minimizer of F. For step size $h = \frac{1}{64L}$ and $m = \frac{256L}{\mu}$, the sequence $x^{(km)}$ in* `StochasticVarianceReducedGradient` *satisfies*

$$\mathbf{E}F(x^{(km)}) - F(x^*) \leq \frac{1}{2^k} \cdot (F(x^{(0)}) - F(x^*)).$$

In particular, it takes $O((n + \frac{L}{\mu})\log(\frac{1}{\epsilon}))$ gradient computations to find x such that $\mathbf{E}F(x) - F(x^*) \leq \epsilon \cdot (F(x^{(0)}) - F(x^*))$.

Remark. Gradient descent gives $O(n\frac{L}{\mu}\log(\frac{1}{\epsilon}))$ instead because computing ∇F involves computing n gradients ∇f.

Proof. The algorithm consists of $\frac{T}{m}$ phases. For the first phase, we compute $\nabla F(x^{(0)})$. Lemma 6.3.6 shows that

$$\mathbf{E}\|\nabla f(x^{(k)}) - \nabla f(x^{(0)}) + \nabla F(x^{(0)})\|^2 \leq 8L \cdot (F(x^{(0)}) - F(x^*)).$$

Hence, Theorem 6.3.4 shows that

$$\begin{aligned}\frac{1}{m}\sum_{k=0}^{m-1} \mathbf{E}F(x^{(k)}) - F(x^*) &\leq 16hL \cdot (F(x^{(0)}) - F(x^*)) + \frac{F(x^{(0)}) - F(x^*)}{\mu h m} \\ &\leq \frac{1}{2}(F(x^{(0)}) - F(x^*)).\end{aligned}$$

Hence, the error decreases by half each phase. This shows the result. □

6.4 Coordinate Descent

If some of the coordinates are more important than others for optimization, it makes sense to update the important coordinates more often.

Lemma 6.4.1. *Given a convex function f, suppose that $\frac{\partial^2}{\partial x_i^2} f(x) \leq L_i$ for all x and let $L = \sum L_i$. If we sample coordinate i with probability $p_i = \frac{L_i}{L}$, then,*

$$\mathbf{E}_i f(x - \frac{1}{L_i}\frac{\partial}{\partial x_i} f(x) e_i) \leq f(x) - \frac{1}{2L}\|\nabla f(x)\|_2^2.$$

Proof. Note that the function $\zeta(t) = f(x + te_i)$ is L_i smooth. Hence, we have that

$$f(x - \frac{1}{L_i}\frac{\partial}{\partial x_i} f(x) e_i) \leq f(x) - \frac{1}{2L_i}\frac{\partial}{\partial x_i} f(x)^2.$$

Since we sample coordinate i with probability $p_i = \frac{L_i}{L}$, we have that

$$\begin{aligned}\mathbf{E}f(x - \frac{1}{L_i}\frac{\partial}{\partial x_i}f(x)e_i) &= f(x) - \sum_i \frac{L_i}{L}\frac{1}{2L_i}\frac{\partial}{\partial x_i}f(x)^2 \\ &= f(x) - \frac{1}{2L}\sum_i \frac{\partial}{\partial x_i}f(x)^2 \\ &= f(x) - \frac{1}{2L}\|\nabla f(x)\|_2^2 .\end{aligned}$$

□

By the same proof as Theorem 2.3.2, we have the following:

Algorithm 18: `CoordinateDescent` (CD)

Input: Initial point $x^{(0)} \in \mathbb{R}^d$, step size $h > 0$.
for $k = 0, 1, \cdots, T$ **do**
  Sample i with probability proportional to L_i.
  $x^{(k+1)} \leftarrow x^{(k)} - \frac{1}{L_i}\frac{\partial}{\partial x_i}f(x^{(k)})e_i$.
end

Theorem 6.4.2 (Coordinate Descent Convergence)**.** *Given a convex function f, suppose that $\frac{\partial^2}{\partial x_i^2}f(x) \le L_i$ for all x and let $L = \sum L_i$. Consider the algorithm $x^{(k+1)} \leftarrow x^{(k)} - \frac{1}{L_i}\frac{\partial}{\partial x_i}f(x^{(k)})e_i$. Then, we have that*

$$\mathbf{E}f(x^{(k)}) - f(x^*) \le \frac{2LR^2}{k+4} \quad \textit{with} \quad R = \max_{f(x)\le f(x^{(0)})} \|x - x^*\|_2$$

for any minimizer x^* of f. If f is μ strongly convex, then we also have

$$\mathbf{E}f(x^{(k)}) - f(x^*) \le (1 - \frac{\mu}{L})^{\Omega(k)}(f(x^{(0)}) - f(x^*)).$$

Remark. Note that $\frac{L}{d} \le \mathrm{Lip}(\nabla f) \le L$. Therefore gradient descent takes at least $\frac{1}{d}$ times as many steps as coordinate descent while each step takes d times longer (usually).

Chapter 7

Acceleration

7.1 Chebyshev Polynomials

The goal of this section is to introduce some basic results in approximation theory. Suppose magically there was a polynomial $q(x)$ such that $q(0) = 1$ and $q(x) = 0$ for all $x > 0$. By assumption, we know that the constant term in q is 1 and hence we can write

$$q(x) = 1 - xp(x)$$

for some $p(x)$. Therefore, for any positive definite matrix A, we have that

$$I - Ap(A) = 0$$

and hence $A^{-1} = p(A)$. Therefore, we could compute $A^{-1}b$ by calculating $p(A)b$ which takes $\deg(p) \cdot \text{nnz}(A)$ time. Unfortunately, there is no such polynomial q. In this section, we will discuss if there is such polynomial that satisfies the condition above approximately.

We will bound $\inf_{q(0)=1} (\max_i |q(\lambda_i)|)$ for matrices A satisfying $\mu \cdot I \preceq A \preceq L \cdot I$. First, we note that we can choose $q(x) = \left(1 - \frac{x}{L}\right)^k$ and get

$$\inf_{q(0)=1} \max_{\mu \le x \le L} |q(x)| \le \left(1 - \frac{\mu}{L}\right)^k.$$

This corresponds to the Richardson iteration, and the above bound shows that it takes $O(\frac{L}{\mu} \log(\frac{1}{\epsilon}))$ degree to make the error bounded by ϵ. The key fact we will use is that we can in general approximate a polynomial of degree k by a polynomial of degree $\tilde{O}(\sqrt{k})$. The construction uses Chebyshev polynomials. This will imply an $\tilde{O}(\sqrt{\frac{L}{\mu}})$ iteration bound for the conjugate gradient method of the next section.

Definition 7.1.1. For any integer d, the d'th Chebyshev polynomial is defined as the unique polynomial that satisfies

$$T_d(\cos\theta) = \cos(d\theta).$$

Exercise 7.1.2. Show that $T_d(x)$ is a degree $|d|$ polynomial in x and that

$$xT_d(x) = \frac{T_{d+1}(x) + T_{d-1}(x)}{2}. \tag{7.1.1}$$

Theorem 7.1.3 (Chernoff Bound)**.** *For independent random variables $Y_1, \cdots, Y_s$ such that $\mathbf{P}(Y_i = 1) = \mathbf{P}(Y_i = -1) = \frac{1}{2}$, for any $a \ge 0$, we have*

$$\mathbf{P}\left(\left|\sum_{i=1}^{s} Y_i\right| \ge a\right) \le 2\exp(-\frac{a^2}{2s}).$$

Now we are ready to show that there is a degree $\tilde{O}(\sqrt{s})$ polynomial that estimates x^s.

Theorem 7.1.4. *For any positive integers s and d, there is a polynomial p of degree d such that*

$$\max_{x \in [-1,1]} |p(x) - x^s| \le 2\exp(-\frac{d^2}{2s}).$$

Proof. Let Y_i are i.i.d. random variable uniform on $\{-1, 1\}$. Let $Z_s = \sum_{i=1}^s Y_i$. Note that

$$\mathbf{E}_{z\sim Z_s} T_z = \mathbf{E}_{z\sim Z_{s-1}} \frac{1}{2}(T_{z+1} + T_{z-1})$$

Now, (7.1.1) shows that $\mathbf{E}_{z\sim Z_s} T_z(x) = \mathbf{E}_{z\sim Z_{s-1}} x T_z(x)$. By induction, we have

$$\mathbf{E}_{z\sim Z_s} T_z(x) = x^s T_0(x) = x^s. \tag{7.1.2}$$

Now, we define the polynomial $p(x) = \mathbf{E}_{z\sim Z_s} T_z(x) 1_{|z|\le d}$. The error of the polynomial can be bounded as follows

$$\begin{aligned}
\max_{x\in[-1,1]} |p(x) - x^s| &= \max_{x\in[-1,1]} |\mathbf{E}_{z\sim Z_s} T_z(x) 1_{|z|>d}| \\
&\le \max_{x\in[-1,1]} \mathbf{E}_{z\sim Z_s} |T_z(x)| 1_{|z|>d} \\
&\le \mathbf{P}_{z\sim Z_s}(|z| > d) \\
&\le 2\exp(-\frac{d^2}{2s})
\end{aligned}$$

where we used (7.1.2) on the first line, $|T_z(x)| \le 1$ for all $x \in [-1, 1]$ on the third line and Chernoff bound (Theorem 7.1.3) on the last line. □

Remark. This proof comes from [55]. Please see [55] for the approximation of other functions.

By scaling and translating Theorem 7.1.4, we have the following guarantee.

Theorem 7.1.5. *For any $0 < \mu \le L$ and any $0 \le \epsilon \le 1$, there is a polynomial q of degree $O(\sqrt{\frac{L}{\mu}} \log(\frac{1}{\epsilon}))$ such that*

$$\inf_{q(0)=1} \max_{\mu\le x\le L} |q(x)| \le \epsilon.$$

Proof. Theorem 7.1.4 shows that there is q such that

$$\max_{x\in[0,L]} \left| q(x) - \left(1 - \frac{x}{L}\right)^s \right| \le 2\exp(-\frac{d^2}{2s}).$$

Hence, we have that $|q(0) - 1| \le 2\exp(-\frac{d^2}{2s})$ and that

$$\max_{x\in[\mu,L]} |q(x)| \le \left(1 - \frac{\mu}{L}\right)^s + 2\exp(-\frac{d^2}{2s}).$$

To make both $|q(0) - 1| \le \frac{\epsilon}{3}$ and $\max_{x\in[\mu,L]} |q(x)| \le \frac{\epsilon}{3}$, we set $d = O(\sqrt{s} \log \frac{1}{\epsilon})$ and $s = O(\frac{L}{\mu} \log(\frac{1}{\epsilon}))$. By rescaling q, we have $q(0) = 1$. □

7.2 Conjugate Gradient

Consider any algorithm that starts with $x^{(0)} = 0$ and satisfies the invariant

$$x^{(k+1)} \in x^{(0)} + \text{span}\{\nabla f(x^{(0)}), \nabla f(x^{(1)}), \cdots, \nabla f(x^{(k)})\}.$$

For $f(x) = \frac{1}{2} x^\top A x - b^\top x$, it is easy to see that $x^{(k)} \in \mathcal{K}_k$ defined as follows:

Definition 7.2.1. Define the Krylov subspaces $\mathcal{K}_0 = \{0\}$, $\mathcal{K}_k = \text{span}\{b, Ab, \cdots, A^{k-1}b\}$.

Therefore, the best such an algorithm can do is to solve $\min_{x\in\mathcal{K}_k} f(x)$. For the quadratic function $f(x) \stackrel{\text{def}}{=} \frac{1}{2} x^\top A x - b^\top x$, one can compute $\min_{x\in\mathcal{K}_k} f(x)$ efficiently using the conjugate gradient method. Note that

$$\arg\min_{x\in\mathcal{K}_k} f(x) = \arg\min_{x\in\mathcal{K}_k} \|x - x^*\|_A^2$$

where $Ax^* = b$.

Throughout this section, we assume A is positive definite. For a given linear system $Ax = b$, we can always go to $A^T Ax = A^T b$ to satisfy the assumption.

Lemma 7.2.2. *Let $x^{(k)} = \operatorname{argmin}_{x\in\mathcal{K}_k} f(x)$ be the Krylov sequence. . Then, the steps $v^{(k)} = x^{(k)} - x^{(k-1)}$ are "conjugate", namely,*

$$v^{(i)\top} A v^{(j)} = 0 \text{ for all } i \neq j.$$

Proof. Assume that $i < j$. The optimality of $x^{(j)}$ shows that $\nabla f(x^{(j)}) \in \mathcal{K}_j^\perp \subset \mathcal{K}_{j-1}^\perp$ and that $\nabla f(x^{(j-1)}) \in \mathcal{K}_{j-1}^\perp$. Hence, we have

$$Av^{(j)} = \nabla f(x^{(j)}) - \nabla f(x^{(j-1)}) \in \mathcal{K}_{j-1}^\perp.$$

Next, we note that $v^{(i)} = x^{(i)} - x^{(i-1)} \in \mathcal{K}_i \subset \mathcal{K}_{j-1}$. Hence, we have $v^{(i)\top} A v^{(j)} = 0$. □

Since the steps are conjugate, $v^{(i)}$ forms a conjugate basis for the Krylov subspaces:

$$\mathcal{K}_k = \operatorname{span}\{v^{(1)}, v^{(2)}, \cdots, v^{(k)}\}.$$

Note that $x^{(k)} = x^{(k-1)} + v^{(k)}$. Hence, it suffices to find a formula for $v^{(k)}$.

Lemma 7.2.3. *We have*

$$v^{(k)} = \frac{v^{(k)\top} r^{(k-1)}}{\|r^{(k-1)}\|^2} \left(r^{(k-1)} - \frac{r^{(k-1)\top} A v^{(k-1)}}{v^{(k-1)} A v^{(k-1)}} v^{(k-1)} \right).$$

Proof. Again, the optimality condition shows that $\nabla f(x^{(k-1)}) \in \mathcal{K}_{k-1}^\perp$ and that $\nabla f(x^{(k-1)}) \in \mathcal{K}_k$ by the definition of $\mathcal{K}$. Hence, we have

$$\mathcal{K}_k = \operatorname{span}\{v^{(1)}, \cdots, v^{(k-1)}, r^{(k-1)}\}$$

where $r^{(k-1)} = b - Ax^{(k-1)}$. Therefore, we have

$$v^{(k)} = c_0 r^{(k-1)} + \sum_{i=1}^{k-1} c_i v^{(i)}$$

for some c_0.

For c_0, we use that $r^{(k-1)} \in \mathcal{K}_{k-1}^\perp$. This gives $v^{(k)\top} r^{(k-1)} = c_0 \|r^{(k-1)}\|^2$.

For c_{k-1}, since $v^{(i)\top} A v^{(k-1)} = 0$ for any $i \neq k-1$, we have

$$0 = v^{(k)\top} A v^{(k-1)} = c_0 r^{(k-1)\top} A v^{(k-1)} + c_{k-1} v^{(k-1)} A v^{(k-1)}$$

and $c_{k-1} = -c_0 \cdot \frac{r^{(k-1)\top} A v^{(k-1)}}{v^{(k-1)} A v^{(k-1)}}$.

For other c_i, we note that $Av^{(j)} \in \mathcal{K}_{j+1}$ and $r^{(k-1)} \in \mathcal{K}_{k-1}^\perp$. Hence, $c_i = 0$ for all $i \notin \{0, k-1\}$.

Substitute the c_i, we get

$$v^{(k)} = \frac{v^{(k)\top} r^{(k-1)}}{\|r^{(k-1)}\|^2} \left(r^{(k-1)} - \frac{r^{(k-1)\top} A v^{(k-1)}}{v^{(k-1)} A v^{(k-1)}} v^{(k-1)} \right).$$

□

To make the formula simpler, we define $p^{(k)} = \frac{\|r^{(k-1)}\|_2^2}{v^{(k)\top} r^{(k-1)}} v^{(k)}$.

Lemma 7.2.4. *We have that $x^{(k)} = x^{(k-1)} + \frac{\|r^{(k-1)}\|_2^2}{p^{(k)\top} A p^{(k)}} p^{(k)}$ and $p^{(k)} = r^{(k-1)} - \frac{\|r^{(k-1)}\|^2}{\|r^{(k-2)}\|_2^2} p^{(k-1)}$.*

Proof. For the first formula, note that

$$x^{(k)} = x^{(k-1)} + v^{(k)} = x^{(k-1)} + \frac{v^{(k)\top} r^{(k-1)}}{\|r^{(k-1)}\|_2^2} p^{(k)}.$$

For the quantity $v^{(k)\top} r^{(k-1)}$, we note that $f(x^{(k-1)} + t v^{(k)})$ is minimized at $t = 1$ and hence

$$\begin{aligned} v^{(k)\top} r^{(k-1)} &= v^{(k)\top} A v^{(k)} \\ &= \frac{v^{(k)\top} r^{(k-1)}}{\|r^{(k-1)}\|_2^2} \frac{v^{(k)\top} r^{(k-1)}}{\|r^{(k-1)}\|_2^2} p^{(k)\top} A p^{(k)} \end{aligned}$$

where we used the definition of $p^{(k)}$. Simplifying gives

$$\frac{v^{(k)\top} r^{(k-1)}}{\|r^{(k-1)}\|_2^2} = \frac{\|r^{(k-1)}\|_2^2}{p^{(k)\top} A p^{(k)}}$$

and hence the first formula.

For the second formula, we use Lemma 7.2.3 and substitute $p^{(k)} = \frac{\|r^{(k-1)}\|_2^2}{v^{(k)\top} r^{(k-1)}} v^{(k)}$. This gives

$$p^{(k)} = r^{(k-1)} - \frac{r^{(k-1)\top} A v^{(k-1)}}{\|r^{(k-2)}\|_2^2} p^{(k-1)}. \tag{7.2.1}$$

Note that

$$r^{(k-1)} = b - Ax^{(k-1)} = r^{(k-2)} - Av^{(k-1)}.$$

Taking inner product with $r^{(k-1)}$ and using $r^{(k-1)} \perp r^{(k-2)}$ gives $\|r^{(k-1)}\|^2 = r^{(k-1)\top} A v^{(k-1)}$. Put this into (7.2.1) gives the result. □

Algorithm 19: `ConjugateGradient` (CG)

$r^{(0)} = b - Ax^{(0)}$. $p^{(0)} = r^{(0)}$.
for $k = 1, 2, \cdots$ **do**
  $\alpha = \frac{\|r^{(k-1)}\|_2^2}{p^{(k)\top} A p^{(k)}}$.
  $x^{(k)} \leftarrow x^{(k-1)} + \alpha p^{(k)}$.
  $r^{(k)} \leftarrow r^{(k-1)} + \alpha A p^{(k)}$.
  if $\|r^{(k)}\|_2 \le \epsilon$ **then return** $x^{(k)}$
  $p^{(k+1)} = r^{(k)} - \frac{\|r^{(k)}\|^2}{\|r^{(k-1)}\|^2} p^{(k)}$.
end

Theorem 7.2.5. *Let $f(x) = \frac{1}{2} x^\top A x - b^\top x$ for some positive definite matrix A and some vector b. Let $x^{(k)}$ be the sequence produced by* `ConjugateGradient`*. Then, we have*

$$f(x^{(k)}) - f(x^*) \le \frac{1}{2} \inf_{q(0)=1} \max_i q(\lambda_i)^2 \cdot \|b\|_{A^{-1}}^2$$

where q searches over polynomials of degree at most k and λ_i are eigenvalues of A.

Proof. Note that $\mathcal{K}_k = \{p(A) : \deg p < k\}$. Hence, we have

$$\begin{aligned} 2(f(x^{(k)}) - f(x^*)) &= \min_{x \in \mathcal{K}_k} \|x - x^*\|_A^2 \\ &= \inf_{\deg p < k} \|(p(A) - A^{-1}) b\|_A^2 \\ &= \inf_{\deg p < k} \|A^{-\frac{1}{2}} (Ap(A) - I) A^{-\frac{1}{2}} b\|_A^2 \\ &= \inf_{\deg p < k} \|(Ap(A) - I) A^{-\frac{1}{2}} b\|_2^2 \end{aligned}$$

Let $q(x) = 1 - xp(x)$. Note that $q(0) = 1$ and $\deg q \le k$ and any such q is of the form $1 - xp$. Hence, we have

$$\begin{aligned} 2(f(x^{(k)}) - f(x^*)) &= \inf_{\deg q \le k, q(0)=1} \|q(A) A^{-\frac{1}{2}} b\|_2^2 \\ &\le \inf_{q(0)=1} \|q(A)\|_{\mathrm{op}}^2 \cdot \|b\|_{A^{-1}}^2. \end{aligned}$$

The result follows from the fact that $\|q(A)\|_{\mathrm{op}}^2 = \max_i q(\lambda_i)^2$. □

It is known that for any $0 < \mu \le L$, there is a degree k polynomial q with $q(0) = 1$ such that

$$\max_i q(\lambda_i)^2 \le 2 \left(1 - \frac{2}{\sqrt{\frac{L}{\mu}} + 1}\right)^k$$

Therefore, it takes $O(\sqrt{\frac{L}{\mu}}\log(\frac{1}{\epsilon}))$ iterations to find x such that $f(x) - f(x^*) \le \epsilon \cdot \|b\|^2_{A^{-1}}$.

Also, we note that if there are only s distinct eigenvalues in A, then conjugate gradient finds the exact solution in s iterations.

7.3 Accelerated Gradient Descent via Plane Search

According to legend, the first proof for accelerated gradient descent is due to Nemirovski in the 70s. The first proof involves a 2-dimensional plane search subroutine. Later on this was improved to line search and finally Nesterov showed how to get rid of line search in 1982. We change the proof slightly to get rid of all uses of parameters.

Algorithm 20: `NemAGD`

Input: Initial point $x^{(1)} \in \mathbb{R}^n$.
for $k = 1, 2, \cdots$ **do**
  $x^{(k)+} \leftarrow \operatorname{argmin}_{x = x^{(k)} + t\nabla f(x^{(k)})} f(x)$.
  $P^{(k)} \leftarrow x^{(1)} + \operatorname{span}(x^{(k)+} - x^{(1)}, \sum_{s=1}^k \frac{\nabla f(x^{(s)})}{\|\nabla f(x^{(s)})\|})$.
  `// Alternatively, one can use` $P^{(k)} \leftarrow x^{(k)} + \operatorname{span}(x^{(k)} - x^{(1)}, \nabla f(x^{(k)}), \sum_{s=1}^k \frac{\nabla f(x^{(s)})}{\|\nabla f(x^{(s)})\|})$ `without defining` $x^{(k)+}$.
  $x^{(k+1)} = \arg\min_{x \in P^{(k)}} f(x)$.
end

Theorem 7.3.1. *Assume that f is convex with $\nabla^2 f(x) \preceq L \cdot I$ for all x. Then, we have that*

$$f(x^{(k)}) - f(x^*) \lesssim \frac{L\|x^* - x^{(1)}\|^2}{k^2}.$$

Proof. Let $\delta_k = f(x^{(k)}) - f(x^*)$. By the convexity of f, we have

$$\begin{aligned}\delta_k &\le \nabla f(x^{(k)})^\top (x^{(k)} - x^*) \\ &= \nabla f(x^{(k)})^\top (x^{(k)} - x^{(1)}) + \nabla f(x^{(k)})^\top (x^{(1)} - x^*).\end{aligned}$$

Since $x^{(k)}$ is the minimizer on $P^{(k-1)}$ which contains $x^{(1)}$, we have $\nabla f(x^{(k)})^\top (x^{(k)} - x^{(1)}) = 0$ and hence

$$\delta_k \le \nabla f(x^{(k)})^\top (x^{(1)} - x^*).$$

Let $\lambda_t = \frac{1}{\|\nabla f(x^{(t)})\|}$. Note that

$$\sum_{k=1}^T \lambda_k \delta_k \le \left\langle \sum_{k=1}^T \frac{\nabla f(x^{(k)})}{\|\nabla f(x^{(k)})\|}, x^{(1)} - x^* \right\rangle \le \|x^* - x^{(1)}\|_2 \cdot \left\| \sum_{k=1}^T \frac{\nabla f(x^{(k)})}{\|\nabla f(x^{(k)})\|} \right\|_2.$$

Finally, we note that $\nabla f(x^{(k+1)}) \perp P^{(k)}$ and hence $\nabla f(x^{(k+1)}) \perp \sum_{s=1}^k \frac{\nabla f(x^{(s)})}{\|\nabla f(x^{(s)})\|}$. Therefore, we have

$$\left\| \sum_{k=1}^T \frac{\nabla f(x^{(k)})}{\|\nabla f(x^{(k)})\|} \right\|_2^2 = \sum_{k=1}^T \left\| \frac{\nabla f(x^{(k)})}{\|\nabla f(x^{(k)})\|} \right\|_2^2 = T.$$

Hence, we have

$$\sum_{k=1}^T \lambda_k \delta_k \le \sqrt{T} \cdot \|x^* - x^{(1)}\|_2.$$

Since $x^{(k)+} \in P$, we have that

$$\delta_{k+1} = f(x^{(k+1)}) - f(x^*) \le f(x^{(k)+}) - f(x^*) \le \delta_k - \frac{1}{2L}\|\nabla f(x^{(k)})\|_2^2.$$

Hence, we have $\|\nabla f(x^{(k)})\|_2^2 \leq 2L(\delta_k - \delta_{k+1})$. This gives

$$\sum_{k=1}^{T} \frac{\delta_k}{\sqrt{\delta_k - \delta_{k+1}}} \leq \sqrt{2LT} \cdot \|x^* - x^{(1)}\|_2$$

for all T. Solving this recursion gives the result. □

Exercise 7.3.2. Solve the recursion omitted at the end of the proof.

Note that the proof above used only the fact that $\{x^{(1)}, x^{(k)+}, \sum_{s=1}^{k} \frac{\nabla f(x^{(s)})}{\|\nabla f(x^{(s)})\|}\} \subset P$. Therefore, one can put extra vectors in P to obtain extra features. For example, if we use the subspace

$$P = x^{(k)} + \text{span}(x^{(k)} - x^{(1)}, \nabla f(x^{(k)}), \sum_{s=1}^{k} \frac{\nabla f(x^{(s)})}{\|\nabla f(x^{(s)})\|}, x^{(k)} - x^{(k-1)}),$$

then one can prove that this algorithm is equivalently to conjugate gradient when f is a quadratic function.

■ 7.4 Accelerated Gradient Descent

Gradient Mapping

To give a general version of acceleration, we consider problem of the form

$$\min_x \phi(x) \stackrel{\text{def}}{=} f(x) + h(x)$$

where $f(x)$ is strongly convex and smooth and $h(x)$ is convex. We assume that we access the function f and h differently via:

1. Let $\mathcal{T}_f$ be the cost of computing $\nabla f(x)$.
2. Let $\mathcal{T}_{h,\lambda}$ be the cost of minimizing $h(x) + \frac{\lambda}{2}\|x - c\|_2^2$ exactly.

The idea is to move whatever we can optimize in ϕ to h and hopefully this makes the remaining part of ϕ, f, as smooth and strongly convex as possible. To make the statement general, we only assume h is convex and hence h may not be differentiable. To handle this issue, we need to define an approximate derivative of h that we can compute.

Definition 7.4.1. We define the gradient step

$$p_x = \text{argmin}_y f(x) + \nabla f(x)^\top (y - x) + \frac{L}{2}\|y - x\|^2 + h(y)$$

and the gradient mapping

$$g_x = L(x - p_x).$$

Note that if $h = 0$, then $p_x = x - \frac{1}{L}\nabla f(x)$ and $g_x = \nabla f(x)$. In general, if $\phi \in \mathcal{C}^2$, then we have that

$$p_x = x - \frac{1}{L}\nabla \phi(x) + O(\frac{1}{L^2}).$$

Therefore, we have that $g_x = \nabla \phi(x) + O(\frac{1}{L})$. Hence, the gradient mapping is an approximation of the gradient of ϕ that is computable in time $\mathcal{T}_g + \mathcal{T}_{h,L}$.

The key lemma we use here is that ϕ satisfies a lower bound defining using g_x. Ideally, we would love to get a lower bound as follows:

$$\phi(z) \geq \phi(x) + g_x^\top (z - x) + \frac{\mu}{2}\|z - x\|_2^2.$$

But it is WRONG. If that was true for all z, then we would have $g_x = \nabla\phi(x)$. However, if $\phi \in \mathcal{C}^2$ is μ strongly convex, then we have

$$\begin{aligned}\phi(z) &\geq \phi(x) + \nabla\phi(x)^\top(z-x) + \frac{\mu}{2}\|z-x\|_2^2 \\ &\geq \phi(x - \frac{1}{L}\nabla\phi(x)) + \frac{1}{2L}\|\nabla\phi(x)\|^2 + \nabla\phi(x)^\top(z-x) + \frac{\mu}{2}\|z-x\|_2^2. \end{aligned} \tag{7.4.1}$$

It turns out that this is true and is exactly what we need for proving gradient descent, mirror descent and accelerated gradient descent.

Theorem 7.4.2. *Given $\phi = f + h$. Suppose that f is μ strongly convex with L-Lipschitz gradient. Then, for any z, we have that*

$$\phi(z) \geq \phi(p_x) + g_x^\top(z-x) + \frac{1}{2L}\|g_x\|_2^2 + \frac{\mu}{2}\|z-x\|_2^2.$$

Proof. Let $\overline{f}(y) = f(x) + \nabla f(x)^\top(y-x) + \frac{L}{2}\|y-x\|_2^2$ and $p_t = p_x + t(z-p_x)$. Using that p_0 is the minimizer of $\overline{f} + h$, we have that

$$\overline{f}(p_0) + h(p_0) \leq \overline{f}(p_t) + h(p_t) \leq \overline{f}(p_0) + \nabla\overline{f}(p_0)^\top(p_t - p_0) + \frac{L}{2}\|p_t - p_0\|^2 + h(p_t).$$

Hence, we have that

$$\begin{aligned}0 &\leq \nabla\overline{f}(p_0)^\top(p_t - p_0) + \frac{L}{2}\|p_t - p_0\|^2 + h(p_t) - h(p_0) \\ &\leq t\cdot\left(\nabla\overline{f}(p_0)^\top(z-p_0) + h(z) - h(p_0)\right) + \frac{Lt^2}{2}\|z-p_0\|^2.\end{aligned}$$

Taking $t \to 0^+$, we have

$$\nabla\overline{f}(p_x)^\top(z-p_x) + h(z) - h(p_x) \geq 0$$

Expanding the term $\nabla\overline{f}(p_x)$, we have

$$\nabla f(x)^\top(z-p_x) + L(p_x-x)^\top(z-p_x) + h(z) - h(p_x) \geq 0.$$

Equivalently,

$$\begin{aligned}h(z) &\geq h(p_x) + \nabla f(x)^\top(p_x - z) + L(p_x-x)^\top(p_x-z) \\ &= h(p_x) + \nabla f(x)^\top(p_x - z) + \frac{1}{L}\|g_x\|^2 + L(p_x-x)^\top(x-z) \\ &= h(p_x) + \nabla f(x)^\top(p_x - z) + \frac{1}{L}\|g_x\|^2 + g_x^\top(z-x).\end{aligned}$$

Using that

$$f(z) \geq f(x) + \nabla f(x)^\top(z-x) + \frac{\mu}{2}\|z-x\|^2$$

and that

$$f(p_x) \leq f(x) + \nabla f(x)^\top(p_x - x) + \frac{1}{2L}\|g_x\|^2,$$

we have the result:

$$\begin{aligned}\phi(z) &\geq h(p_x) + f(x) + \nabla f(x)^\top(p_x - x) + \frac{1}{L}\|g_x\|^2 + g_x^\top(z-x) + \frac{\mu}{2}\|z-x\|^2 \\ &\geq \phi(p_x) + \frac{1}{2L}\|g_x\|^2 + g_x^\top(z-x) + \frac{\mu}{2}\|z-x\|^2.\end{aligned}$$

□

The next lemma shows that $\|g_x\|_2 \leq 2G$ if ϕ is G-Lipschitz.

Lemma 7.4.3. *If ϕ is G-Lipschitz, then $\|g_x\|_2 \leq 2G$ for all x.*

Proof. By the definition of gradient mapping (namely, p_x is the minimizer of a function), we have that

$$f(x) + \nabla f(x)^\top (p_x - x) + \frac{L}{2} \|p_x - x\|^2 + h(p_x) \le f(x) + h(x).$$

Using $h(p_x) \ge h(x) + \nabla h(x)^\top (p_x - x)$, we have that

$$0 \ge \nabla \phi(x)^\top (p_x - x) + \frac{L}{2} \|p_x - x\|^2 \ge -G \|p_x - x\|_2 + \frac{L}{2} \|p_x - x\|^2 .$$

Hence, we have that $\|p_x - x\|_2 \le \frac{2}{L} G$ and hence $\|g_x\|_2 \le 2G$. □

Gradient Descent Using Gradient Mapping

Putting $z = x$ in Theorem 7.4.2, we have the following:

Lemma 7.4.4 (Gradient Descent Lemma)**.** *We have that*

$$\phi(p_x) \le \phi(x) - \frac{1}{2L} \|g_x\|_2^2 .$$

This shows that each step of the gradient step decreases the function value by $\frac{1}{2L} \|g_x\|_2^2$. Therefore, if the gradient is large, then we decrease the function value by a lot. On the other hand, Putting $z = x^*$ for Theorem 7.4.2 shows that

$$\phi(x^*) \ge \phi(p_x) + g_x^\top (x^* - x).$$

If the gradient is small and domain is bounded, this shows that we are close to the optimal. Combining these two facts, we can get the gradient descent.

Mirror Descent Using Gradient Mapping

Consider the mirror descent as

$$x^{(k+1)} = x^{(k)} - \eta g_{x^{(k)}}. \tag{7.4.2}$$

The main difference between this and gradient descent is that we will take a larger step size η. To analyze the mirror descent, we use Theorem 7.4.2 and the convexity of ϕ to get that

$$\phi(\frac{1}{k} \sum_{i=1}^{k} p_{x^{(i)}}) - \phi(x^*) \le \frac{1}{k} \sum_{i=1}^{k} (\phi(p_{x^{(i)}}) - \phi(x^*)) \le \frac{1}{k} \sum_{i=1}^{k} g_{x^{(i)}}^\top (x^{(i)} - x^*). \tag{7.4.3}$$

Therefore, it suffices to upper bound $g_{x^{(i)}}^\top (x^{(i)} - x^*)$. The following lemma shows that if $g_{x^{(i)}}^\top (x^{(i)} - x^*)$ is large, then either the gradient is large or the distance to optimum moves a lot. It turns out this holds for any vector g, not necessarily an approximate gradient.

Lemma 7.4.5 (Mirror Descent Lemma)**.** *Let $p = x - \eta g$. Then, we have that*

$$g^\top (x - u) = \frac{\eta}{2} \|g\|_2^2 + \frac{1}{2\eta} \left(\|x - u\|_2^2 - \|p - u\|_2^2 \right)$$

for any u.

Algorithm and Analysis

Recall Lemma 7.4.4 shows that if the gradient is large, gradient descent makes a large progress. On the other hand, if the gradient is small, (7.4.3) shows that mirror descent makes a large progress. Therefore, it is natural to combine

two approaches.

Algorithm 21: AGD

Input: Initial point $x^{(1)} \in \mathbb{R}^n$.
$y^{(1)} \leftarrow x^{(1)}$, $z^{(1)} \leftarrow x^{(1)}$.
for $k = 1, 2, \cdots, T$ **do**
  $x^{(k+1)} \leftarrow \tau z^{(k)} + (1-\tau) y^{(k)}$.
  Perform a gradient step: $y^{(k+1)} = x^{(k+1)} - \frac{1}{L} g_{x^{(k+1)}}$.
  Perform a mirror step: $z^{(k+1)} = z^{(k)} - \eta g_{x^{(k+1)}}$.
end
Return $\frac{1}{T}\sum_{k=1}^{T} p_{x^{(k+1)}}$.

Note that if $\tau = 1$, the algorithm is simply mirror descent and if $\tau = 0$, the algorithm is gradient descent.

Theorem 7.4.6. *Setting $\frac{1-\tau}{\tau} = \eta L$ and $\eta = \frac{1}{\sqrt{\mu L}}$, we have that*

$$\phi\left(\frac{1}{T}\sum_{k=1}^{T} p_{x^{(k+1)}}\right) - \phi(x^*) \le \frac{2}{T}\sqrt{\frac{L}{\mu}}\left(\phi(x^{(1)}) - \phi(x^*)\right).$$

In particular, if we restart the algorithm every $4\sqrt{\frac{L}{\mu}}$ iterations, we can find x such that

$$\phi(x) - \phi(x^*) \le 2\left(1 - \sqrt{\frac{\mu}{L}}\right)^{\Omega(T)}\left(\phi(x^{(1)}) - \phi(x^*)\right)$$

in T steps. Furthermore, each step takes $\mathcal{T}_f + \mathcal{T}_{h,L}$

Proof. Lemma 7.4.5 showed that

$$g_{x^{(k+1)}}^{\top}(z^{(k)} - x^*) \le \frac{\eta}{2}\left\|g_{x^{(k+1)}}\right\|_2^2 + \frac{1}{2\eta}\left(\left\|z^{(k)} - x^*\right\|_2^2 - \left\|z^{(k+1)} - x^*\right\|_2^2\right) \tag{7.4.4}$$

This shows that if the mirror descent has large error $g_{x^{(k+1)}}^{\top}(z^{(k)} - x^*)$, then the gradient descent makes a large progress ($\frac{\eta}{2}\left\|g_{x^{(k+1)}}\right\|_2^2$).

To make the left-hand side usable, note that $x^{(k+1)} = z^{(k)} + \frac{1-\tau}{\tau}\cdot(y^{(k)} - x^{(k+1)})$ and hence

$$\begin{aligned}
g_{x^{(k+1)}}^{\top}(x^{(k+1)} - x^*) &= g_{x^{(k+1)}}^{\top}(z^{(k)} - x^*) + \frac{1-\tau}{\tau}\cdot g_{x^{(k+1)}}^{\top}(y^{(k)} - x^{(k+1)}) \\
&\le g_{x^{(k+1)}}^{\top}(z^{(k)} - x^*) + \frac{1-\tau}{\tau}(\phi(y^{(k)}) - \phi(y^{(k+1)}) - \frac{1}{2L}\left\|g_{x^{(k+1)}}\right\|_2^2) \\
&\le \frac{\eta}{2}\left\|g_{x^{(k+1)}}\right\|_2^2 + \frac{1}{2\eta}\left(\left\|z^{(k)} - x^*\right\|_2^2 - \left\|z^{(k+1)} - x^*\right\|_2^2\right) + \frac{1-\tau}{\tau}(\phi(y^{(k)}) - \phi(y^{(k+1)}) - \frac{1}{2L}\left\|g_{x^{(k+1)}}\right\|_2^2).
\end{aligned}$$

where we used Theorem 7.4.2 in the middle and (7.4.4) at the end.

Now, we set $\frac{1-\tau}{\tau} = \eta L$ and get

$$g_{x^{(k+1)}}^{\top}(x^{(k+1)} - x^*) \le \eta L(\phi(y^{(k)}) - \phi(y^{(k+1)})) + \frac{1}{2\eta}\left(\left\|z^{(k)} - x^*\right\|_2^2 - \left\|z^{(k+1)} - x^*\right\|_2^2\right).$$

Taking a sum on both side and let $\overline{x} = \frac{1}{T}\sum_{k=1}^{T} p_{x^{(k+1)}}$, we have that

$$\begin{aligned}
\phi(\overline{x}) - \phi(x^*) &\leq \frac{1}{T}\sum_{k=1}^{T}(\phi(p_{x^{(k+1)}}) - \phi(x^*)) \\
&\leq \frac{1}{T}\sum_{k=1}^{T} g_{x^{(k+1)}}^{\top}(x^{(k+1)} - x^*) \\
&\leq \frac{\eta L}{T}\left(\phi(y^{(1)}) - \phi(y^{(T+1)})\right) + \frac{1}{2\eta T}\left\|z^{(1)} - x^*\right\|_2^2 \\
&\leq \left(\frac{\eta L}{T} + \frac{1}{2\eta T\mu}\right)\left(\phi(x^{(1)}) - \phi(x^*)\right)
\end{aligned}$$

The conclusion follows from our setting of η. □

■ 7.5 Accelerated Coordinate Descent

Recall from Theorem 6.4.2 that if $\frac{\partial^2}{\partial x_i^2}\ell(x) \leq L_i$ for all x and that ℓ is μ strongly convex, then we can find $x^{(k)}$ in k coordinate steps such that

$$\mathbf{E}\ell(x^{(k)}) - \ell(x^*) \leq (1 - \frac{\mu}{L})^{\Omega(k)}(\ell(x^{(0)}) - f(x^*))$$

where $L = \sum_i L_i$. Now, the really fun part is here. Consider the function

$$\phi(x) = f(x) + h(x) \quad \text{with} \quad f(x) = \frac{\mu}{2}\|x\|_2^2 \text{ and } h(x) = \ell(x) - \frac{\mu}{2}\|x\|_2^2.$$

Since f is $\mu + \frac{L}{n}$ smooth (YES!, I know this is also μ smooth) and μ strongly convex and since h is convex, we apply Theorem 7.4.6 and get an algorithm that takes $O^*(\sqrt{\frac{L}{n\mu}})$ steps. Note that each step involves $\mathcal{T}_f + \mathcal{T}_{h,\mu+\frac{L}{n}}$. Obviously, $\mathcal{T}_f = 0$. Next, note that $\mathcal{T}_{h,\mu+\frac{L}{n}}$ involves solving a problem of the form

$$\begin{aligned}
y_x &= \text{argmin}_y(\frac{\mu}{2} + \frac{L}{2n})\|y - x\|^2 + (\ell(y) - \frac{\mu}{2}\|x\|^2) \\
&= \text{argmin}_y \ell(y) - \mu y^{\top}x + \frac{L}{2n}\|y - x\|^2.
\end{aligned}$$

Now, we can apply Theorem 6.4.2 to solve this problem. It takes

$$O^*(\frac{L + (L/n)\cdot n}{L/n}) = O^*(n) \text{ coordinate steps.}$$

Therefore, in total it takes

$$O^*(\sqrt{\frac{L}{n\mu}})\cdot O^*(n) = O^*(\sqrt{\frac{Ln}{\mu}}) \text{ coordinate steps.}$$

Hence, we have the following theorem

Theorem 7.5.1 (Accelerated Coordinate Descent Convergence)**.** *Given an μ strongly-convex function ℓ. Suppose that $\frac{\partial^2}{\partial x_i^2}\ell(x) \leq L_i$ for all x and let $L = \sum L_i$. We can minimize ℓ in $O^*(\sqrt{\frac{Ln}{\mu}})$* coordinate steps.

Remark 7.5.2. It is known how to do it in $O^*(\sum_i \sqrt{\frac{L_i}{\mu}})$ steps [3].

■ 7.6 Accelerated Stochastic Descent

Recall from Theorem 6.3.7 that if $\nabla^2\ell_i(x) \leq L$ for all x and i and that $\ell(x) = \frac{1}{n}\sum_{i=1}^{n}\ell_i(x)$ is μ strongly convex, then we can find x in $O((n + \frac{L}{\mu})\log(\frac{1}{\epsilon}))$ stochastic steps such that

$$\mathbf{E}\ell(x) - \ell(x^*) \leq \epsilon(\ell(x^{(0)}) - \ell(x^*)).$$

Similar to the coordinate descent, we can accelerate it using the accelerated gradient descent (Theorem 7.4.6). To apply Theorem 7.4.6, we consider the function

$$\phi(x) = f(x) + h(x) \quad \text{with} \quad f(x) = \frac{\mu}{2}\|x\|_2^2 \text{ and } h(x) = \ell(x) - \frac{\mu}{2}\|x\|_2^2.$$

Since f is $\mu + \frac{L}{n}$ smooth (YES!, I know this is also μ smooth) and μ strongly convex and since h is convex, we apply Theorem 7.4.6 and get an algorithm that takes $O^*(1+\sqrt{\frac{L}{n\mu}})$ steps. Note that each step involves $\mathcal{T}_f + \mathcal{T}_{h,\mu+\frac{L}{n}}$. Obviously, $\mathcal{T}_f = 0$. Next, note that $\mathcal{T}_{h,\mu+\frac{L}{n}}$ involves solving a problem of the form

$$\begin{aligned} y_x &= \operatorname{argmin}_y (\frac{\mu}{2} + \frac{L}{2n})\|y-x\|^2 + (\ell(y) - \frac{\mu}{2}\|x\|^2) \\ &= \operatorname{argmin}_y \ell(y) - \mu y^\top x + \frac{L}{2n}\|y-x\|^2 \\ &= \operatorname{argmin}_y \frac{1}{n}\sum_i (\ell_i(y) + \frac{L}{2n}\|y-x\|^2 - \mu y^\top x) \end{aligned}$$

Now, we can apply Theorem 6.3.7 to solve this problem. It takes

$$O^*(n + \frac{L+\frac{L}{n}}{\mu+\frac{L}{n}}) = O^*(n)$$

Therefore, in total it takes

$$O^*(1+\sqrt{\frac{L}{n\mu}}) \cdot O^*(n) = O^*(n + \sqrt{n\frac{L}{\mu}}).$$

Theorem 7.6.1. *Given a convex function $\ell = \frac{1}{n}\sum \ell_i$. Suppose that $\nabla^2 \ell_i(x) \le L$ for all i and x and that ℓ is μ strongly convex. Suppose we can compute $\nabla \ell_i$ in $O(1)$ time. We have an algorithm that outputs an x such that*

$$\mathbf{E}\ell(x) - \ell(x^*) \le \varepsilon(\ell(x^{(0)}) - \ell(x^*))$$

in $O^*(n + \sqrt{n\frac{L}{\mu}})$ stochastic steps.

Part II

Sampling

Chapter 8

Introduction and Gradient Methods

8.1 Gradient-based methods: Langevin Dynamics

Here we study a simple stochastic process for generating samples from a desired distribution $e^{-f(x)}$. It can also be viewed as a stochastic version of gradient descent. While gradient descent corresponds to an ordinary differential equation (ODE), stochastic gradient descent corresponds to a stochastic differential equation (SDE).

Algorithm 22: `LangevinDynamics` (LD)

Input: Initial point $x_0 \in \mathbb{R}^n$.
Solve the stochastic differential equation

$$dx_t = -\nabla f(x_t)dt + \sqrt{2}dW_t$$

Output: x_t.

Here $f : \mathbb{R}^n \to \mathbb{R}$ is a function, x_t is the random variable at time t and dW_t is infinitesimal Brownian motion also known as a Wiener process. While it takes some care to define rigorously, for now we can view it as the following discrete process

$$x_{t+h} = x_t - h\nabla f(x_t) + \sqrt{2h}\zeta_t$$

with ζ_t sampled independently from $N(0, I)$. When we take the step size $h \to 0$, this discrete process converges to the continuous one. We choose to discuss the continuous version here only for simplicity.

A crucial difference between ordinary differentials and stochastic differentials is in the chain rule. Unlike classical differentials where we have $df(x) = \nabla f(x)dx$, we have the following chain rule for stochastic calculus.

Lemma 8.1.1 (Itô's lemma)**.** *For any process $x_t \in \mathbb{R}^n$ satisfying $dx_t = \mu(x_t)dt + \sigma(x_t)dW_t$ where $\mu(x_t) \in \mathbb{R}^n$ and $\sigma(x_t) \in \mathbb{R}^{n\times m}$, we have that*

$$df(x_t) = \nabla f(x_t)^\top \mu(x_t)dt + \nabla f(x_t)^\top \sigma(x_t)dW_t + \frac{1}{2}\mathrm{tr}(\sigma(x_t)^\top \nabla^2 f(x_t)\sigma(x_t))dt.$$

First, we will see that the Langevin SDE converges to e^{-f} in continuous time. The proof relies on the following general theorem about the distribution induced by an SDE.

Theorem 8.1.2 (Fokker–Planck equation*)**.** *For any process $x_t \in \mathbb{R}^n$ satisfying $dx_t = \mu(x_t)dt + \sigma(x_t)dW_t$ where $\mu(x_t) \in \mathbb{R}^n$ and $\sigma(x_t) \in \mathbb{R}^{n\times m}$ with the initial point x_0 drawn from p_0. Then, the distribution p_t of x_t satisfies the equation*

$$\frac{dp_t}{dt} = -\sum_i \frac{\partial}{\partial x_i}(\mu(x)_i p_t(x)) + \frac{1}{2}\sum_{i,j}\frac{\partial^2}{\partial x_i \partial x_j}\left[(D(x))_{ij}p_t(x)\right]$$

where $D(x) = \sigma(x)\sigma(x)^\top$.

Proof. For any smooth function ϕ, we have that

$$\mathbf{E}_{x\sim p_t}\phi(x) = \mathbf{E}\phi(x_t).$$

Taking derivatives on the both sides with respect to t and using Itô's lemma (Lemma 8.1.1), we have that

$$\begin{aligned}\int \phi(x) dp_t(x) dx &= \mathbf{E}\left(\nabla\phi(x_t)^\top \mu(x_t) dt + \nabla\phi(x_t)^\top \sigma(x_t) dW_t + \frac{1}{2}\mathrm{tr}(\sigma(x_t)^\top \nabla^2\phi(x_t)\sigma(x_t)) dt\right) \\ &= \mathbf{E}\left(\nabla\phi(x_t)^\top \mu(x_t) dt + \frac{1}{2}\mathrm{tr}(\nabla^2\phi(x_t) D(x_t)) dt\right).\end{aligned}$$

Using $x_t \sim p_t$, we have that

$$\int \phi(x)\frac{dp_t}{dt} dx = \int \nabla\phi(x)^\top \mu(x) p_t(x) + \frac{1}{2}\mathrm{tr}(\nabla^2\phi(x) D(x)) p_t(x) dx.$$

Integrating by parts,

$$\int \nabla\phi(x)^\top \mu(x) p_t(x) dx = -\int \phi(x) \sum_i \frac{\partial}{\partial x_i}(\mu_i(x) p_t(x)) dx.$$

Similarly, integrating by parts twice gives

$$\begin{aligned}\int \mathrm{tr}(\sigma(x)^\top \nabla^2\phi(x)\sigma(x)) p_t(x) dx &= \int \mathrm{tr}(\nabla^2\phi(x)\sigma(x)\sigma(x)^\top) p_t(x) dx \\ &= \sum_{i,j} \int \phi(x) \frac{\partial^2}{\partial x_i \partial x_j}\left[(D(x))_{ij} p_t(x)\right] dx.\end{aligned}$$

Hence,

$$\int \phi(x)\left[\frac{dp_t}{dt} + \sum_i \frac{\partial}{\partial x_i}(\mu(x)_i p_t(x)) - \frac{1}{2}\sum_{i,j}\frac{\partial^2}{\partial x_i \partial x_j}\left[(D(x))_{ij} p_t(x)\right]\right] dx = 0$$

for any smooth ϕ. Therefore, we have the conclusion of the lemma.

We apply the Fokker–Planck equation to the Langevin dynamics. □

Theorem 8.1.3. *For any smooth function f, the density proportional to $F = e^{-f}$ is stationary for the Langevin dynamics.*

Proof. The Fokker–Planck equation (Theorem 8.1.2) shows that the distribution p_t of x_t satisfies

$$\frac{dp_t}{dt} = \sum_i \frac{\partial}{\partial x_i}\left(\frac{\partial f(x)}{\partial x_i} p_t(x)\right) + \sum_i \frac{\partial^2}{\partial x_i^2}\left[p_t(x)\right]. \tag{8.1.1}$$

We can verify that $p_t(x) \propto e^{-f(x)}$ is a solution. □

Convergence via Coupling. Next we turn to the rate of convergence, which will also prove uniqueness. For this, we assume that f is strongly convex. The proof is via the classical coupling technique (cite: Aldous). Briefly, two distributions are compared via some probabilistic distance. For example, the total variation distance can be bounded by the probability that random variables from the two distributions are not identical, under any matching (assignment) between the two distributions. We then couple two copies x_t, y_t of the process with different starting points (the coupling is a joint distribution $D(x_t, y_t)$ with the property that its marginal for each of x_t, y_t is exactly the process) and show that their distributions get closer over time. While the challenge is usually to find a good coupling, in the present case, the simplest identity coupling works (for the continuous time convergence).

Lemma 8.1.4. *Let x_t, y_t evolves according to the Langevin diffusion for a μ-strongly convex function $f : \mathbb{R}^n \to \mathbb{R}$. Then, there is a coupling between x_t, y_t s.t.*

$$\mathbf{E}\left\|x_t - y_t\right\| \le e^{-\mu t}\left\|x_0 - y_0\right\|.$$

Proof. From the definition of LD, and by using the same Gaussian dW_t for both process, we have that

$$\frac{d}{dt}(x_t - y_t) = \nabla f(y_t) - \nabla f(x_t).$$

Hence,

$$\frac{1}{2}\frac{d}{dt}\|x_t - y_t\|^2 = 2\left\langle \nabla f(y_t) - \nabla f(x_t), x_t - y_t \right\rangle.$$

Next, from the strong convexity of f, we have

$$f(y_t) - f(x_t) \geq \nabla f(x_t)^\top (y_t - x_t) + \frac{\mu}{2}\|x_t - y_t\|^2,$$
$$f(x_t) - f(y_t) \geq \nabla f(y_t)^\top (x_t - y_t) + \frac{\mu}{2}\|x_t - y_t\|^2.$$

Adding two equations together, we have

$$(\nabla f(x_t) - \nabla f(y_t))^\top (x_t - y_t) \geq \mu \|x_t - y_t\|^2.$$

Therefore,

$$\frac{1}{2}\frac{d}{dt}\|x_t - y_t\|^2 \leq -\mu\|x_t - y_t\|^2.$$

Hence,

$$\frac{d}{dt}\log\|x_t - y_t\|^2 = \frac{\frac{d}{dt}\|x_t - y_t\|^2}{\|x_t - y_t\|^2} \leq -2\mu.$$

Integrating both sides from 0 to t, we get

$$\log\|x_t - y_t\|^2 \leq \log\|x_0 - y_0\|^2 - 2\mu t$$

owhich proves the result. □

■ 8.2 Langevin Dynamics is Gradient Descent in Density Space*[1]

Here we show that Langevin dynamics is simply gradient descent for the function $F(\rho) = D_{\mathrm{KL}}(\rho\|\nu)$ on the Wasserstein space where $\nu = e^{-f(x)}/\int e^{-f(y)}dy$. For this, we first define the Wasserstein space.

Definition 8.2.1. The Wasserstein space $P_2(\mathbb{R}^n)$ on $\mathbb{R}^n$ is the manifold on the set of probability measures on $\mathbb{R}^n$ such that the shortest path distance of two measures x, y in this manifold is exactly equal to the Wasserstein distance between x and y.

Lemma 8.2.2. *For any $p \in P_2(\mathbb{R}^n)$ and $v \in T_pP_2(\mathbb{R}^n)$, we can write $v(x) = \nabla\cdot(p(x)\nabla\lambda(x))$ for some function λ on $\mathbb{R}^n$. Furthermore, the length of v in this metric is given by*

$$\|v\|_p^2 = \mathbf{E}_{x\sim p}\|\nabla\lambda(x)\|^2.$$

Proof. Let $p \in P_2(\mathbb{R}^n)$ and $v \in T_pP_2(\mathbb{R}^n)$. We will show that any change of density v can be represented by a vector field c on $\mathbb{R}^n$ as follows: Consider the process $x_0 \sim p$ and $\frac{d}{dt}x_t = c(x_t)$. Let p_t be the density of the distribution of x_t. To compute $\frac{d}{dt}p_t$, we follow the same idea as in the proof as Theorem 8.1.2. For any smooth function ϕ, we have that $\mathbf{E}_{x\sim p_t}\phi(x) = \mathbf{E}\phi(x_t)$. Taking derivatives on the both sides with respect to t, we have that

$$\int \phi(x)\frac{d}{dt}p_t(x)dx = \int \nabla\phi(x)^\top c(x)p_t(x)dx = -\int \nabla\cdot(c(x)p_t(x))\phi(x)dx$$

where we used integration by parts at the end. Since this holds for all ϕ, we have that

$$\frac{dp_t(x)}{dt} = -\nabla\cdot(p_t(x)c(x)).$$

Since we are interested only inthe vector fields that generate minimum movement in Wasserstein distance, we consider the optimization problem

$$\min_{-\nabla\cdot(pc)=v}\frac{1}{2}\int p(x)\|c(x)\|^2dx$$

[1] Anything marked with * means it is probably too mathematical and could be skipped.

where we can think v is the change of p_t. Let $\lambda(x)$ be the Lagrangian multiplier of the constraint $-\nabla \cdot (pc) = v$. Then, the problem becomes

$$\begin{aligned}&\min_c \frac{1}{2}\int p(x)\|c(x)\|^2 dx - \int \lambda(x)\nabla\cdot(p(x)c(x))dx.\\ =&\min_c \frac{1}{2}\int p(x)\|c(x)\|^2 dx + \int \nabla\lambda(x)^\top c(x)\cdot p(x)dx.\end{aligned}$$

Now, we note that the problem is a pointwise optimization problem with the minimizer is given by

$$c(x) = -\nabla\lambda(x).$$

This proves that any vector fields that generate minimum movement in Wasserstein distance is a gradient field. Also, we have that $v(x) = \nabla\cdot(p(x)\nabla\lambda(x))$. Note that the right hand side is an elliptical differential equation and hence for any v with $\int v(x)dx = 0$, there is an unique solution $\lambda(x)$. Therefore, we can write $v(x) = \nabla\cdot(p(x)\nabla\lambda(x))$ for some $\lambda(x)$.

Next, we note that the movement is given by

$$\|v\|_p^2 = \int p(x)\|c(x)\|^2 dx = \mathbf{E}_{x\sim p}\|\nabla\lambda(x)\|^2.$$

□

As we discussed in the gradient descent section, one can use norms other than ℓ_2 norm. For the Wasserstein space, we should use the local norm.

Theorem 8.2.3. *Let ρ_t be the density of the distribution produced by Langevin Dynamics for the target distribution $\nu = e^{-f(x)}/\int e^{-f(y)}dy$. Then, we have that*

$$\frac{d\rho}{dt} = \mathrm{argmin}_{v\in T_p P_2(\mathbb{R}^n)} \langle \nabla F(\rho), v\rangle_p + \frac{1}{2}\|v\|_p^2.$$

Namely, ρ_t follows continuous gradient descent in the density space for the function $F(\rho) = D_{\mathrm{KL}}(\rho\|\nu)$ under the Wasserstein metric.

Proof. For any function c, the optimization problem of interest satisfies

$$\min_{\delta=\nabla\cdot(\rho\nabla\lambda)} \langle c,\delta\rangle + \frac{1}{2}\int\rho(x)\|\nabla\lambda(x)\|^2dx = \min_{\nabla\lambda} -\int\rho(x)\cdot\nabla c(x)^\top\nabla\lambda(x)dx + \frac{1}{2}\int\rho(x)\|\nabla\lambda(x)\|^2dx.$$

Solving the right hand side, we have $\nabla c = \nabla\lambda$ and hence $\delta = \nabla\cdot(\rho\nabla c)$. Now, we note that $\nabla F(\rho) = \log\frac{\rho}{\nu} - 1$. Therefore,

$$\begin{aligned}\frac{d\rho}{dt} &= \nabla\cdot(\rho\nabla(\log\frac{\rho}{\nu} - 1))\\ &= \nabla\cdot(\rho\nabla\log\frac{\rho}{\nu})\\ &= \nabla\cdot(\rho\nabla f) + \Delta\rho\end{aligned}$$

which is exactly equal to (8.1.1). □

To analyze this continuous descent in Wasserstein space, we first prove that continuous gradient descent converges exponentially whenever F is strongly convex.

Lemma 8.2.4. *Let F be a function satisfying "Gradient Dominance":*

$$\|\nabla F(x)\|_x^2 \geq \alpha\cdot(F(x) - \min_y F(y)) \quad \textit{for all } x \tag{8.2.1}$$

on the manifold with the metric $\|\cdot\|_x$ where ∇ is the gradient on the manifold. Then, the process $dx_t = -\nabla F(x_t)dt$ converges exponentially, i.e., $F(x_t) - \min_y F(y) \leq e^{-\alpha t}(F(x_0) - \min_y F(y))$.

Proof. We write

$$\frac{d}{dt}(F(x) - \min_y F(y)) = \langle \nabla F(x_t), \frac{dx_t}{dt} \rangle_{x_t} = -\|\nabla F(x_t)\|_{x_t}^2 \leq -\alpha(F(x) - \min_y F(y)).$$

The conclusion follows. □

Finally, we note that the log-Sobolev inequality for the density ν can be re-stated as the condition (8.2.1).

Lemma 8.2.5. *Fix a density ν. Then the log-Sobolev inequality, namely, for every smooth function g,*

$$2\int \|\nabla g\|^2 \, d\nu \geq \alpha \int g(x)^2 \log g(x)^2 \, d\nu$$

implies the condition (8.2.1).

Proof. Take $g(x) = \sqrt{\frac{\rho(x)}{\nu(x)}}$, the log-Sobolev inequality shows that

$$\frac{1}{2}\int \rho(x) \left\| \nabla \log \frac{\rho(x)}{\nu(x)} \right\|^2 dx \geq \alpha \cdot \int \rho(x) \log \frac{\rho(x)}{\nu(x)} \, dx \text{ for all } \rho.$$

As we calculate in Theorem 8.2.3, we have that

$$\|\nabla F(\rho)\|_\rho^2 = \int \rho(x) \left\| \nabla \log \frac{\rho(x)}{\nu(x)} \right\|^2 dx.$$

Therefore, this is exactly the condition (8.2.1) with coefficient 2α. □

Combining Lemma 8.2.5 and Lemma 8.2.4, we have the following result:

Theorem 8.2.6. *Let f be a smooth function with log-Sobolev constant α. Then the Langevin dynamics*

$$dx_t = -\nabla f(x)dt + \sqrt{2}dW_t$$

converges exponentially in KL-divergence to the density $\nu(x) \propto e^{-f(x)}$ with mixing rate $O(\frac{1}{\alpha})$, i.e., $KL(x_t, \nu) \leq e^{-2\alpha t} KL(x_0, \nu)$.

See [40] for a tight estimate of log-Sobolev constant for logconcave measures. In particular for a logconcave measure with support of diameter D, the log-Sobolev constant is $\Omega(1/D)$.

Discussion

Langevin dynamics samples from strongly logconcave densities, and the convergence is fast in continuous time. Turning this into an efficient algorithm takes more work and assumptions. As we saw in Section 8.2, it turns out that Langevin dynamics is in fact gradient descent in the space of probability measures under the Wasserstein metric, where the function being minimized is the KL-divergence of the current density from the target stationary density. For more on this view, see [63].

■ 8.3 Cutting Plane method for Volume Computation

The cutting plane method gives a simple algorithm for computing the volume of a bounded set, or the integral of a function. The algorithm relies crucially on the ability to compute the center of gravity of the original set/function

restricted by halfspaces. For simplicity we describe it here for the case when the input object is a convex body.

Algorithm 23: `CuttingPlaneVolume`

Input: A body $K \subset \mathbb{R}^n$, $r \in R$, and an oracle for computing centroid.
Fix an ordering on the axes: $e_1, e_2, \dots e_n$
$K^{(0)} = K$, $z^{(0)} = \text{centroid}(K^{(0)}), V = 1, k = 0$.
for $k = 0, \cdots$ **do**
  Let e_i be an axis vector so that the width of $K^{(k)}$ along e_i is greater than r.
  if *There is no such* e_i **then Break**.
  Let a be e_i or $-e_i$ with sign chosen so that $H^{(k)} \stackrel{\text{def}}{=} \left\{x : a^\top x \le a^\top z^{(k)}\right\}$ contains $z^{(0)}$.
  $K^{(k+1)} \leftarrow K^{(k)} \cap H^{(k)}$.
  $z^{(k+1)} \leftarrow \text{centroid}(K^{(k+1)})$.
  $\widehat{z} \leftarrow \text{centroid}(K^{(k)} \setminus K^{(k+1)})$.
  $V \leftarrow V \cdot \frac{\left\|\widehat{z} - z^{(k+1)}\right\|}{\left\|\widehat{z} - z^{(k)}\right\|}$.
end
Return $V \cdot \prod_{i=1}^n w_i(K^{(k+1)})$ where $w_i(\cdot)$ is the width along e_i.

The idea behind the algorithm is simple: when we cut a set with a hyperplane through its centroid, the line joining the centroids of the two sides passes through the original centroid; moreover, the ratio of the two segments is exactly the ratio of the volumes of the two halfspaces.

Lemma 8.3.1. *For any measurable bounded set S in $\mathbb{R}^n$ and any halfspace H with bounding hyperplane containing the centroid of S, we have*

$$\textit{centroid}(S) = \frac{\text{vol}(S \cap H)}{\text{vol}(S)} \textit{centroid}(S \cap H) + \frac{\textit{vol}(S \cap \overline{H})}{\text{vol}(S)} \textit{centroid}(S \cap \overline{H}).$$

The following lemma has a proof similar to that of the Grunbaum theorem.

Lemma 8.3.2. *Let $K \subseteq \mathbb{R}^n$ be a convex body with centroid at the origin. Suppose that for some unit vector θ, the support of K along θ is $[a, b]$. Then,*

$$|a| \ge \frac{b}{n}.$$

Exercise 8.3.3. Prove Lemma 8.3.2. [Hint: Use Theorem 1.6.5.]

Using the above property, we can show that the algorithm reaches a cuboid in a small number of iterations.

Theorem 8.3.4. *Let K be a convex body in $\mathbb{R}^n$ containing a cube of side length r around its centroid and contained in a cube of side length R. Algorithm* `CuttingPlaneVolume` *correctly computes the volume of K using $O(n \log \frac{nR}{r})$ centroid computations.*

Proof. By Lemma **??**, at each iteration, the volume of the remaining set $K^{(k)}$ decreases by a factor of at most $(1 - \frac{1}{e})$. When the directional width along any axis is less than $r/2$ (namely $\max_{x \in K} \left|e_i^\top x\right| \le \frac{r}{2}$), the algorithm stops cutting along that axis. Thus, by Lemma 8.3.2, the directional width along every axis is at least $r/2(n+1)$. Since the set always contains the origin, and the original set contains a cube of side length r, when the algorithm stops, it must be an axis-parallel cuboid. So the final volume is at least $(r/2n)^n$. The initial volume is at most R^n. Therefore the number of iterations is at most

$$\log_{(1-\frac{1}{e})} \left(\frac{R^n}{(r/2(n+1))^n} \right) = O\left(n \log(nR/r)\right).$$

In each iteration, by Lemma 8.3.1, the algorithm maintains the ratio of the volume of the original K to the current $K^{(k)}$. □

The above algorithm shows that computing the volume is polytime reducible to computing the centroid. Since volume is known to be #P-hard for explicit polytopes, this means that centroid computation is also #P-hard for polytopes [54]. In later chapters we will see randomized polytime algorithms for sampling and hence for approximating centroid and volume.

Chapter 9

Geometrization and Sampling

In this chapter, we study some polynomial-time sampling algorithms.

Ball Walk

The simplest continuous algorithm for exploring space is Brownian motion with the ODE $dx_t = dW_t$. To turn this into a sampling algorithm, for a convex function f, we saw an extension using the gradient of f, namely

$$dx_t = -\nabla f(x_t) + \sqrt{2}dW_t$$

which can be used to sample according to the density proportional to e^{-f}. In this chapter we will begin with an even simpler method, which does not need access to the gradient or assume differentiability, only an oracle that can evaluate F.

Algorithm 24: `BallWalk`

Input: Step-size δ, number of steps T, starting point x_0 in the support of target density Q.
Repeat T times: at a point x,

1. Pick a random point y in the δ-ball centered at x.
2. Go to y with probability $\min\left\{1, \frac{Q(y)}{Q(x)}\right\}$.

return x.

Exercise 9.0.1. Show that the distribution with density Q is *stationary* for the ball walk in a connected, compact full-dimensional set, i.e., if the distribution of the current point x has density Q, it remains Q.

Under mild conditions, the distribution of the current point approaches the target density Q. The main question is the rate of convergence, which would allow us to bound the number of steps. Note that each step involves only a function evaluation, to an oracle that outputs the value of a function proportional to the desired density. To bound the rate of convergence (and as a result the uniqueness of the stationary distribution), we first develop some general tools.

9.1 Basics of Markov chains

For more detailed reading, including additional properties, see Section 1 of [47].

A Markov chain is defined using a σ-algebra $(K, \mathcal{A})$, where K is the state space and $\mathcal{A}$ is a set of subsets of K that is closed under complements and countable unions. For each element u of K, we have a probability measure P_u on $(K, \mathcal{A})$, i.e., each set $A \in \mathcal{A}$ has a probability $P_u(A)$. Informally, P_u is the distribution obtained upon taking one step from u. The triple $(K, \mathcal{A}, \{P_u : u \in K\})$ along with a starting distribution Q_0 defines a Markov chain, i.e., a sequence of elements of K, $w_0, w_1, \ldots$, where w_0 is chosen from Q_0 and each subsequent w_i is chosen from $P_{w_{i-1}}$. The choice of w_{i+1} depends only on w_i and is independent of $w_0, \ldots, w_{i-1}$.

A distribution Q on $(K, \mathcal{A})$ is called *stationary* if taking one step from it maintains the distribution, i.e., for any $A \in \mathcal{A}$,

$$\int_K P_u(A)\, dQ(u) = Q(A).$$

A distribution Q is *atom-free* if there is no $x \in K$ with $Q(x) > 0$.

Example. For the ball walk in a convex body, the state space K is the convex body, and A is the set of all measurable subsets of K. The next step distribution is

$$P_u(\{u\}) = 1 - \frac{\text{vol}\,(K \cap (u + \delta B_n))}{\text{vol}(\delta B_n)}$$

and for any measurable subset A,

$$P_u(A) = \frac{\text{vol}\,(A \cap (u + \delta B_n))}{\text{vol}(\delta B_n)} + 1_{u \in A}(u) P_u(\{u\})$$

The uniform distribution is stationary, i.e., $Q(A) = \frac{\text{vol}(A)}{\text{vol}(K)}$.

The *ergodic flow* of a subset A w.r.t. the distribution Q is defined as

$$\Phi(A) = \int_A P_u(K \setminus A)\, dQ(u).$$

A distribution Q is stationary if and only if $\Phi(A) = \Phi(K \setminus A)$. The existence and uniqueness of the stationary distribution Q for general Markov chains is a subject on its own. One way to ensure uniqueness of a stationary distribution is to use *lazy* Markov chains. In a lazy version of a given Markov chain, at each step, with probability $1/2$, we do nothing; with the rest we take a step according to the Markov chain. The next theorem is folklore.

Theorem 9.1.1. *If Q is stationary w.r.t. a lazy Markov chain then it is the unique stationary distribution for that Markov chain.*

Informally, the mixing rate of a random walk is the number of steps required to reduce some measure of the distance of the current distribution to the stationary distribution by a constant factor. The following notions will be useful for comparing two distributions P, Q.

1. Total variation distance is $d_{tv}(P, Q) = \sup_{A \in \mathcal{A}} |P(A) - Q(A)|$.
2. L_2 distance of P with respect to Q is

$$d_2(P, Q) = \int_K \frac{dP(u)}{dQ(u)}\, dP(u) = \int_K \left(\frac{dP(u)}{dQ(u)}\right)^2 dQ(u).$$

3. P is said to be M-warm w.r.t. Q if $M = \sup_{A \in \mathcal{A}} \frac{P(A)}{Q(A)}$.

Convergence via Conductance

Now we introduce an important tool to bound the rate of convergence of Q_t,the distribution after t steps to Q. Assume that Q is the unique stationary distribution. The *conductance* of a subset A is defined as

$$\phi(A) = \frac{\Phi(A)}{\min\{Q(A), Q(K \setminus A)\}}$$

and the conductance of the Markov chain is

$$\phi = \min_A \phi(A) = \min_{0 < Q(A) \le \frac{1}{2}} \frac{\int_A P_u(K \setminus A)\, dQ(u)}{Q(A)}.$$

The *local* conductance of an element u is $\ell(u) = 1 - P_u(\{u\})$.

For any $0 \leq s < \frac{1}{2}$, the s-conductance of a Markov chain is defined as

$$\phi_s = \min_{A: s < Q(A) \leq \frac{1}{2}} \frac{\phi(A)}{Q(A) - s}.$$

Ideally we would like to show that $d(Q_t, Q)$, the distance between the distribution after t steps and the target Q is monotonically (and rapidly) decreasing. While this is not true in general for the total variation distance, it is true for a slight extension. We consider

$$\sup_{A: Q(A) = x} Q_t(A) - Q(A)$$

for each $x \in [0, 1]$. To prove inductively that this quantity decreases , Let $\mathcal{G}_x$ be the set of functions defined as

$$\mathcal{G}_x = \left\{ g : K \to [0, 1] \ : \ \int_{u \in K} g(u) \ dQ(u) = x \right\}.$$

Using this, define

$$h_t(x) = \sup_{g \in \mathcal{G}_x} \int_{u \in K} g(u) \ (dQ_t(u) - dQ(u)) = \sup_{g \in \mathcal{G}_x} \int_{u \in K} g(u) \ dQ_t(u) - x.$$

This function is strongly concave.

Exercise 9.1.2. Show that the function h_t is concave, and if Q is atom-free , then $h_t(x) = \sup_{A: Q(A) = x} Q_t(A) - Q(A)$ and the supremum is achieved by some subset.

Lemma 9.1.3. *Let Q be atom-free and $t \geq 1$. For any $0 \leq x \leq 1$, let $y = \min\{x, 1 - x\}$. Then,*

$$h_t(x) \leq \frac{1}{2} h_{t-1}(x - 2\phi y) + \frac{1}{2} h_{t-1}(x + 2\phi y).$$

Proof. Assume that $0 \leq x \leq \frac{1}{2}$. We construct two functions, g_1 and g_2, and use these to bound $h_t(x)$. Let A be a subset that achieves $h_t(x)$. Define

$$g_1(u) = \begin{cases} 2P_u(A) - 1 & \text{if } u \in A, \\ 0 & \text{if } u \notin A, \end{cases} \quad \text{and} \quad g_2(u) = \begin{cases} 1 & \text{if } u \in A, \\ 2P_u(A) & \text{if } u \notin A. \end{cases}$$

Note that $\frac{1}{2}(g_1 + g_2)(u) = P_u(A)$ for all $u \in K$, which means that

$$\frac{1}{2} \int_{u \in K} g_1(u) \ dQ_{t-1}(u) + \frac{1}{2} \int_{u \in K} g_2(u) \, dQ_{t-1}(u) = \int_{u \in K} P_u(A) \, dQ_{t-1}(u) = Q_t(A).$$

Since the walk is lazy, $P_u(A) \geq \frac{1}{2}$ iff $u \in A$, the range of the functions g_1, g_2 is $[0, 1]$. We let

$$x_1 = \int_{u \in K} g_1(u) \ dQ(u) \quad \text{and} \quad x_2 = \int_{u \in K} g_2(u) \ dQ(u),$$

then $g_1 \in \mathcal{G}_{x_1}$ and $g_2 \in \mathcal{G}_{x_2}$. Moreover,

$$\frac{1}{2}(x_1 + x_2) = \frac{1}{2} \int_{u \in K} g_1(u) \ dQ(u) + \frac{1}{2} \int_{u \in K} g_2(u) \ dQ(u) = \int_{u \in K} P_u(A) \ dQ(u) = Q(A) = x.$$

since Q is stationary.

$$\begin{aligned} h_t(x) &= Q_t(A) - Q(A) \\ &= \frac{1}{2} \int_{u \in K} g_1(u) \ dQ_{t-1}(u) + \frac{1}{2} \int_{u \in K} g_2(u) \ dQ_{t-1}(u) - Q(A) \\ &= \frac{1}{2} \int_{u \in K} g_1(u) \ (dQ_{t-1}(u) - dQ(u)) + \frac{1}{2} \int_{u \in K} g_2(u) \ (dQ_{t-1}(u) - dQ(u)) \\ &\leq \frac{1}{2} h_{t-1}(x_1) + \frac{1}{2} h_{t-1}(x_2). \end{aligned}$$

Next,

$$\begin{aligned} x_1 &= \int_{u\in K} g_1(u)\, dQ(u) \\ &= 2\int_{u\in A} P_u(A)\, dQ(u) - \int_{u\in A} dQ(u) \\ &= 2\int_{u\in A} (1 - P_u(K\setminus A))\, dQ(u) - x \\ &= x - 2\int_{u\in A} P_u(K\setminus A)\, dQ(u) \\ &= x - 2\Phi(A) \\ &\le x - 2\phi x \\ &= x(1-2\phi). \end{aligned}$$

Thus we have, $x_1 \le x(1-2\phi) \le x \le x(1+2\phi) \le x_2$. Since h_{t-1} is concave, the chord from x_1 to x_2 on h_{t-1} lies below the chord from $[x(1-2\phi), x(1+2\phi)]$. Therefore,

$$h_t(x) \le \frac{1}{2}h_{t-1}(x(1-2\phi)) + \frac{1}{2}h_{t-1}(x(1+2\phi)).$$

□

A proof along the same lines implies the following generalization.

Lemma 9.1.4. *Let Q be atom-free and $0 \le s \le 1$. For any $s \le x \le 1-s$, let $y = \min\{x-s, 1-x-s\}$. Then for any integer $t > 0$,*

$$h_t(x) \le \frac{1}{2}h_{t-1}(x - 2\phi_s y) + \frac{1}{2}h_{t-1}(x + 2\phi_s y).$$

These results can be extended to the case when Q has atoms with slightly weaker bounds [47].

Theorem 9.1.5. *Let $0 \le s \le 1$ and C_0 and C_1 be such that*

$$h_0(x) \le C_0 + C_1 \min\{\sqrt{x-s}, \sqrt{1-x-s}\}.$$

Then

$$h_t(x) \le C_0 + C_1 \min\{\sqrt{x-s}, \sqrt{1-x-s}\}\left(1 - \frac{\phi_s^2}{2}\right)^t.$$

The proof is by induction on t.

Corollary 9.1.6. *We have*

1. *Let $M = \sup_A Q_0(A)/Q(A)$. Then,*

$$d_{TV}(Q_t, Q) \le \sqrt{M}\left(1 - \frac{\phi^2}{2}\right)^t.$$

2. *Let $0 < s \le \frac{1}{2}$ and $H_s = \sup\{|Q_0(A) - Q(A)| : Q(A) \le s\}$. Then,*

$$d_{TV}(Q_t, Q) \le H_s + \frac{H_s}{s}\left(1 - \frac{\phi_s^2}{2}\right)^t.$$

3. *Let $M = d_2(Q_0, Q)$. Then for any $\varepsilon > 0$,*

$$d_{TV}(Q_t, Q) \le \varepsilon + \sqrt{\frac{M}{\varepsilon}}\left(1 - \frac{\phi^2}{2}\right)^t.$$

Convergence via Log-Sobolev

For a warm start, the convergence rate established by conductance is asymptotically optimal in many cases of interest, including the ball walk for convex body. However, when the starting distribution is more focused, e.g., a single point, then there is a significant starting penalty usually a factor of the dimension or larger. One way to avoid this is to observe that the conductance of smaller subsets is in fact even higher and that one does not need to pay this large starting penalty. A classical technique in this regard is the log-Sobolev constant. For a Markov chain with stationary density Q and transition operator P, we can define it as follows.

$$\rho = \inf_{g:\text{smooth},\ \int g(x)^2 dQ(x) = 1} \frac{\int_{x,y\in K} (f(x) - f(y))^2 P(x,y) dQ(x)}{\int f(x)^2 \log f(x)^2 dQ(x)}.$$

This parameter allows us to show convergence of the current distribution to the target in relative entropy. Recall that the relative entropy of a distribution P with respect to a distribution Q is

$$H_Q(P) = \int_K P(x) \log \frac{P(x)}{Q(x)} dQ(x).$$

Theorem 9.1.7. *For a Markov chain with distribution Q_t at time t, and log-Sobolev parameter ρ, we have*

$$H_Q(Q_t) \le e^{-2\rho t} H_Q(Q_0).$$

■ 9.2 Conductance of the Ball Walk

In the section we bound the conductance of the ball walk when applied to the indicator function of a convex body. At first glance, the ball walk is not an efficient algorithm, even for uniformly sampling a convex body. The reason is simply that the local conductance could be exponentially small (consider a point close to the vertex of a polyhedron). We can get around this in two ways. The first, which is simpler, but less efficient is to "smoothen" the convex body by taking the Minkowski sum with a small Euclidean ball, i.e., replace K with $K + \alpha B^n$.

Exercise 9.2.1. Let K be a convex body in $\mathbb{R}^n$ containing the unit ball. Show that (a) $\text{vol}(K + \alpha B^n) \le (1+\alpha)^n K$ and (b) with $\delta = \alpha/\sqrt{n}$ the local conductance of every point in $K + \alpha B^n$ is at least an absolute constant.

Using the exercise, it suffices to set $\alpha = 1/n$, so that a sample from $K + \alpha B^n$ has a large probability of being in K and then $\delta = 1/n^{3/2}$.

The second approach is to show that the local conductance is in fact large almost everywhere, and if the starting distribution is "warm" then these points can effectively be ignored. This will allow us to make δ much larger, namely $\delta = 1/\sqrt{n}$. Larger step sizes should allows us to prove faster mixing.

To convey the main ideas of the analysis, we focus on the first approach here. The goal is to show that the conductance of any subset is large, i.e., the probability of crossing over in one step is at least proportional to the measure of the set or its complement, whichever is smaller. First, we argue that the one-step distributions of two points will have a significant overlap if the points are sufficient close.

Lemma 9.2.2 (One-step overlap)**.** *Let $u, v \in K$ s.t. $\ell(u), \ell(v) \ge \ell$ and $\|u - v\| \le \frac{t\delta}{\sqrt{n}}$. Then the one-step distributions from them satisfy $d_{TV}(P_u, P_v) \le 1 + t - \ell$.*

Setting $t = \ell/2$, this says that if the total variation distance between the one-step distributions from u, v is greater than $1 - \ell/2$, then the distance between them is at least $\frac{\ell\delta}{2\sqrt{n}}$. What this effectively says is that points close to the internal boundary of a subset are likely to cross over to the other side. To complete a proof we would need to show that the internal boundary of any subset is large if the subset (or its complement) is large, a purely geometric property.

Theorem 9.2.3 (Isoperimetry)**.** *Let S_1, S_2, S_3 be a partition of a convex body K of diameter D. Then,*

$$\text{vol}(S_3) \ge \frac{2}{D} d(S_1, S_2) \min\left\{\text{vol}(S_1), \text{vol}(S_2)\right\}.$$

This can be generalized to any logconcave measure. We will discuss this and other extensions in detail later. But first we bound the conductance.

Theorem 9.2.4. *Let K be a convex body in $\mathbb{R}^n$ of diameter D containing the unit ball and with every $u \in K$ having $\ell(u) \geq \ell$. Then the conductance of the ball walk on K with step size δ is*

$$\Omega\left(\frac{\ell^2\delta}{\sqrt{n}D}\right).$$

Proof. Let $K = S_1 \cup S_2$ be a partition into measurable sets. We will prove that

$$\int_{S_1} P_x(S_2)\,dx \quad \geq \quad \frac{\ell^2\delta}{16\sqrt{n}D}\min\{\mathrm{vol}(S_1), \mathrm{vol}(S_2)\} \tag{9.2.1}$$

Note that since the uniform distribution is stationary,

$$\int_{S_1} P_x(S_2)\,dx = \int_{S_2} P_x(S_1)\,dx.$$

Consider the points that are "deep" inside these sets, i.e., unlikely to jump out of the set:

$$S_1' = \left\{x \in S_1 : P_x(S_2) < \frac{\ell}{4}\right\} \text{ and } S_2' = \left\{x \in S_2 : P_x(S_1) < \frac{\ell}{4}\right\}.$$

Let S_3' be the rest i.e., $S_3' = K \setminus S_1' \setminus S_2'$.

Suppose $\mathrm{vol}(S_1') < \mathrm{vol}(S_1)/2$. Then

$$\int_{S_1} P_x(S_2)\,dx \geq \frac{\ell}{4}\mathrm{vol}(S_1 \setminus S_1') \geq \frac{\ell}{8}\mathrm{vol}(S_1)$$

which proves (9.2.1).

So we can assume that $\mathrm{vol}(S_1') \geq \mathrm{vol}(S_1)/2$ and similarly $\mathrm{vol}(S_2') \geq \mathrm{vol}(S_2)/2$. Now, for any $u \in S_1'$ and $v \in S_2'$,

$$||P_u - P_v||_{tv} \geq 1 - P_u(S_2) - P_v(S_1) > 1 - \frac{\ell}{2}.$$

Applying Lemma 9.2.2 with $t = \ell/2$, we get that

$$|u - v| \geq \frac{\ell\delta}{2\sqrt{n}}.$$

Thus $d(S_1, S_2) \geq \ell\delta/2\sqrt{n}$. Applying Theorem 9.2.3 to the partition S_1', S_2', S_3', we have

$$\begin{aligned}\mathrm{vol}(S_3') &\geq \frac{\ell\delta}{\sqrt{n}D}\min\{\mathrm{vol}(S_1'), \mathrm{vol}(S_2')\} \\ &\geq \frac{\ell\delta}{2\sqrt{n}D}\min\{\mathrm{vol}(S_1), \mathrm{vol}(S_2)\}.\end{aligned}$$

We can now prove (9.2.1) as follows:

$$\begin{aligned}\int_{S_1} P_x(S_2)\,dx &= \frac{1}{2}\int_{S_1} P_x(S_2)\,dx + \frac{1}{2}\int_{S_2} P_x(S_1)\,dx \\ &\geq \frac{1}{2}\mathrm{vol}(S_3')\frac{\ell}{4} \\ &\geq \frac{\ell^2\delta}{16\sqrt{n}D}\min\{\mathrm{vol}(S_1), \mathrm{vol}(S_2)\}.\end{aligned}$$

□

Corollary 9.2.5. *The ball walk in a convex body with local conductance at least ℓ everywhere has mixing rate $O(nD^2\delta^2/\ell^4)$.*

Using the construction above of adding a small ball to every point of K gives a lower bound of $\delta = 1/n^{3/2}$ and $\ell = \Omega(1)$ and thus a polynomial bound of $O(n^4D^2)$ on the mixing time.

Warm Start and s-Conductance

A more careful analysis, using a warm start and only bounding the conductance of large sets gives a better (and tight) bound for the mixing of the ball walk.

Theorem 9.2.6. *From a warm start, the ball walk in a convex body of diameter D containing a unit ball has a mixing rate of $O(n^2D^2)$ steps.*

This is based on two ideas: (1) most points of a convex body containing a unit ball have large local conductance and we can use $\delta = 1/\sqrt{n}$ instead of $1/n^{3/2}$, (2) the s-conductance is large and hence the walk mixes from a suitably warm start.

Lemma 9.2.7. *Let K be a convex body containing a unit ball. For the ball walk with δ step size, let $K_\delta = \left\{u \in K : \ell(u) \geq \frac{3}{4}\right\}$. Then K_δ is a convex set and* $\mathrm{vol}(K_\delta) \geq (1 - 2\delta\sqrt{n})\mathrm{vol}(K)$.

Exercise 9.2.8. Prove the first part of the previous lemma.

Theorem 9.2.9. *The s-conductance of the ball walk with $\delta = \frac{s}{4\sqrt{n}}$ step size in a convex body K of diameter D and containing the unit ball satisfies*

$$\phi_s \gtrsim \frac{s}{nD}.$$

The mixing rate follows by applying Theorem 9.1.5 and Corollary 9.1.6.

Tightness of the bound

The mixing rate of $O(n^2D^2)$ for the ball walk is in fact the best possible even from a warm start (with the assumption of a unit ball inside the convex body and diameter D). To see this, consider a cylinder whose cross-section is a unit ball and axis is $[0, D]$ along e_1. Suppose the starting distribution is uniform in the part of the cylinder in $[0, D/3]$. Then we claim that to reach the opposite third of the cylinder needs $\Omega(n^2D^2)$ steps with high probability. Each step has length at most δ in a random direction, and this is about $\delta/\sqrt{n}$ along e_1. Viewing this as an unbiased random walk along e_1, the effective diameter is $D/(\delta/\sqrt{n})$ and hence the number of steps to cross an interval of length $D/3$ is $\Omega(nD^2/\delta^2) = \Omega(n^2D^2)$.

Exercise 9.2.10. Prove the above claim rigorously.

Speedy walk

In the above analysis of the ball walk, the dependence on the error parameter ε, the distance to the target distribution, is polynomial in $1/\varepsilon$ rather than its logarithm. The speedy walk is a way to improve the analysis. In the speedy walk, at a point x, we sample uniformly from the intersection of $(x + \delta B^n) \cap K$ and go there. The resulting Markov chain is the subsequence of *proper* steps of the ball walk.

Theorem 9.2.11. *The conductance of the speedy walk is $\Omega(1/nD)$.*

To analyze the ball walk, we then need to show that the number of "wasted" steps is not too many. This follows from the assumption of a wam start and Lemma 9.2.7.

9.3 Generating a warm start

To get the mixing rate of $O(n^2D^2)$, we need a warm start, i.e., a distribution whose density at any point is within $O(1)$ of the target density.

Algorithm 25: `Warm Start`

Input: membership oracle for K s.t. $B^n \subseteq K \subseteq DB^n$.
Let x be a random point in B^n. Define $K_i = 2^{i/n}B^n \cap K$.
for $i = 1, \cdots, n \log D$ **do**
1. Use ball walk from x to generate random point y in K_i.
2. Set $x = y$.

end
return x.

Since $K_{i+1} \subseteq 2^{1/n}K_i$, we have $\text{vol}(K_{i+1}) \leq 2\text{vol}(K_i)$ and hence a 2-warm start is maintained. Once we have a random point from K, subsequent random points can be generated by simply continuing the ball walk; thus the cost of the warm start is only for the first sample.

9.4 Isotropic Transformation

The complexity of sampling with the ball walk is polynomial in n, D and $\log(1/\varepsilon)$ to get within ε of the target density. This is not a polynomial algorithm since the dependence is on D and not $\log D$. To get a polynomial algorithm, we need one more ingredient.

We say that a distribution Q is *isotropic* if $\mathbf{E}_Q(x) = 0$ and $\mathbf{E}_Q(xx^T) = I$, i.e., the mean is zero and the covariance matrix (exists and) is the identity. We say that the distribution is C-isotropic if the eigenvalues of its covariance matrix are in $[\frac{1}{C}, C]$. An affine transformation is said to be an *isotropic tranformation* if the resulting distribution is isotropic.

Any distribution with bounded second moments has an isotropic transformation. It is clear that satisfying the first condition is merely a translation, so assume the mean is zero. For the second, suppose the covariance matrix is $\mathbf{E}_Q(xx^T) = A$. Then consider $y = A^{-1/2}x$. It is easy to see that

$$\mathbf{E}(yy^T) = A^{-1/2}\mathbf{E}\left(xx^T\right)A^{-1/2} = I.$$

For convex bodies, isotropic position comes with a strong guarantee.

Theorem 9.4.1. *For a convex body in isotropic position (i.e., the uniform distribution over the body is isotropic), we have*

$$\sqrt{\frac{n+2}{n}}B^n \subseteq K \subseteq \sqrt{n(n+2)}B^n.$$

Thus the effective diameter is $O(n)$. If we could place a convex body in isotropic position before sampling, we would have a poly(n) algorithm. In fact, it is even better than this as most points are within distance $O(\sqrt{n})$ of the center of gravity. We quote a theorem due to Paouris.

Theorem 9.4.2. *For an isotropic logconcave density p in $\mathbb{R}^n$ and any $t \geq 1$,*

$$\Pr_p\left(\|x\| \geq ct\sqrt{n}\right) \leq e^{-t\sqrt{n}}.$$

How to compute an isotropic transformation? This is easy, from the definition, all we need is to estimate its covariance matrix, which can be done from random samples. Thus, if we could sample K, we can compute an

isotropic transformation for it. This appears cyclic – we need isotropy for efficient sampling and efficient sampling for isotropy. The solution is simply to bootstrap them.

Algorithm 26: `IsotropicTransform`

Input: membership oracle for K s.t. $B^n \subseteq K \subseteq DB^n$.
Let x be a random point in B^n, $A = I$ and $K_i = 2^{i/n}B^n \cap K$.
for $i = 1, \cdots, n \log D$ **do**

1. Use the ball walk from x to generate N random points $x_1 \dots x_N$ in AK_i.
2. Compute $C = \frac{1}{N}\sum_{i=1}^{N} x_i x_i^T$ and set $A = C^{-1/2}A$.
3. Set $x = x_N$.

end
return x.

We will choose N large enough so that after the transformation K_i is 2-isotropic and therefore K_{i+1} is 6-isotropic. We can bound N as follows.

Exercise 9.4.3. Show that if K is isotropic, then with $N = O(n^2)$, the matrix $A = \frac{1}{N}\sum_{i=1}^{N} x_i x_i^T$ for N random samples from K satisfies $\|A - I\|_{\text{op}} \leq 0.5$.

A tight bound on the sample complexity was established by [1].

Theorem 9.4.4. *For an isotropic logconcave distribution Q in $\mathbb{R}^n$, the covariance $N = O(n)$ random samples satisfies $\|A - I\|_{\text{op}} \leq 0.5$.*

Thus the overall algorithm needs $O(n \log D)$ phases, with $O(n)$ samples in each phase from a near-isotropic distribution, and thus poly(n) steps per sample.

9.5 Isoperimetry via localization

Theorem 9.2.3 was refined by KLS [29] as follows (we state it here for logconcave densities).

Theorem 9.5.1. *For any partition S_1, S_2, S_3 of $\mathbb{R}^n$, and any logconcave measure μ in $\mathbb{R}^n$,*

$$\mu(S_3) \geq \frac{\ln 2}{\mathbf{E}_\mu(\|x - \bar{x}\|)} \min\{\mu(S_1), \mu(S_2)\}.$$

Thus, for a (near-)isotropic distribution, the diameter can be replaced by $O(\sqrt{n})$ and this gives a bound of $O(n^3)$ from a warm start. One way to summarize the analysis so far is that the complexity of sampling a convex body (and in fact a logconcave density) from a warm start is $O^*(n^2/\psi^2)$ where ψ is the isoperimetric ratio of the convex body. In other words, the expansion of the Markov chain reduces to the expansion of the target logconcave density. It then becomes a natural question to find the best possible estimate for the isoperimetric ratio. KLS also provided a conjecture for this.

Conjecture 9.5.2. *The isoperimetric ratio of any isotropic logconcave density in $\mathbb{R}^n$ is $\Omega(1)$.*

The bound of the conjecture holds for all halfspace induced subsets. So the conjecture says that the worst isoperimetry is achieved up to a constant factor by a halfspace (this version does not need isotropic position). Here we discuss a powerful technique for proving such inequalities.

Classical proofs of isoperimetry for special distributions are based on different types of symmetrization that effectively identify the extremal subsets. Bounding the Cheeger constant for general convex bodies and logconcave densities is more complicated since the extremal sets can be nonlinear and hard to describe precisely, due to the trade-off between minimizing the boundary measure of a subset and utilizing as much of the "external" boundary as possible. The main technique to prove bounds in the general setting has been *localization*, a method to reduce inequalities in high dimension to inequalities in one dimension. We now describe this technique with a few applications.

Localization

We will sketch a proof of the following theorem to illustrate the use of localization. This theorem was also proved by Karzanov and Khachiyan [31] using a different, more direct approach.

Theorem 9.5.3 ([19, 44, 31]). *Let f be a logconcave function whose support has diameter D and let π_f be the induced measure. Then for any partition of $\mathbb{R}^n$ into measurable sets S_1, S_2, S_3,*

$$\pi_f(S_3) \geq \frac{2d(S_1, S_2)}{D} \min\{\pi_f(S_1), \pi_f(S_2)\}.$$

Before discussing the proof, we note that there is a variant of this result in the Riemannian setting.

Theorem 9.5.4 ([42]). *If $K \subset (M, g)$ is a locally convex bounded domain with smooth boundary, diameter D and $Ric_g \geq 0$, then the Poincaré constant is at least $\frac{\pi^2}{4D^2}$, i.e., for any g with $\int g = 0$, we have that*

$$\int |\nabla g(x)|^2 \, dx \geq \frac{\pi^2}{4D^2} \int g(x)^2 dx.$$

For the case of convex bodies in $\mathbb{R}^n$, this result is equivalent to Theorem 9.5.3 up to a constant. One benefit of localization is that it does not require a carefully crafted potential. Localization has recently been generalized to Riemannian setting [35]. The origins of this method were in a paper by Payne and Weinberger [53].

We begin the proof of Theorem 9.5.3. For a proof by contradiction, let us assume the converse of its conclusion, i.e., for some partition S_1, S_2, S_3 of $\mathbb{R}^n$ and logconcave density f, assume that

$$\int_{S_3} f(x)\, dx < C \int_{S_1} f(x)\, dx \quad \text{and} \quad \int_{S_3} f(x)\, dx < C \int_{S_2} f(x)\, dx$$

where $C = 2d(S_1, S_2)/D$. This can be reformulated as

$$\int_{\mathbb{R}^n} g(x)\, dx > 0 \quad \text{and} \quad \int_{\mathbb{R}^n} h(x)\, dx > 0 \tag{9.5.1}$$

where

$$g(x) = \begin{cases} Cf(x) & \text{if } x \in S_1, \\ 0 & \text{if } x \in S_2, \\ -f(x) & \text{if } x \in S_3. \end{cases} \quad \text{and} \quad h(x) = \begin{cases} 0 & \text{if } x \in S_1, \\ Cf(x) & \text{if } x \in S_2, \\ -f(x) & \text{if } x \in S_3. \end{cases}$$

These inequalities are for functions in $\mathbb{R}^n$. The next lemma will help us analyze them.

Lemma 9.5.5 (Localization Lemma [27]). *Let $g, h : \mathbb{R}^n \to \mathbb{R}$ be lower semi-continuous integrable functions such that*

$$\int_{\mathbb{R}^n} g(x)\, dx > 0 \quad \textit{and} \quad \int_{\mathbb{R}^n} h(x)\, dx > 0.$$

Then there exist two points $a, b \in \mathbb{R}^n$ and an affine function $\ell : [0, 1] \to \mathbb{R}_+$ such that

$$\int_0^1 \ell(t)^{n-1} g((1-t)a + tb)\, dt > 0 \quad \textit{and} \quad \int_0^1 \ell(t)^{n-1} h((1-t)a + tb)\, dt > 0.$$

The points a, b represent an interval and one may think of $\ell(t)^{n-1}$ as proportional to the cross-sectional area of an infinitesimal cone. The lemma says that over this cone truncated at a and b, the integrals of g and h are positive. Also, without loss of generality, we can assume that a, b are in the union of the supports of g and h.

Proof outline. The main idea is the following. Let H be any halfspace such that

$$\int_H g(x)\, dx = \frac{1}{2} \int_{\mathbb{R}^n} g(x)\, dx.$$

Let us call this a bisecting halfspace. Now either

$$\int_H h(x)\,dx > 0 \quad \text{or} \quad \int_{\mathbb{R}^n \setminus H} h(x)\,dx > 0.$$

Thus, either H or its complementary halfspace will have positive integrals for both g and h, reducing the domain of the integrals from $\mathbb{R}^n$ to a halfspace. If we could repeat this, we might hope to reduce the dimensionality of the domain. For any $(n-2)$-dimensional affine subspace L, there is a bisecting halfspace containing L in its bounding hyperplane. To see this, let H be a halfspace containing L in its boundary. Rotating H about L we get a family of halfspaces with the same property. This family includes H', the complementary halfspace of H. The function $\int_H g - \int_{\mathbb{R}^n \setminus H} g$ switches sign from H to H'. Since this is a continuous family, there must be a halfspace for which the function is zero.

If we take all $(n-2)$-dimensional affine subspaces defined by $\{x \in \mathbb{R}^n : x_i = r_1,\, x_j = r_2\}$ where r_1, r_2 are rational, then the intersection of all the corresponding bisecting halfspaces is a line or a point (by choosing only rational values for x_i, we are considering a countable intersection). To see why it is a line or a point, assume we are left with a two or higher dimensional set. Since the intersection is convex, there is a point in its interior with at least two coordinates that are rational, say $x_1 = r_1$ and $x_2 = r_2$. But then there is a bisecting halfspace H that contains the affine subspace given by $x_1 = r_1$, $x_2 = r_2$ in its boundary, and so it properly partitions the current set.

Thus the limit of this bisection process is a function supported on an interval (which could be a single point), and since the function itself is a limit of convex sets (intersections of halfspaces) containing this interval, it is a limit of a sequence of concave functions and is itself concave, with positive integrals. Simplifying further from concave to linear takes quite a bit of work. For the full proof, we refer the reader to [45]. □

Going back to the proof sketch of Theorem 9.5.3, we can apply the localization lemma to get an interval $[a, b]$ and an affine function ℓ such that

$$\int_0^1 \ell(t)^{n-1} g((1-t)a + tb)\,dt > 0 \quad \text{and} \quad \int_0^1 \ell(t)^{n-1} h((1-t)a + tb)\,dt > 0. \tag{9.5.2}$$

The functions g, h as we have defined them are not lower semi-continuous. However, this can be addressed by expanding S_1 and S_2 slightly so as to make them open sets, and making the support of f an open set. Since we are proving strict inequalities, these modifications do not affect the conclusion.

Let us partition $[0,1]$ into Z_1, Z_2, Z_3 as follows:

$$Z_i = \{t \in [0,1] : (1-t)a + tb \in S_i\}.$$

Note that for any pair of points $u \in Z_1, v \in Z_2$, $|u - v| \geq d(S_1, S_2)/D$. We can rewrite (9.5.2) as

$$\int_{Z_3} \ell(t)^{n-1} f((1-t)a + tb)\,dt < C \int_{Z_1} \ell(t)^{n-1} f((1-t)a + tb)\,dt$$

and

$$\int_{Z_3} \ell(t)^{n-1} f((1-t)a + tb)\,dt < C \int_{Z_2} \ell(t)^{n-1} f((1-t)a + tb)\,dt.$$

The functions f and $\ell(\cdot)^{n-1}$ are both logconcave, so $F(t) = \ell(t)^{n-1} f((1-t)a + tb)$ is also logconcave. We get,

$$\int_{Z_3} F(t)\,dt < C \min\left\{\int_{Z_1} F(t)\,dt, \int_{Z_2} F(t)\,dt\right\}. \tag{9.5.3}$$

Now consider what Theorem 9.5.3 asserts for the function $F(t)$ over the interval $[0,1]$ and the partition Z_1, Z_2, Z_3:

$$\int_{Z_3} F(t)\,dt \geq 2d(Z_1, Z_2) \min\left\{\int_{Z_1} F(t)\,dt, \int_{Z_2} F(t)\,dt\right\}. \tag{9.5.4}$$

We have substituted 1 for the diameter of the interval $[0,1]$. Also, $2d(Z_1, Z_2) \geq 2d(S_1, S_2)/D = C$. Thus, Theorem 9.5.3 applied to the function $F(t)$ contradicts (9.5.3) and to prove the theorem in general, and it suffices to prove it in the one-dimensional case. A combinatorial argument reduces this to the case when each Z_i is a single interval. Proving the resulting inequality up to a factor of 2 is a simple exercise and uses only the unimodality of F. The improvement to the tight bound requires one-dimensional logconcavity. This completes the proof of Theorem 9.5.3.

The localization lemma has been used to prove a variety of isoperimetric inequalities. The next theorem is a refinement of Theorem 9.5.3, replacing the diameter by the square-root of the expected squared distance of a random point from the mean. For an isotropic distribution this is an improvement from n to $\sqrt{n}$. This theorem was proved by Kannan-Lovász-Simonovits in the same paper in which they proposed the KLS conjecture.

Theorem 9.5.6 ([27]). *For any logconcave density p in $\mathbb{R}^n$ with covariance matrix A, the KLS constant satisfies*

$$\psi_p \gtrsim \frac{1}{\sqrt{\mathrm{tr}(A)}}.$$

The next theorem shows that the KLS conjecture is true for an important family of distributions. The proof is again by localization [16], and the one-dimensional inequality obtained is a Brascamp-Lieb Theorem. We note that the same theorem can be obtained by other means [37, 21].

Theorem 9.5.7. *Let $h(x) = f(x)e^{-\frac{1}{2}x^\top Bx}/\int f(y)e^{-\frac{1}{2}y^\top By}dy$ where $f : \mathbb{R}^n \to \mathbb{R}_+$ is an integrable logconcave function and B is positive definite. Then h is logconcave and for any measurable subset S of $\mathbb{R}^n$,*

$$\frac{h(\partial S)}{\min\left\{h(S), h\left(\mathbb{R}^n \setminus S\right)\right\}} \gtrsim \frac{1}{\|B^{-1}\|_{\mathrm{op}}^{\frac{1}{2}}}.$$

In other words, the expansion of h is $\Omega(\|B^{-1}\|_{\mathrm{op}}^{-\frac{1}{2}})$.

The analysis of the Gaussian Cooling algorithm for volume computation [17] uses localization.

Next we mention an application to the anti-concentration of polynomials. This is a corollary of a more general result by Carbery and Wright.

Theorem 9.5.8 ([11]). *Let q be a degree d polynomial in $\mathbb{R}^n$. Then for a convex body $K \subset \mathbb{R}^n$ of volume 1, any $\epsilon > 0$, and x drawn uniform from K,*

$$\Pr_{x\sim K}\left(|q(x)| \le \epsilon \max_K |q(x)|\right) \lesssim \epsilon^{\frac{1}{d}} n$$

We conclude this section with a nice interpretation of the localization lemma by Fradelizi and Guedon. They also give a version that extends localization to multiple inequalities.

Theorem 9.5.9 (Reformulated Localization Lemma [23]). *Let K be a compact convex set in $\mathbb{R}^n$ and f be an upper semi-continuous function. Let P_f be the set of logconcave distributions μ supported by K satisfying $\int f d\mu \ge 0$. The set of extreme points of convP_f is exactly:*

1. *the Dirac measure at points x such that $f(x) \ge 0$, or*
2. *the distributions v satisfies*
   (a) *density function is of the form e^ℓ with linear ℓ,*
   (b) *support equals to a segment $[a, b] \subset K$,*
   (c) *$\int f dv = 0$,*
   (d) *$\int_a^x f dv > 0$ for $x \in (a, b)$ or $\int_x^b f dv > 0$ for $x \in (a, b)$.*

Since we know the maximizer of any convex function is at extreme points, this shows that one can optimize $\max_{\mu\in P_f} \Phi(\mu)$ for any convex Φ by checking Dirac measures and log-affine functions.

Chapter 10

Volume

10.1 Simulated Annealing

In this chapter we study a sampling-based approach for optimization and integration in high dimension. The main idea is to sample a sequence of logconcave distributions, starting with one that is easy to integrate and ending with the function whose integral is desired. This process is known as *simulated annealing.* The same high-level algorithm can be used for optimization, volume computation/integration or rounding. For integration, the desired integral can be expressed as the following telescoping product:

$$\int_{\mathbb{R}^n} f = \int f_0 \frac{\int f_i}{\int f_0} \frac{\int f_2}{\int f_3} \cdots \frac{\int f_m}{\int f_{m-1}}$$

where $f_m = f$. Each ratio $\int f_{i+1} / \int f_i$ is the expectation of the estimator $Y = \frac{f_{i+1}(X)}{f_i(X)}$ for X drawn from the density proportional to f_i. What sequence of functions should we use?

Algorithm 27: `SimulatedAnnealing`

1. For $i = 0, \ldots, m$, define
$$a_i = b\left(1 + \frac{1}{\sqrt{n}}\right)^i \text{ and } f_i(x) = f(x)^{a_i}.$$
2. Let $X_0^1, \ldots, X_0^k$ be independent random points with density proportional to f_0.
3. For $i = 0, \ldots, m-1$: starting with $X_i^1, \ldots, X_i^k$, generate random points $X_{i+1} = \{X_{i+1}^1, \ldots, X_{i+1}^k\}$; update a running estimate g based on these samples; update the isotropy transformation using the samples.
4. Output the final estimate of g.

return x.

For optimization, the function f_m is set to be a sufficiently high power of f, the function to be maximized while g is simply the maximum objective value so far. For integration and rounding, $f_m = f$, the target function to be integrated or rounded. For integration, the function g starts out as the integral of f_0 and is multiplied by the ratio $\int f_{i+1} / \int f_i$ in each step. For rounding, g is simply the estimate of the isotropic transformation for the current function.

For sampling and optimization, it is natural to ask why one uses a sequence of functions rather than jumping straight to the target function f_m. For optimizing a linear function $c^T x$ over a convex set K, all we need to do is sample according to the density proportional to $e^{-a(c^T x)}$ for a sufficiently large coefficient a (recall Theorem 4.5.1). The reason for using a sequence is that the function f_m and corresponding density p_m can be very far from the starting function or distribution, and hence the mixing time can be high. The sequence ensures that samples from the current function provide a good (warm) start for sampling from the next function. This is captured in the next lemma.

Theorem 10.1.1. *Let $p_i(x) = f_i(x) / \int f_i$ with f_i defined as in the annealing algorithm for some logconcave function*

$f : \mathbb{R}^n \to \mathbb{R}_+$. Then a random sample $X \sim p_i$ satisfies

$$\mathbf{E}_{p_i}\left(\frac{p_i(x)}{p_{i+1}(x)}\right) = O(1).$$

The underlying mathematical property behind this lemma is the following.

Lemma 10.1.2. *Let f be a logconcave function in $\mathbb{R}^n$. Then $Z(a) = a^n \int_{\mathbb{R}^n} f(x)^a$ is logconcave for $a \geq 0$. If f has support K, then $Z(a) = a^n \int_K f(ax)$ is logconcave for $a > 0$.*

10.2 Volume Computation

For volume computation, we can apply annealing as follows. We assume that the input convex body K contains a unit ball, has diameter bounded by D and is given by a membership oracle. The polynomial-time algorithm of Dyer, Frieze and Kannan [20] uses a sequence of uniform distributions on convex bodies, starting with the ball contained inside the input body K. Each body in the sequence is a ball intersected with the given convex body K: $K_i = 2^{\frac{i}{n}} rB \cap K$ and $f_i(x)$ is the indicator of K_i. The length the sequence is $m = O(n \log D)$ so that the final body is just K. A variance computation shows that $O(m/\epsilon^2)$ samples per distribution suffice to get an overall $1+\epsilon$ multiplicative error approximation with high probability. The total number of samples is $O^*(m^2) = O^*(n^2)$ and the complexity of the resulting algorithm is $O^*(n^5)$ as shown in [28]. Table 10.1 below summarizes progress on the volume problem over the past three decades. Besides improving the complexity of volume computation, each step has typically resulted in new techniques. For more details, we refer the reader to surveys on the topic [57, 61].

Year/Authors	New ingredients	Steps
1989/Dyer-Frieze-Kannan [20]	Everything	n^{23}
1990/Lovász-Simonovits [44]	Better isoperimetry	n^{16}
1990/Lovász [43]	Ball walk	n^{10}
1991/Applegate-Kannan [4]	Logconcave sampling	n^{10}
1990/Dyer-Frieze [19]	Better error analysis	n^8
1993/Lovász-Simonovits [45]	Localization lemma	n^7
1997/Kannan-Lovász-Simonovits [28]	Speedy walk, isotropy	n^5
2003/Lovász-Vempala [46]	Annealing, hit-and-run	n^4
2015/Cousins-Vempala [17] (well-rounded)	Gaussian Cooling	n^3
2017/Lee-Vempala (polytopes)	Hamiltonian Walk	$mn^{\frac{2}{3}}$

Table 10.1: The complexity of volume estimation, each step uses $\widetilde{O}(n)$ bit of randomness. The last algorithm needs $\widetilde{O}\left(mn^{\omega-1}\right)$ steps per iteration while the rest need $O(n^2)$ per oracle query.

In [46] this was improved by sampling from a sequence of nonuniform distributions. Then we consider the following estimator:

$$Y = \frac{f_{i+1}(X)}{f_i(X)}.$$

We see that

$$\mathbf{E}_{f_i}(Y) = \frac{\int f_{i+1}}{\int f_i}.$$

In the algorithm of DFK and KLS, this ratio is bounded by a constant in each phase, giving a total of $O^*(n)$ phases since the ratio of final to initial integrals is exponential. Instead of uniform densities, we consider

$$f_i(x) \propto \exp(-a_i \|x\|)\chi_K(x) \text{ or } f_i(x) \propto \exp(-a_i \|x\|^2)\chi_K(x).$$

The coefficient a_i (inverse "temperature") will be changed by a factor of $(1 + \frac{1}{\sqrt{n}})$ in each phase, which implies that $m = \widetilde{O}(\sqrt{n})$ phases suffice to reach the target distribution. This is perhaps surprising since the ratio of the initial integral to the final is typically $n^{\Omega(n)}$. Yet the algorithm uses only $\widetilde{O}(\sqrt{n})$ phases, and hence estimates a ratio of

$n^{\tilde{\Omega}(\sqrt{n})}$ in one or more phases. The key insight is that even though the expected ratio might be large, its variance is not.

Lemma 10.2.1. *For $X \sim f_i$ with $f_i(x) = e^{-a_i\|x\|}\chi_K(x)$ for a convex body K, or $f_i(x) = f(x)^{a_i}$ for a logconcave function f, we have that the estimator $Y = \frac{f_{i+1}(X)}{f_i(X)}$ satisfies*

$$\frac{\mathbf{E}\left(Y^2\right)}{\mathbf{E}\left(Y\right)} \leq \left(\frac{a_{i+1}^2}{(2a_{i+1} - a_i)a_i}\right)^n$$

which is bounded by a constant for $a_i = a_{i+1}\left(1 + \frac{1}{\sqrt{n}}\right)$.

Theorem 10.2.2 ([46])**.** *The volume of a convex body in $\mathbb{R}^n$ (given by a membership oracle) can be computed to relative error ε using $\widetilde{O}(n^4/\varepsilon^2)$ oracle queries and $\widetilde{O}(n^2)$ arithmetic operations per query.*

The LV algorithm has two parts. In the first it finds a transformation that puts the body in near-isotropic position. The complexity of this part is $\widetilde{O}(n^4)$. In the second part, it runs the annealing schedule, while maintaining that the distribution being sampled is *well-rounded*, a weaker condition than isotropy. Well-roundedness requires that a level set of measure $\frac{1}{8}$ contains a constant-radius ball and the trace of the covariance (expected squared distance of a random point from the mean) to be bounded by $O(n)$, so that R/r is effectively $O(\sqrt{n})$. To achieve the complexity guarantee for the second phase, it suffices to use the KLS bound of $\psi_p \gtrsim n^{-\frac{1}{2}}$. Connecting improvements in the Cheeger constant directly to the complexity of volume computation is an open question. To apply improvements in the Cheeger constant, one would need to replace well-roundedness with (near-)isotropy and maintain that. However, maintaining isotropy appears to be much harder — possibly requiring a sequence of $\Omega(n)$ distributions and $\Omega(n)$ samples from each, providing no gain over the current complexity of $O^*(n^4)$ even if the KLS conjecture turns out to be true.

A faster algorithm is known for well-rounded convex bodies (any isotropic logconcave density satisfies $\frac{R}{r} = O(\sqrt{n})$ and is well-rounded). This variant of simulated annealing, called Gaussian cooling utilizes the fact that the KLS conjecture holds for a Gaussian density restricted by any convex body, and completely avoids computing an isotropic transformation.

Theorem 10.2.3 ([17])**.** *The volume of a well-rounded convex body, i.e., with $R/r = O^*(\sqrt{n})$, can be computed using $O^*(n^3)$ oracle calls.*

■ 10.3 Rounding

Theorem 10.3.1. *Any convex body can be put in 2-isotropic position in $O^*(n^4)$ membership oracle queries and $O(n^2)$ additional computation per query.*

[1] Radosław Adamczak, Alexander Litvak, Alain Pajor, and Nicole Tomczak-Jaegermann. Quantitative estimates of the convergence of the empirical covariance matrix in log-concave ensembles. *Journal of the American Mathematical Society*, 23(2):535–561, 2010.

[2] Deeksha Adil, Rasmus Kyng, Richard Peng, and Sushant Sachdeva. Iterative refinement for âp-norm regression. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1405–1424. SIAM, 2019.

[3] Zeyuan Allen-Zhu, Zheng Qu, Peter Richtárik, and Yang Yuan. Even faster accelerated coordinate descent using non-uniform sampling. In *International Conference on Machine Learning*, pages 1110–1119, 2016.

[4] D. Applegate and R. Kannan. Sampling and integration of near log-concave functions. In *STOC*, pages 156–163, 1991.

[5] Shiri Artstein-Avidan and Vitali Milman. The concept of duality in convex analysis, and the characterization of the legendre transform. *Annals of mathematics*, pages 661–674, 2009.

[6] David S Atkinson and Pravin M Vaidya. A cutting plane algorithm for convex programming that uses analytic centers. *Mathematical Programming*, 69(1-3):1–43, 1995.

[7] Alexandre Belloni, Tengyuan Liang, Hariharan Narayanan, and Alexander Rakhlin. Escaping the local minima via simulated annealing: Optimization of approximately convex functions. In *Conference on Learning Theory*, pages 240–265, 2015.

[8] Dimitris Bertsimas and Santosh Vempala. Solving convex programs by random walks. *Journal of the ACM (JACM)*, 51(4):540–556, 2004.

[9] Thomas Blumensath and Mike E Davies. Iterative hard thresholding for compressed sensing. *Applied and computational harmonic analysis*, 27(3):265–274, 2009.

[10] Sébastien Bubeck, Ronen Eldan, and Yin Tat Lee. Kernel-based methods for bandit convex optimization. *arXiv preprint arXiv:1607.03084*, 2016.

[11] Anthony Carbery and James Wright. Distributional and l^ q norm inequalities for polynomials over convex bodies in r^ n. *Mathematical research letters*, 8(3):233–248, 2001.

[12] Yair Carmon, John C Duchi, Oliver Hinder, and Aaron Sidford. Lower bounds for finding stationary points i. *arXiv preprint arXiv:1710.11606*, 2017.

[13] Michael B Cohen. Nearly tight oblivious subspace embeddings by trace inequalities. In *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*, pages 278–287. SIAM, 2016.

[14] Michael B Cohen, Yin Tat Lee, Cameron Musco, Christopher Musco, Richard Peng, and Aaron Sidford. Uniform sampling for matrix approximation. In *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science*, pages 181–190. ACM, 2015.

[15] Michael B Cohen, Yin Tat Lee, and Zhao Song. Solving linear programs in the current matrix multiplication time. *arXiv preprint arXiv:1810.07896*, 2018.

[16] B. Cousins and S. Vempala. A cubic algorithm for computing Gaussian volume. In *SODA*, pages 1215–1228, 2014.

[17] B. Cousins and S. Vempala. Bypassing KLS: Gaussian cooling and an $O^*(n^3)$ volume algorithm. In *STOC*, pages 539–548, 2015.

[18] Dmitriy Drusvyatskiy, Maryam Fazel, and Scott Roy. An optimal first order method based on optimal quadratic averaging. *arXiv preprint arXiv:1604.06543*, 2016.

[19] M. E. Dyer and A. M. Frieze. Computing the volume of a convex body: a case where randomness provably helps. In *Proc. of AMS Symposium on Probabilistic Combinatorics and Its Applications*, pages 123–170, 1991.

[20] M. E. Dyer, A. M. Frieze, and R. Kannan. A random polynomial time algorithm for approximating the volume of convex bodies. In *STOC*, pages 375–381, 1989.

[21] R. Eldan. Thin shell implies spectral gap up to polylog via a stochastic localization scheme. *Geometric and Functional Analysis*, 23:532–569, 2013.

[22] Vitaly Feldman, Cristobal Guzman, and Santosh Vempala. Statistical query algorithms for stochastic convex optimization. *CoRR*, abs/1512.09170, 2015.

[23] Matthieu Fradelizi and Olivier Guédon. The extreme points of subsets of s-concave probabilities and a geometric localization theorem. *Discrete & Computational Geometry*, 31(2):327–335, 2004.

[24] Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric algorithms and combinatorial optimization*, volume 2. Algorithms and Combinatorics, 1988.

[25] Haotian Jiang, Yin Tat Lee, Zhao Song, and Sam Chiu-wai Wong. An improved cutting plane method for convex optimization, convex-concave games, and its applications. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 944–953, 2020.

[26] Adam Tauman Kalai and Santosh Vempala. Simulated annealing for convex optimization. *Math. Oper. Res.*, 31(2):253–266, February 2006.

[27] R. Kannan, L. Lovász, and M. Simonovits. Isoperimetric problems for convex bodies and a localization lemma. *Discrete & Computational Geometry*, 13:541–559, 1995.

[28] R. Kannan, L. Lovász, and M. Simonovits. Random walks and an $O^*(n^5)$ volume algorithm for convex bodies. *Random Structures and Algorithms*, 11:1–50, 1997.

[29] Ravi Kannan, László Lovász, and Miklós Simonovits. Isoperimetric problems for convex bodies and a localization lemma. *Discrete & Computational Geometry*, 13(1):541–559, 1995.

[30] Ravindran Kannan and Santosh Vempala. Randomized algorithms in numerical linear algebra. *Acta Numerica*, 26:95–135, 2017.

[31] Alexander Karzanov and Leonid Khachiyan. On the conductance of order Markov chains. *Order*, 8(1):7–15, 1991.

[32] Tarun Kathuria, Yang P Liu, and Aaron Sidford. Unit capacity maxflow in almost $o(m^{4/3})$ time. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 119–130. IEEE, 2020.

[33] L Khachiyan, S Tarasov, and E Erlich. The inscribed ellipsoid method. In *Soviet Math. Dokl*, volume 298, 1988.

[34] Leonid G Khachiyan. Polynomial algorithms in linear programming. *USSR Computational Mathematics and Mathematical Physics*, 20(1):53–72, 1980.

[35] B. Klartag. Needle decompositions in riemannian geometry. 2017.

[36] Rasmus Kyng, Richard Peng, Sushant Sachdeva, and Di Wang. Flows in almost linear time via adaptive preconditioning. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 902–913, 2019.

[37] Michel Ledoux. Concentration of measure and logarithmic sobolev inequalities. *Seminaire de probabilites de Strasbourg*, 33:120–216, 1999.

[38] Yin Tat Lee, Aaron Sidford, and Santosh S Vempala. Efficient convex optimization with membership oracles. *arXiv preprint arXiv:1706.07357*, 2017.

[39] Yin Tat Lee, Aaron Sidford, and Sam Chiu-wai Wong. A faster cutting plane method and its implications for combinatorial and convex optimization. In *Foundations of Computer Science (FOCS), 2015 IEEE 56th Annual Symposium on*, pages 1049–1065. IEEE, 2015.

[40] Yin Tat Lee and Santosh S Vempala. Stochastic localization+ stieltjes barrier= tight bound for log-sobolev. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1122–1129. ACM, 2018.

[41] A Yu Levin. On an algorithm for the minimization of convex functions. In *Soviet Mathematics Doklady*, volume 160, pages 1244–1247, 1965.

[42] Peter Li and Shing Tung Yau. Estimates of eigenvalues of a compact riemannian manifold. *Geometry of the Laplace operator*, 36:205–239, 1980.

[43] L. Lovász. How to compute the volume? *Jber. d. Dt. Math.-Verein, Jubiläumstagung 1990*, pages 138–151, 1990.

[44] L. Lovász and M. Simonovits. Mixing rate of Markov chains, an isoperimetric inequality, and computing the volume. In *ROCS*, pages 482–491, 1990.

[45] L. Lovász and M. Simonovits. Random walks in a convex body and an improved volume algorithm. In *Random Structures and Alg.*, volume 4, pages 359–412, 1993.

[46] L. Lovász and S. Vempala. Simulated annealing in convex bodies and an $O^*(n^4)$ volume algorithm. *J. Comput. Syst. Sci.*, 72(2):392–417, 2006.

[47] László Lovász and Miklós Simonovits. Random walks in a convex body and an improved volume algorithm. *Random structures & algorithms*, 4(4):359–412, 1993.

[48] László Lovász and Santosh Vempala. The geometry of logconcave functions and sampling algorithms. *Random Struct. Algorithms*, 30(3):307–358, 2007.

[49] Haihao Lu, Robert M Freund, and Yurii Nesterov. Relatively-smooth convex optimization by first-order methods, and applications. *arXiv preprint arXiv:1610.05708*, 2016.

[50] Paolo Manselli and Carlo Pucci. Maximum length of steepest descent curves for quasi-convex functions. *Geometriae Dedicata*, 38(2):211–227, 1991.

[51] Yu Nesterov. Introductory lectures on convex programming volume i: Basic course. *Lecture notes*, 1998.

[52] Donald J Newman. Location of the maximum on unimodal surfaces. *Journal of the ACM (JACM)*, 12(3):395–398, 1965.

[53] Lawrence E Payne and Hans F Weinberger. An optimal poincaré inequality for convex domains. *Archive for Rational Mechanics and Analysis*, 5(1):286–292, 1960.

[54] Luis Rademacher. Approximating the centroid is hard. In *Proceedings of the 23rd ACM Symposium on Computational Geometry, Gyeongju, South Korea, June 6-8, 2007*, pages 302–305, 2007.

[55] Sushant Sachdeva, Nisheeth K Vishnoi, et al. Faster algorithms via approximation theory. *Foundations and Trends® in Theoretical Computer Science*, 9(2):125–210, 2014.

[56] Naum Z Shor. Cut-off method with space extension in convex programming problems. *Cybernetics and systems analysis*, 13(1):94–96, 1977.

[57] Miklós Simonovits. How to compute the volume in high dimension? *Math. Program.*, 97(1-2):337–374, 2003.

[58] George J Stigler. The cost of subsistence. *Journal of farm economics*, 27(2):303–314, 1945.

[59] Joel A Tropp et al. An introduction to matrix concentration inequalities. *Foundations and Trends® in Machine Learning*, 8(1-2):1–230, 2015.

[60] Pravin M Vaidya. A new algorithm for minimizing convex functions over convex sets. In *Foundations of Computer Science, 1989., 30th Annual Symposium on*, pages 338–343. IEEE, 1989.

[61] S. Vempala. Geometric random walks: A survey. *MSRI Combinatorial and Computational Geometry*, 52:573–612, 2005.

[62] S. S. Vempala. *The Random Projection Method*. AMS, 2004.

[63] Andre Wibisono. Sampling as optimization in the space of measures: The langevin dynamics as a composite optimization problem. *arXiv preprint arXiv:1802.08089*, 2018.

[64] David P Woodruff et al. Sketching as a tool for numerical linear algebra. *Foundations and Trends® in Theoretical Computer Science*, 10(1–2):1–157, 2014.

[65] David B Yudin and Arkadii S Nemirovski. Evaluation of the information complexity of mathematical programming problems. *Ekonomika i Matematicheskie Metody*, 12:128–142, 1976.

Appendix A

Calculus - Review

A.1 Tips for Computing Gradient

In this section, we give some tips on how to do calculations.

Computing Gradient via Directional Derivatives

Usually, computing gradient coordinate-by-coordinate is not the best way and we should avoid using summation notation as much as possible as it creates too many subscripts and is prone to mistakes. Instead, it is usually better to compute the gradient via directional deriatives. Here, we give few examples for this. We define $Df(x)[h]$ be the directional derivative of f at x along the direction h. Namely,

$$Df(x)[h] \stackrel{\text{def}}{=} \left.\frac{d}{dt}\right|_{t=0} f(x+th).$$

Similarly, we use $D^k f(x)[h_1, h_2, \cdots, h_k]$ to denote the directional k-th derivative of f at x along directions $h_1, \cdots, h_k$.

Lemma A.1.1. *Given $A \in \mathbb{R}^{n\times d}$. Let $\Phi(x) = \sum_{i=1}^n f(a_i^\top x)$ where a_i is the i-th row of A. Then, we have $\nabla\Phi(x) = A^\top f'(Ax)$ and $\nabla^2\Phi(x) = A^\top \mathrm{diag}(f''(Ax))A$ where $f'(Ax)$ is the vector defined by $(f'(Ax))_i = f'(a_i^\top x)$.*

Proof. Note that

$$D\Phi(x)[h] = \sum_{i=1}^n f'(a_i^\top x)a_i^\top h,$$
$$D^2\Phi(x)[h,h] = \sum_{i=1}^n f''(a_i^\top x)(a_i^\top h)^2.$$

To write it in the traditional form, we note that

$$\nabla\Phi(x)^\top h = D\Phi(x)[h] = f'(Ax)^\top Ah = (A^\top f'(Ax))^\top h.$$

Since both side are the same for all h, we have $\nabla\Phi(x) = A^\top f'(Ax)$.

Similarly, we have

$$\begin{aligned} h^\top\nabla^2\Phi(x)h &= D^2\Phi(x)[h,h] \\ &= \sum_{i=1}^n (f''(Ax))_i (Ah)_i^2 \\ &= h^\top A^\top \mathrm{diag}(f''(Ax))Ah. \end{aligned}$$

Since $\nabla^2\Phi(x) - A^\top\mathrm{diag}(f''(Ax))A$ is symmetric and both side are the same for all h, we have $\nabla^2\Phi(x) = A^\top\mathrm{diag}(f''(Ax))A$. □

Exercise A.1.2. Use the above method to compute the gradient and Hessian of $f(X) = \log\det A^T XA$.

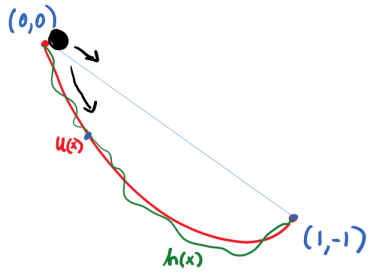


Figure A.1.1: The "fastest" curve

Here is a more complicated example:

Lemma A.1.3 (Brachistochrone Problem)**.** *Let $(x, u(x))$ be the curve from $(0,0)$ to $(1,-1)$ where the first coordinate is the x axis and the second coordinate is the y axis. Suppose that this is the curve that takes the shortest time for a bead to slide along the curve frictionlessly from $(0,0)$ to $(1,-1)$ under uniform gravity. Then, we have that*

$$2uu'' + (u')^2 + 1 = 0.$$

Remark. Take a look at Wikipedia for Brachistochrone curve. It is very counterintuitive!

Proof. Given a curve $u = u(x)$, the total travel time is

$$T(u) = \int_0^1 \frac{ds(x)}{v(x)} = \int_0^1 \frac{\sqrt{1 + (u'(x))^2}}{v(x)} dx$$

where ds is the arc length element and $v(x)$ is the velocity at x. By conservation of energy, i.e., the gained kinetic energy must equal the lost potential energy for every point along the curve, we know that

$$\frac{1}{2} m v(x)^2 = -mgu(x).$$

Hence, we have $v(x) = \sqrt{-2gu(x)}$ and so

$$T(u) = \int_0^1 \sqrt{\frac{1 + (u'(x))^2}{-2gu(x)}} dx.$$

Since u is a *shortest* curve, any local change in u cannot reduce the time, i.e.,

$$DT(u)[h] = 0$$

for any change h of the curve u. We next compute the directional derivative of $T(u)$, i.e., $\frac{d}{dt}|_{t=0} T(u + th)$:

$$\begin{aligned} DT(u)[h] &= \int_0^1 -\frac{1}{2} \frac{\sqrt{1 + u'(x)^2}}{\sqrt{-2gu(x)^{3/2}}} \frac{d}{dt} u(x) dx + \int_0^1 \frac{1}{2} \frac{2u'(x)}{\sqrt{-2gu(x)}\sqrt{1 + u'(x)^2}} \frac{d}{dt} u'(x) dx. \\ &= \int_0^1 -\frac{1}{2} \frac{\sqrt{1 + u'(x)^2}}{\sqrt{-2gu(x)}u(x)} h(x) dx + \int_0^1 \frac{u'(x)h'(x)}{\sqrt{-2gu(x)}\sqrt{1 + u'(x)^2}} dx. \end{aligned}$$

Note that the second term involves $h'(x)$. To change the term $h'(x)$ to $h(x)$, we use the integration by parts (with respect to x, not t!):

$$\int_0^1 \frac{u'(x)h'(x)}{\sqrt{-2gu(x)}\sqrt{1 + (u'(x))^2}} dx = \left[\frac{u'(x)h(x)}{\sqrt{-2gu(x)}\sqrt{1 + (u'(x))^2}} \right]_0^1 - \int_0^1 \frac{d}{dx} \left(\frac{u'(x)}{\sqrt{-2gu(x)}\sqrt{1 + (u'(x))^2}} \right) h(x) dx.$$

Since the endpoints of the curve are fixed, we have $h(1) = h(0) = 0$. Hence, the first term on the right hand side is 0. Continuing,

$$\begin{aligned}
DT(u)[h] &= \int_0^1 -\frac{1}{2}\frac{\sqrt{1+u'(x)^2}}{\sqrt{-2gu(x)}u(x)}h(x)dx - \int_0^1 \frac{d}{dx}\left(\frac{u'(x)}{\sqrt{-2gu(x)}\sqrt{1+u'(x)^2}}\right)h(x)dx \\
&= \int_0^1 -\frac{1}{2}\frac{\sqrt{1+(u'(x))^2}}{\sqrt{-2gu(x)}u(x)}h(x)dx - \int_0^1 \frac{u''(x)}{\sqrt{-2gu(x)}\sqrt{1+u'(x)^2}}h(x)dx \\
&\quad + \int_0^1 \frac{1}{2}\frac{u'(x)^2}{\sqrt{-2gu(x)}u(x)\sqrt{1+u'(x)^2}}h(x)dx \\
&\quad + \int_0^1 \frac{u'(x)u'(x)u''(x)}{\sqrt{-2gu(x)}(1+u'(x)^2)^{3/2}}h(x)dx.
\end{aligned}$$

Hence, we have $DT(u)[h] = \int_0^1 a(x)h(x)dx$ where

$$\begin{aligned}
a(x) =& \frac{-1}{2}\frac{\sqrt{1+(u'(x))^2}}{\sqrt{-2gu(x)}u(x)} - \frac{u''(x)}{\sqrt{-2gu(x)}\sqrt{1+u'(x)^2}} + \frac{1}{2}\frac{u'(x)^2}{\sqrt{-2gu(x)}u(x)\sqrt{1+u'(x)^2}} \\
&+ \frac{u'(x)u'(x)u''(x)}{\sqrt{-2gu(x)}(1+u'(x)^2)^{3/2}}.
\end{aligned} \tag{A.1.1}$$

Note that $a(x)$ is the gradient of T. Since $DT(u)[h] = 0$ for all $h(x)$, we have that $a(x) = 0$ for all x. Multiplying both sides of (A.1.1) by $2\sqrt{-2gu(x)}(1+(u'(x))^2)^{3/2}u(x)$, we have

$$\begin{aligned}
0 =& -(1+(u'(x))^2)^2 - 2u(x)u''(x)(1+(u'(x))^2) + u'(x)^2(1+(u'(x))^2) + 2u(x)u'(x)^2u''(x) \\
=& -1 - u'(x)^2 - 2u(x)u''(x).
\end{aligned}$$

□

Taking Derivatives on Both Sides

Suppose we have a function $f(x, y)$ and $g(x)$ such that $f(x, g(x)) = 0$. The implicit function theorem shows that

$$D_x f(x, g(x)) + D_y f(x, g(x))Dg(x) = 0$$

where $D_x f$ is the Jacobian of f with respect to x variables and $Dg(x)$ is the Jacobian of g. We note that the formula can be obtained from taking derivative on both sides with respective to x. Sometimes, taking derivatives on the both sides can greatly simplify calculations. Here are some examples.

Lemma A.1.4. *Consider $x_t = \operatorname{argmin}_{x\in\mathbb{R}^n} f_t(x)$ where f_t are strictly convex. Then, we have*

$$\frac{dx_t}{dt} = (\nabla^2 f_t(x_t))^{-1}\nabla\frac{df_t}{dt}(x_t).$$

Proof. By the optimality condition, we have $\nabla f_t(x_t) = 0$. Taking derivatives on both sides, we have

$$\nabla^2 f_t(x_t)\frac{dx_t}{dt} + \nabla\frac{df_t}{dt}(x_t) = 0.$$

Since f_t are strictly convex, $\nabla^2 f_t(x_t)$ is positive definite and is invertible. Hence, we have that the result. □

In section 5.4, we used this to compute the derivative of central path.

■ A.2 Solving Optimization Problems by Hand

In this section, we introduce the KKT condition and show how to use it to solve optimization problem by hand.

Theorem A.2.1 (Karush–Kuhn–Tucker theorem)**.** *Consider the following optimization problem*

$$\min_{x\in\Omega} f(x) \text{ subject to } h_i(x) \le 0 \text{ and } \ell_j(x) = 0 \text{ for all } i, j$$

for some open set Ω and continuously differentiable functions f, h_i and ℓ_j. If x is a local minimum, x satisfies the KKT conditions:

- *Stationary:* $\nabla f(x) + \sum_i u_i \nabla h_i(x) + \sum_j v_j \nabla \ell_j(x) = 0$
- *Complementary Slackness:* $u_i h_i(x) = 0$ *for all* i
- *Primal Feasibility:* $h_i(x) \le 0$ *and* $\ell_j(x) = 0$ *for all* i, j
- *Dual Feasibility:* $u_i \ge 0$ *for all* i

We prove Holder's inequality as an example:

Fact A.2.2. *For any vector $x, y \in \mathbb{R}^n$, we have $\|xy\|_1 \le \|x\|_p \|y\|_q$ for any $1 \le p \le \infty$ and $1 \le q \le \infty$ with $\frac{1}{p} + \frac{1}{q} = 1$.*

Proof. By symmetries, it suffices to compute

$$\max_{\|x\|_p \le 1} \sum_i x_i y_i$$

for non-zero y. Now, we use the KKT theorem with $f(x) = -\sum_i x_i y_i$, $h(x) = \|x\|_p - 1$ and $\Omega = \mathbb{R}^n$. By the KKT conditions, for any maximizer x, we have that

$$\begin{aligned} -\nabla f(x) + u\nabla h(x) &= 0, \\ uh(x) &= 0, \\ h(x) \le 0, u &\ge 0. \end{aligned}$$

Note that $\nabla f(x) = y$ and $(\nabla h(x))_i = \frac{1}{p}\|x\|_p^{1-p} \cdot p x_i^{p-1} = \|x\|_p^{1-p} \cdot x^{p-1}$. From the stationary condition, we have

$$y = u\|x\|_p^{1-p} \cdot x^{p-1}.$$

To compute u, we note that y is non-zero and hence $u \neq 0$. From the complementary slackness, we have $h(x) = 0$ and hence $\|x\|_p = 1$. Therefore, we have

$$y = u \cdot x^{p-1}.$$

Hence, we have $1 = \sum_i x_i^p = \sum_i (y_i/u)^{\frac{p}{p-1}} = \sum_i (y_i/u)^q$. Hence, we have $u = \|y\|_q$

Now, we can compute $\sum_i x_i y_i$ as follows

$$\sum_i x_i y_i = \sum_i (\frac{y_i}{u})^{\frac{1}{p-1}} y_i = \frac{1}{u^{1/(p-1)}} \sum_i y_i^{p/(p-1)} = \|y\|_q.$$

□

Appendix B

Notation

Symbol	Description
$o_\epsilon(f)$	$o(f)$ for any fixed ϵ
$\langle a, b \rangle$	inner product $a^T b$