CPSC 516: Algorithms via Convex Optimization

Felix Zhou [1]

June 1, 2023

[1]From Professor Nisheeth Vishnoi's Lectures at Yale University in Spring 2023

Contents

I Background **7**

1 Multivariate Calculus **9**

1.1 Derivatives 9

1.1.1 Integrals 9

1.1.2 Taylor Approximation 10

1.2 Linear Algebra 10

1.2.1 Norms 10

2 Graphs & Matrices **13**

2.1 Graphs 13

2.2 Matrices 13

2.2.1 $Lx = b$ 14

3 Convexity **15**

3.1 Convex Sets 15

3.2 Convex Functions 16

3.2.1 Stronger Notions of Convexity 19

3.2.2 Optimality Conditions 20

II Convex Optimization **23**

4 Convex Optimization **25**

4.1 Membership & Separation . . . 25
4.1.1 Membership . . . 25
4.1.2 Separation . . . 26
4.2 Solving Convex Programs . . . 26
4.3 Encoding a Convex Function . . . 26

5 Gradient Descent **27**
5.1 The Algorithm . . . 27
5.1.1 Input . . . 27
5.1.2 Output . . . 27
5.1.3 Strategy . . . 27
5.2 Analysis . . . 28
5.3 Handling Constraints . . . 30
5.4 Maximum Flow . . . 30
5.4.1 The Problem . . . 30
5.4.2 Gradient Descent . . . 31
Convexity . . . 31
Lipschit Gradient . . . 31
Projection . . . 31
Initial Point . . . 32
Summary . . . 32

6 Multiplicative Weights Update Method **33**
6.1 Motivation . . . 33
6.2 Weighted Majority Method . . . 33
6.2.1 Analysis . . . 34
6.3 Multiplicative Weights Update Method . . . 35
6.3.1 The Algorithm . . . 35

6.3.2 Analysis . . . 36
6.4 Perfect Bipartite Matching . . . 37
6.4.1 The Algorithm . . . 37
6.4.2 Analysis . . . 38

7 Newton's Method **41**
7.1 Newton's Method . . . 41
7.2 Newton's Method for Optimization . . . 42
7.2.1 Interpretation . . . 42
7.2.2 Analysis . . . 43

8 Interior Point Methods **45**
8.1 Linear Programming . . . 45
8.2 Full-Dimensional IPM . . . 45
8.2.1 Intuition . . . 45
8.2.2 Analysis . . . 46
8.2.3 Initialization . . . 50
8.3 Subspace IPM . . . 50
8.4 Minimum Cost Flow . . . 51
8.4.1 Starting Point . . . 52

9 Ellipsoid Method **55**
9.1 Separation . . . 55
9.2 Optimization & Feasibility . . . 55
9.3 Ellipsoid Method . . . 56
9.3.1 Minimum Volume Ellipsoid . . . 57

Part I

Background

Chapter 1

Multivariate Calculus

We will work ignoring pathological examples and assume that derivates always exist and are continuous when it makes sense. This avoids bizarre behavior such as the Hessian not being symmetric.

1.1 Derivatives

We write $\frac{d}{dt}g(t), \dot{g}(t), g'(t)$ to denote the derivative in one dimension. Recall the *directional derivative* of $f : \mathbb{R}^n \to \mathbb{R}$ is given by

$$\boldsymbol{\nabla} f(x)[h] := \lim_{\eta \to 0} \frac{f(x + \eta h) - f(x)}{\eta}.$$

More generally, the k-th directional derivative is written as $D^k f(x)[h_1, \ldots, h_k]$.

The *gradient* of f is a vector $\boldsymbol{\nabla} f(x) \in \mathbb{R}^n$ such that the i-th entry is

$$\boldsymbol{\nabla}_i f(x) = Df(x)[e_i]$$

where $\{e_i\}$ is the canonical basis of $\mathbb{R}^n$.

Taking this one step further, the *Hessian* of $f : \mathbb{R}^n \to \mathbb{R}$ is matrix $\boldsymbol{\nabla}^2 f(x) \in \mathbb{R}^{n \times n}$ given by

$$\boldsymbol{\nabla}^2_{ij} f(x) = D^2 f(x)[e_i, e_j]$$

1.1.1 Integrals

Recall the FTC II, which states that if $f : [a, b] \to \mathbb{R}$ is continuously differentiable, then

$$\int_a^b \dot{f}(t)dt = f(b) - f(a).$$

Proposition 1.1.1 (Integral Representation)
Let $f : \mathbb{R}^n \to \mathbb{R}$ be continuously differentiable. For $x, y \in \mathbb{R}$, suppose g is given by

$$g(t) := f[x + t(y - x)].$$

The following hold:
(i) $\dot{g}(t) = \langle \nabla f[x + t(y - x)], y - x \rangle$
(ii) $f(y) = f(x) + \int_0^1 \dot{g}(t)dt$
Furthermore, if f has a continuous Hessian,
(i) $\ddot{g}(t) = (y - x)^T \nabla^2 f[x + t(y - x)](y - x)$
(ii) $\langle \nabla f(y) - \nabla f(x), y - x \rangle = \int_0^1 \ddot{g}(t)dt$

1.1.2 Taylor Approximation

The first order approximation of $f : \mathbb{R}^n \to \mathbb{R}$ at some $a \in \mathbb{R}^n$ is given by

$$f(x) \approx f(a) + \langle x - a, \nabla f(a) \rangle.$$

Similarly, the second order approximation is given by

$$f(x) \approx f(a) + \langle x - a, \nabla f(x) \rangle + \frac{1}{2}(x - a)^T \nabla^2 f(x)(x - a).$$

1.2 Linear Algebra

Recall that we can define a *pseudo-inverse* even for non-invertible linear maps which act on the orthogonal complement of the kernel.

A symmetric matrix $M \in \S^n$ is *positive semidefinite* if $x^T M x \geq 0$ for all $x \in \mathbb{R}^n$. Similarly, M is *positive definite* if $x^T M x > 0$ for all non-zero x. Recall that a PSD matrix can be decomposed as

$$M = BB^T.$$

1.2.1 Norms

We define the operator norm

$$\|A\|_{\text{op}} := \max_{x \neq 0} \frac{\|Ax\|_2}{\|x\|_2} = \max_i |\lambda_i|.$$

This can be generalized to other norms.

For $A \succeq 0$, we write

$$\|x\|_A := \sqrt{x^T A x}.$$

Recall the dual norm of a normed vector space on the vector space of bounded linear functionals is the operator norm. In the case of ℓ_p norms, linear functionals are uniquely defined by another vector so that the dual norm can be shown to be ℓ_q norm where q is the Lebesgue conjugate of p.

Chapter 2

Graphs & Matrices

2.1 Graphs

Definition 2.1.1 (Cut)
A *cut* is some $F \subseteq E$ such that $(V, E \setminus F)$ is disconnected.

An (s,t)-*cut* is some $F \subseteq E$ such that $(V, E \setminus F)$ has no (s,t)-paths.

Recall that a *spanning tree* is a connected, acyclic subgraph (V, F) where $F \subseteq E$. A special case is a *Hamiltonian path*.

Definition 2.1.2 ((s,t)-Flow)
An (s,t)-flow for an undirected graph G is a function $f : E \to \mathbb{R}$ such that for every $v \neq s, t$,

$$\sum_{e \sim v} f(t) = 0.$$

Note that we can think of f as a function $V \times V \to \mathbb{R}$ that should be *skew-symmetric*. That is,

$$f(u, v) = -f(v, u).$$

2.2 Matrices

Recall that the adjacency matrix $A \in \mathbb{R}^{V \times V}$ of a graph is a 0-1 matrix such that

$$A_{uv} = 1 \iff uv \in E.$$

The *Laplacian* is defined as

$$L = D - A$$

where D is the diagonal degree matrix.

Proposition 2.2.1
$L\mathbb{1} = 0$.

Recall that the *vertex-edge incidence matrix* $B \in \mathbb{R}^{V \times E}$ is the matrix whose columns consists of signed edge vectors

$$b_{uv}[i] = \begin{cases} -1, & i = u \\ 1, & i = v \\ 0, & \text{else} \end{cases}$$

Note that the convention of which vertex being negative is not standard and does not matter much as long as we are consistent.

Proposition 2.2.2
$L = BB^T$.

Proof
BB^T consists of all the inner products of rows vectors a_v. But $a_v^T a_v = \deg(v)$ and for $u \neq v$, $a_u^T a_v$ is just -1 if the edge uv exists and is 0 otherwise.

Corollary 2.2.2.1
If G is connected, the eigenvalue 0 has multiplicity 1.

Corollary 2.2.2.2
The system of linear equations $Lx = b$ has a solution for a connected graph G if and only if $\mathbb{1}^T b = 0$.

2.2.1 $Lx = b$

We wish to determine if we can solve the system of linear questions $Lx = b$ in near linear time $\tilde{O}(m)$. Note that we do not wish to explicitly (pseudo-)invert L since that is an expensive operation.

The idea is first to approximate L as a quadratic function with the Laplacian of a tree. Then, we can perform Gaussian elimination "bottom-up" from the leaves of the tree.

Chapter 3

Convexity

3.1 Convex Sets

Recall that $K \subseteq \mathbb{R}^n$ is said to be *convex* if for every $x, y \in K$ and $\lambda \in [0,1]$,

$$\lambda x + (1-\lambda)y \in K.$$

As a quick refresher of commonly see convex sets, a *hyperplane* is a set of the form

$$K = \{x \in \mathbb{R}^n : \langle a, x\rangle = c\}.$$

A *half-space* is of the form

$$K = \{x \in \mathbb{R}^n : \langle a, x\rangle = c\}.$$

A *polyhedron* is a finite intersection of half-spaces while a *polytope* is a bounded polyhedron. The closed ball of a norm $\|\cdot\|$ is given by

$$K = \{x : \|x - a\| \leq c\}.$$

An *ellipsoid* is a particular example of a closed ball of the form

$$K = \left\{x : \|x-a\|_A = \sqrt{(x-a)^T A(x-a) \leq c}\right\}$$

for some $A \succ 0$.

Proposition 3.1.1
Let $K \subseteq \mathbb{R}^n$ be closed, bounded, and convex. For every $y \in \mathbb{R}^n \setminus K$, there is some $0 \neq h \in \mathbb{R}^n$ for which

$$\langle h, x\rangle < \langle h, y\rangle$$

for every $x \in K$.

Proof
The idea is to project y onto K as

$$x^* := \text{argmin}_{x \in K} \|x - y\|^2$$

and consider $h := y - x^*$. It suffices to show that

$$\langle y - x^*, y - x \rangle > 0$$

for every $x \in K$.

Indeed, for any $x \in K$, we write

$$x_t := tx + (1-t)x^*.$$

We have

$$\begin{aligned} 0 &< \|x^* - y\|_2^2 \\ &\leq \|x_t - y\|_2^2 \\ &= \|x^* - y + t(x - x^*)\|_2^2 \\ &= \|x^* - y\|_2^2 + 2t\langle x^* - y, x - x^* \rangle + t^2 \|x - x^*\|_2^2. \end{aligned}$$

This shows that $\langle x^* - y, x - x^* \rangle = -\langle h, x - x^* \rangle \geq 0$, lest the RHS becomes smaller than the LHS for sufficiently small t. Consequently,

$$\begin{aligned} \langle h, y \rangle - \langle h, x \rangle &\geq \langle h, y \rangle - \langle h, x^* \rangle \\ &= \|h\|_2^2 \\ &> 0 \end{aligned}$$

for all $x \in K$.

3.2 Convex Functions

Recall that a function $f : K \subseteq \mathbb{R}^n \to \mathbb{R}$ where K is convex is said to be *convex* if for every $x, y \in K$ and $\lambda \in [0, 1]$,

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

It is a fact that f is convex if and only if its epigraph

$$\text{epi}(f) := \{(x, y) : f(x) \leq y\}$$

is convex.

Proposition 3.2.1
Suppose $f : K \to \mathbb{R}$ is differentiable. Then f is convex if and only if

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

for all $x, y \in K$.

Intuitively, the proposition above says that the first-order approximation always underestimates $f(y)$.

Definition 3.2.1 (Bregman Divergence)
The *Bregman Divergence* of f at y with respect to x is given by

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

Proof
Suppose f is convex. For any $x, y \in K$ and $\lambda \in [0, 1]$,

$$\begin{aligned}
(1-\lambda)f(x) + \lambda f(y) &\geq f[x + \lambda(y - x)] \\
f(y) &\geq f(x) + \frac{f[x + \lambda(y-x)] - f(x)}{\lambda} \\
f(y) &\geq f(x) + \langle \nabla f(x), y - x \rangle \qquad \lambda \to 0.
\end{aligned}$$

The last step follows since the directional gradient towards $y - x$ is precisely given by $\langle \nabla f(x), y - x \rangle$.

Conversely, take $x, y \in K$ and pick some

$$z := \lambda x + (1 - \lambda) y.$$

We have

$$\begin{aligned}
f(x) &\geq f(z) + \langle \nabla f(z), x - z \rangle \\
f(y) &\geq f(z) + \langle \nabla f(z), y - z \rangle \\
(1-\lambda)f(x) + \lambda f(y) &\geq f(z) + \langle \nabla f(z), z - z \rangle \\
&= f(z).
\end{aligned}$$

Proposition 3.2.2
Suppose $f : K \to \mathbb{R}$ is continously differentiable over K convex. Then f is convex if and only if for all $x, y \in K$,

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

Proof

$(\implies)$ Simply apply the previous characterization twice, sum the inequalities, and rearrange.

$(\impliedby)$ Conversely, suppose the gradient is monotone. For $\lambda \in [0,1]$, we define $x_\lambda := x + \lambda(y-x)$. The integral representation of $g(\lambda) = f(x_\lambda)$ yields

$$\begin{aligned} f(y) &= f(x) + \int_0^1 \langle \nabla f[x + \lambda(y-x)], y - x\rangle d\lambda \\ &= f(x) + \langle \nabla f(x), y - x\rangle + \int_0^1 \langle \nabla f(x_\lambda) - \nabla f(x), y - x\rangle d\lambda \\ &= f(x) + \langle \nabla f(x), y - x\rangle + \int_0^1 \frac{1}{\lambda}\langle \nabla f(x_\lambda) - \nabla f(x), x_\lambda - x\rangle d\lambda \\ &\geq f(x)\langle \nabla f(x), y - x\rangle. \end{aligned}$$

Recall that $v \in \mathbb{R}^n$ is a *subgradient* of $f : K \to \mathbb{R}$ at the point x, written $v \in \partial f(x)$, if for every $y \in K$,

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

It can be shown that $f : K \to \mathbb{R}$ is convex if and only if $\partial f(x) \neq \varnothing$ for all $x \in K$.

Proposition 3.2.3
Suppose $f : K \to \mathbb{R}$ is twice continuously differentiable over some K open and convex. Then f is convex if and only if $\nabla^2 f(x) \succeq 0$ for all $x \in K$.

For instance, the quadratic form defined by a graph Laplacian is convex.

Proof

Suppose f is twice continuously differentiable and convex. Pick any $x \in K$ and note that there is some $\epsilon > 0$ such that $x + \epsilon s \in K$ for all unit vectors $s \in \mathbb{R}^n$. From the Taylor expansion,

$$f(x + \lambda s) = f(x) + \lambda \nabla f(x)^T s + \frac{\lambda^2}{2} s^T \nabla^2 f(x) s + o(\lambda^2)$$

$$\frac{\lambda^2}{2} s^T \nabla^2 f(x) s + o(\lambda^2) \geq 0 \qquad \text{first-order characterization}$$

$$s^T \nabla^2 f(x) s + o(1) \geq 0.$$

Taking the limit as $\lambda \to 0^+$ yields the desired result.

Conversely, suppose that $\nabla^2 f \succeq 0$. From the Taylor expansion of f, there is some $z \in [x, y]$ such that

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

for all $x, y \in K$. If $\nabla^2 f \succ 0$, this implies the first-order characterization of convexity as required.

3.2.1 Stronger Notions of Convexity

Recall that f is strictly convex if Jensen's inequality is satisfied with strictly inequality. It can be shown that f is strictly convex if and only if the first and second order characterizations are satisfied with strict inequality.

Definition 3.2.2 (Strong Convexity)
We say that $f : K \to \mathbb{R}$ is σ-strongly convex with respect to some norm $\|\cdot\|$ if

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

We remark that strong convexity implies strict convexity. Moreover, the notion of strong convexity can be defined for non-differentiable functions through the use of the subgradient. We note that if $\|\cdot\| = \|\cdot\|_2$, then σ-strong is implied by the condition

$$\nabla^2 f \succeq \sigma I.$$

This can be verified using Taylor approximations.

Proposition 3.2.4
The following are equivalent conditions to characterize that a differentiable function f is σ-strongly convex with respect to $\|\cdot\|_2$.

(i) $f(y) \geq f(x) + \nabla f(x)^T (y - x) + \frac{\sigma}{2}\|y - x\|^2$ for all $x, y \in K$

(ii) $g(x) = f(x) - \frac{\sigma}{2}\|x\|^2$ is convex

(iii) $[\nabla f(x) - \nabla f(y)]^T (y - x) \geq \sigma \|x - y\|^2$ for all $x, y \in K$

(iv) $f[\lambda x + (1-\lambda)y] \leq \lambda f(x) + (1-\lambda) f(y) - \frac{\lambda(1-\lambda)}{2}\mu\|x - y\|^2$ for all $x, y \in K, \lambda \in [0, 1]$

Proof
(i) $\iff$ (ii) This follows from the first-order condition for convexity.

$$\begin{aligned}
g(y) &\geq g(x) + \nabla g(x)^T (y - x) \\
f(y) - \frac{\sigma}{2}\|y\|^2 &\geq f(x) - \frac{\sigma}{2}\|x\|^2 + [\nabla f(x) - \sigma x]^T (y - x) \\
f(y) &\geq f(x) + \nabla f(x)^T (y - x) + \frac{\sigma}{2}\|y\|^2 + \frac{\sigma}{2}\|x\|^2 - \sigma x^T y \\
&= f(x) + \nabla f(x)^T (y - x) \frac{\sigma}{2}\|y - x\|^2.
\end{aligned}$$

(ii) $\iff$ (iii) This follows from the monotone gradient condition for the conveixty of g.

(iii) $\iff$ (iv) This follows by expanding Jensen's inequality.

Again, we note that these equivalences hold for non-differentiable functions by replacing the gradient with the subgradient.

Proposition 3.2.5
For a continuously differentiable $f : K \to \mathbb{R}$ that is σ-strongly convex, the following inquality holds for all $x \in K$.

$$\frac{1}{2}\|\nabla f(x)\|^2 \geq \sigma[f(x) - f(x^*))]$$

We remark the inequality above is known as the *Polyak-Lojasiewicz (PL)* condition.

Proof
Consider the inequality

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

and minimize with respect to y. The LHS becomes $f(y^*)$ and the RHS is minimized if and only if

$$\nabla f(x) + \sigma(y - x) = 0.$$

Substituting in the value of y to the RHS yields the inequality.

3.2.2 Optimality Conditions

Suppose our goal is to solve the following optimization problem

$$\begin{aligned} &\min f(x) \\ &x \in K \end{aligned}$$

where f, K are both convex.

Proposition 3.2.6
If f is convex and differentiable, then $\nabla f(x) = 0$ implies that x is a global minimizer.

The proof of the above uses the first-order characterization.

Proposition 3.2.7
If f is differentiable and x is a global minimizer, then $\nabla f(x) = 0$.

Proof
We have

$$\begin{aligned} f(x+tr) &\geq f(x) && \forall r \in \mathbb{R}^n, t \in \mathbb{R} \\ &\leq \lim_{t \to 0} \frac{f(x+tr) - f(x)}{t} && = \langle \boldsymbol{\nabla} f(x), r \rangle \\ &= -\|\boldsymbol{\nabla} f(x)\|^2. && r := -\boldsymbol{\nabla} f(x) \end{aligned}$$

Thus it can only be that $\boldsymbol{\nabla} f(x) = 0$.

Part II

Convex Optimization

Chapter 4

Convex Optimization

We wish to solve problems of the form

$$\begin{aligned} &\min f(x) \\ x &\in K \end{aligned}$$

where f, K are both convex.

An important example of convex programming is linear programming.

$$\begin{aligned} &\min c^T x \\ Ax &= b \\ x &\geq 0 \end{aligned}$$

4.1 Membership & Separation

We begin by remarking that we cannot encode arbitrary real numbers in the Turing machine model. Thus we assume out inputs are rational. Define $L(x)$ to be the number of bits needed to encode a number x.

4.1.1 Membership

We can test for membership of convex sets efficiently, even though we may not be able to write down a succient description. For instead, we can test for membership for ellipsoids, ℓ_1-ball, ℓ_∞-ball, the spanning tree polytope, and PSD done all in polynomial time.

4.1.2 Separation

Separation is a harder problem than membership. We would like to determine if $x \in K$, and if not, provide a "proof" in the form of a hyperplane $h \in \mathbb{R}^n, c \in \mathbb{R}$ that separates x from K.

4.2 Solving Convex Programs

From our discussion of representing real numbers, it is not hard to see that we cannot solve convex programs exactly in general. Thus our goal is to find some $x \in K$ such that

$$f(x^*) \leq f(x) \leq f(x^*) + \varepsilon$$

for some $\varepsilon > 0$. Here x^* is some true optimal solution.

The goal is to obtain this in runtime that is polylog$(1/\varepsilon)$, but it turns out that this is very difficult. Instead, we first attempt to do so in poly$(1/\varepsilon)$ time.

4.3 Encoding a Convex Function

For certain problems, we can explicitly give the function. For example, the least squares problem

$$\min_x \|Ax - b\|_2^2$$

is described by A, b.

Alternative options include

$x \mapsto f(x)$	value oracle
$x \mapsto \nabla f(x)$	1st-order oracle
$(x, v) \mapsto \nabla^2 f(x)v.$	2nd-order oracle

We remark that in the oracle model, we are charged for writing the input and reading the output of the oracle but not for the evaluation of the oracle.

Chapter 5

Gradient Descent

5.1 The Algorithm

5.1.1 Input

We have as input

1) 0, 1-st order access to f
2) $\varepsilon > 0$
3) A constant $L > 0$ such that our problem satisfies some Lipchitz condition (ie Lipschitz gradient)
4) $D > 0, x_0 \in \mathbb{R}^n$ such that $\|x_0 - x^*\| \leq D$

5.1.2 Output

Again, we wish to produce some $x \in K$ such that $f(x) \leq f(x^*) + \varepsilon$.

5.1.3 Strategy

The idea is to iteratively take steps towards a direction which maximizes the "rate of reduction"

$$x_{t+1} \leftarrow x_t + \delta u.$$

Here u is a unit vector and $\delta > 0$ is some step-size.

Now,

$$\begin{aligned}\max_{u\in S^{n-1}} \lim_{\delta\to 0} \frac{f(x)-f(x+\delta u)}{\delta} &= \max_{u\in S^{n-1}} -Df(x)[u] \\ &= \max_{u\in S^{n-1}} \langle \nabla f(x), u\rangle \\ u &= \frac{-\nabla f(x)}{\|\nabla f(x)\|}.\end{aligned}$$

Thus we should take a step in the direction of the negative gradient!

Furthermore, we combine the step-size and normalization constant. Our algorithm is thus given by

$$x_{t+1} \leftarrow x_t - \eta_t \nabla f(x_t).$$

5.2 Analysis

The first question we ask is whether we need additional assumptions. We want the scale of our function to be relatively well-behaved. The following are some options.

1) f is L-Lipschitz.
2) ∇f is L-Lipschitz.
3) $\|\nabla f\| \le G$ for all x.

Remark that 1), 3) are actually equivalent.

Lemma 5.2.1
We have

$$f(y) - [f(x) + \langle \nabla f(x), y-x\rangle] \le \frac{L}{2}\|x-y\|^2.$$

Proof
Let $\lambda \in [0,1]$ and define

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

FTC tells us that

$$\int_0^2 g(\lambda)d\lambda = f(y) - f(x).$$

Moreover, the derivative of g is the directional gradient of f

$$\dot{g}(x) = \langle \nabla f[(1-\lambda)x + \lambda y], y-x\rangle.$$

It follows that

$$
\begin{aligned}
& f(y)-f(x) \\
&= \int_0^1 \langle \nabla f[(1-\lambda)x+\lambda y], y-x\rangle d\lambda \\
&= \int_0^1 \langle \nabla f(x), y-x\rangle d\lambda + \int_0^1 \langle \nabla f[(1-\lambda)x+\lambda y]-\nabla f(x), y-x\rangle d\lambda \\
&\le \langle \nabla f(x), y-x\rangle + \int_0^1 \|\nabla f[(1-\lambda)x+\lambda y]-\nabla f(x)\| \cdot \|y-x\| d\lambda \\
&\le \langle \nabla f(x), y-x\rangle + \int_0^1 L \cdot \|(1-\lambda)x+\lambda y-x\| \cdot \|y-x\| d\lambda \\
&\le \langle \nabla f(x), y-x\rangle + L\|y-x\|^2 \int_0^1 \lambda d\lambda \\
&\le \langle \nabla f(x), y-x\rangle + \frac{L}{2}\|y-x\|^2.
\end{aligned}
$$

Theorem 5.2.2
Given

1) 1st-order access to $f:\mathbb{R}^n \to \mathbb{R}$ convex
2) L such that ∇f is L-Lipschitz
3) Initial point $x_0 \in \mathbb{R}^n$
4) $D>0$ such that $\max\{\|x-x^*\| : f(x) \le f(x_0)\} \le D$
5) $\varepsilon > 0$ (parameter)

Then following holds for gradient descent.

a) Finds some x such that $f(x) \le f(x^*)+\varepsilon$
b) Makes $T = O(DL^2/\varepsilon)$ queries to the oracle
c) Makes $O(nT)$ arithmetic operations

Proof
Let us apply our lemma with $y = x_{t+1}, x = x_t$. We have

$$
\begin{aligned}
f_{t+1} &\le f(x_t) + \langle \nabla f(x_t), x_{t+1}-x_t\rangle + \frac{L}{2}\|x_{t+1}-x_t\|^2 \\
&= f(x_t) + \left(-\eta + \frac{\eta^2 L}{2}\right)\|\nabla f(x)\|^2 \\
&\le f(x_t) - \frac{1}{2L}\|\nabla f(x_t)\|_2^2. \qquad \eta = \frac{1}{L}
\end{aligned}
$$

We now employ convexity. Suppose $f(x_t) - f(x^*) \geq \varepsilon$. Then

$$\begin{aligned}\varepsilon &\leq f(x_t) - f(x^*) \\ &\leq \langle \nabla f(x_t), x_t - x^* \rangle \\ &\leq \|\nabla f(x_t)\| \cdot \|x_t - x^*\| \\ &\leq D \cdot \|\nabla f(x_t)\|.\end{aligned}$$

Hence we have a lower bound ε/D on the gradient norm.

To shrink the objective gap from $2\varepsilon \to \epsilon$, this requires $2D^2L/\varepsilon$ steps. Thus to go from the initial point to ε, the number of steps is a geometric sum and is thus dominated by the last step $2\varepsilon \to \varepsilon$.

Since each iteration requires $O(n)$ arithmetic operations, this concludes the proof.

5.3 Handling Constraints

Suppose now we wish to optimize $f : K \to \mathbb{R}$ for some convex K. The only change in algorithm for gradient descent is an additiona projection step

$$x_{t+1} = \text{Proj}_K[x_t - \eta \nabla f(x_t)].$$

The exact same analysis holds for projected GD since projections onto closed convex sets are contractions.

5.4 Maximum Flow

5.4.1 The Problem

We are given as input $G = (V, E)$ undirected and we wish to find an $s - t$ *flow*, ie $x \in \mathbb{R}^E, F > 0$ such that

$$Bx = F \cdot [1, \dots 0, \dots, -1]^T$$

Here the target vector has 1 in the s-th coordinate and -1 kin the t-th coordinate and we say x has *value* F.

We also constrain each to have some capacity $|x_e| \leq 1$. Note that it is possible to handle general capacities but the canonical case of all 1's illustrates the most ideas.

Now, we will assume that G is connected so there is some $s - t$ path, thus the optimal value $F^* \geq 1$. Trivially, $F^* \leq m$ as well.

5.4.2 Gradient Descent

By guessing $O(\log m)$ values, we may assume without loss of generality that F^* is know to us. This then becomes a feasibility problem $Bx = Fb$ and $\|x\|_\infty \leq 1$. Let P denote the projection onto the ℓ_∞ ball. Consider the following minimization problem.

$$\begin{aligned} &\min \|x - P(x)\|^2 \\ Bx &= Fb \end{aligned}$$

We wish to gradient descent. This requires us to check several conditions.

1. Convexity
2. $\boldsymbol{\nabla} f$ is L-Lipschitz
3. Complexity of projection
4. x_0 with small D

Convexity

We first observe that the objective $f = \sum_{i=1}^n f_i(x_i)$ is separable. Hence it suffices to check each f_i is convex. Indeed,

$$f_i(x_i) = \begin{cases} 0, & x_i \in [-1, 1] \\ (x_i - 1)^2 & x_i > 1 \\ (x_i + 1)^2 & x_i < -1 \end{cases}$$

Since the Hessian is PSD, f_i and hence f is certainly convex.

Lipschit Gradient

Since the Hessian is bounded above by 2, we see that the gradient is L-Lipschitz for $L = 2$.

Projection

Now, suppose $x_t \in \{y : Bx = Fb\}$. Then

$$\{y : By = Fb\} = x_t + \ker B.$$

Hence

$$\text{Proj}(x_t + \eta \boldsymbol{\nabla} f(x_t)) = x_t + \text{Proj}_{\ker B}(\boldsymbol{\nabla} f(x_t)).$$

From elementary linear algebra, the kernel is the orthogonal complement of the row space. Hence if we write

$$P := B^T(BB^T)^\dagger B$$

as the projection onto the row space of B, then $I - P$ is the projection onto $\ker B$.

While naive computation of $(I-P)\boldsymbol{\nabla} f$ can be expensive, projection onto the row space can be written as

$$\min_y \left\| B^T y - \boldsymbol{\nabla} f \right\|,$$

from which the normal equations tell us to solve

$$BB^T y = B\boldsymbol{\nabla} f.$$

Since BB^T is just the Laplacian, we can do this in $\tilde{O}(m)$ time.

Initial Point

Consider taking $x_0 := \text{Proj}_{\{x:Bx=Fb\}}(0)$, again using the Laplacian solver. We know that the Euclidean radius of the ℓ_∞ ball is $\sqrt{m}$. Hence if $\|x_0\| > \sqrt{m}$, then we should halve our guess of F since no such flow exists. On the other hand, all feasible flows has radius at most $\sqrt{m}$. Thus all in all, we have

$$D \leq 2\sqrt{m}.$$

Summary

Theorem 5.4.1
We can find an $s-t$ flow x such that

$$\|x - P(x)\|_2^2 \leq \varepsilon$$

in time $\tilde{O}(m^2/\varepsilon)$.

Note that in order to derive a truly feasible flow, we need to complete some rounding step. This can also be done.

Chapter 6

Multiplicative Weights Update Method

6.1 Motivation

Consider the following stock bidding game. Every day, we are asked to bid whether a certain stock goes up or down, given access to predictions of some experts.

We assume that there is at least one expert whose predictions are "good". Define m_i^t as the number of mistakes the i-th expert makes up to and including day t and M^t as the number of mistakes we make up to time t. Recall the *regret* is defined as

$$R_t := M^t - \min_i m_i^t.$$

The goal is to obtain an algorithm which minimizes regret.

6.2 Weighted Majority Method

The idea is to keep a collection of confidence scores $w_i^t \geq 0$ for each expert starting from $w_i^0 = 1$ for all i. Then, we proceed with the weighted majority of experts. Say we observe an outcome

$$f_i^t := \begin{cases} 1, & \text{expert } i \text{ is wrong at time } t \\ 0, & \text{else} \end{cases}$$

We would make the update

$$w_i^{t+1} = w_i^t(1 - \varepsilon f_i^t).$$

6.2.1 Analysis

The key to the analysis is the potential function

$$\phi^t := \sum_{i=1}^{n} w_i^t.$$

If we made a mistake at time t, say 1 was the wrong choice, then the weight of the majority exceeds half of the potential function

$$\sum_{i \to +1} w_i^t \geq \frac{1}{2}\phi^t.$$

It follows that

$$\begin{aligned}
\phi^{t+1} &= \sum_{i=1}^{n} w_i^t (1 - \varepsilon f_i^t) \\
&= \sum_{i=1}^{n} w_i^t - \varepsilon \sum_{i=1}^{n} w_i^t f_i^t \\
&= \phi^t \left(1 - \frac{\varepsilon}{\phi^t} \sum_{i=1}^{n} w_i^t f_i^t\right).
\end{aligned}$$

Thus in the case we made a mistake,

$$\phi^{t+1} \leq \phi^t \left(1 - \frac{\varepsilon}{2}\right).$$

Now, $\phi^0 = n$ and we make M^t mistakes up to time t, so

$$\phi^{t+1} \leq n \left(1 - \frac{\varepsilon}{2}\right)^{M^t}.$$

On the other hand

$$\phi^{t+1} \geq w_i^t = (1 - \varepsilon)^{m_i^t}$$

for all i. It follows that for every $i \in [n]$,

$$\begin{aligned}
(1 - \varepsilon)^{m_i^t} &\leq n \left(1 - \frac{\varepsilon}{2}\right)^{M^t} \\
&\leq n \exp\left(-\frac{\varepsilon M^t}{2}\right) && 1 - x \leq e^{-x} \\
m_i^t(-\varepsilon - \varepsilon^2) &\leq m_i^t \log(1 - \varepsilon) && \forall x \in [0, 1/2], -x - x^2 \leq \log(1 - x) \\
&\leq \log n - \frac{\varepsilon M^t}{2} \\
M^t - 2m_i^t(1 + \varepsilon) &\leq \frac{2 \log n}{\varepsilon} \\
\frac{M^T}{T} - \frac{2m_i^T}{T}(1 + \varepsilon) &\leq \frac{2 \log n}{\varepsilon T}.
\end{aligned}$$

Theorem 6.2.1
For

$$T \geq \frac{2 \log n}{\varepsilon^2},$$

we have

$$\frac{1}{T} M^T - 2(1+\varepsilon)\frac{1}{T} m_i^T \leq \varepsilon$$

for all $i \in [n]$.

6.3 Multiplicative Weights Update Method

Suppose now that our outcome vectors $f^t = (f_1^t, \ldots, f_n^t)$ satisfy

$$\left\| f^t \right\|_\infty \leq 1.$$

We now play a probability vector $p^t \in \Delta^n$ and observe a payoff

$$\langle p^t, f^t \rangle.$$

Our goal remains to minimize regret

$$\frac{1}{T} \sum_{t=0}^{T-1} \langle p^t, f^t \rangle - \frac{1}{T} \min_{p \in \Delta^n} \sum_{t=0}^{T-1} \langle p, f^t \rangle.$$

Note that we can view $f^t = \boldsymbol{\nabla} F$ for some function F whose gradient is bounded. This technique generalizes to the online convex optimization setting where if we play p^t, we observe

$$\boldsymbol{\nabla} F^t(p^t) = f^t.$$

6.3.1 The Algorithm

1) Initialize $w_i^0 = 1$ for all $i \in [n]$.
2) Set $p_i^t \leftarrow \frac{w_i^t}{\mathbb{1}^T w^t} = \frac{w_i^t}{\phi^t}$.
3) Play p^t.
4) Incur loss $\langle p^t, f^t \rangle$.
5) Update $w_i^{t_1} \leftarrow w_i^t (1 - \varepsilon f_i^t)$.

6.3.2 Analysis

Once again, we have the trivial power bound

$$\begin{aligned}\phi^{t+1} &\geq w_i^t(1-\varepsilon f_i^t) \\ &\geq w_i^t \exp\bigl[-\varepsilon f_i^t - (\varepsilon f_i^t)^2\bigr]. && \varepsilon \in [0, 1/2] \\ &\geq 1 \cdot \exp\left[-\varepsilon \sum_{\tau=0}^{t} f_i^\tau - \varepsilon^2 \sum_{\tau=0}^{t} (f_i^\tau)^2\right].\end{aligned}$$

We also have a similar but more general upper bound

$$\begin{aligned}\phi^{t+1} &= \sum_{i=1}^{n} w_i^t(1-\varepsilon f_i^t) \\ &= \phi^t - \varepsilon \sum_{i=1}^{n} f_i^t w_i^t \\ &= \phi^t - \varepsilon \sum_{i=1}^{n} f_i^t \phi^t p_i^t \\ &= \phi^t \left(1 - \varepsilon \langle f^t, p^t \rangle\right) \\ &\leq \phi^t \exp\bigl[-\varepsilon \langle f^t, p^t \rangle\bigr] \\ &\leq n \cdot \exp\left[-\varepsilon \sum_{\tau=0}^{t} \langle f^t, p^t \rangle\right].\end{aligned}$$

Combining these two inequalities, taking logarithms, and rearranging yields

$$\begin{aligned}-\varepsilon \sum_{t=0}^{T-1} f_i^t - \varepsilon^2 \sum_{t=0}^{T-1} (f_i^t)^2 &\leq \log n - \varepsilon \sum_{t=0}^{T-1} \langle f^t, p^t \rangle \\ \frac{1}{T} \sum_{t=0}^{T-1} \langle f^t, p^t \rangle - \frac{1}{T} \sum_{t=0}^{T-1} f_i^t &\leq \frac{\log n}{\varepsilon T} + \varepsilon \frac{1}{T} \sum_{t=0}^{t} (f_i^t)^2.\end{aligned}$$

Theorem 6.3.1
For every $i \in [n]$,

$$\frac{1}{T}\sum_{t=0}^{T-1}\langle p^t, f^t\rangle - \frac{1}{T}\sum_{t=0}^{T-1} f_i^t \le \frac{\ln n}{\varepsilon T} + \varepsilon$$
$$\frac{1}{T}\sum_{t=0}^{T-1}\langle p^t, f^t\rangle - \min_{p\in\Delta^n}\frac{1}{T}\left\langle \sum_{t=0}^{T-1} f^t, p\right\rangle \le \frac{\ln n}{\varepsilon T} + \varepsilon.$$

In particular, if $T = \frac{\ln n}{\varepsilon^2}$ implies the average regret is at most 2ε.

6.4 Perfect Bipartite Matching

We wish to find some $x \in \mathbb{R}^m$ such that

$$\begin{aligned} \sum_e x_e &= n \\ \sum_{e\in N(v)} x_v &\le 1 && \forall v \in V \\ x_e &\ge 0 && \forall e \in E \end{aligned}$$

Note that this is justified since the polytope is integral. There is also a way to found any fractional solutions to integral solutions in $\tilde{O}(m)$ time, hence we focus on finding a fractional perfect matching as above.

In fact, we relax the "difficult" constraints ever so slightly

$$\begin{aligned} \sum_e x_e &= n \\ \sum_{e\in N(v)} x_v &\le 1+\varepsilon && \forall v \in V \\ x_e &\ge 0 && \forall e \in E \end{aligned}$$

6.4.1 The Algorithm

The idea is to combine the $2n$ inequalities into a single weighted inequality, say w_v^t denotes the "importance" of the v-th inequality.

Initialize $w^0 \equiv 1$. Then for $t = 0, \dots, T-1$:

1) Find x^t satisfying

(a) $\sum_v w_v^t \sum_{e \in N(v)} x_e^t \leq \sum_{v \in V} w_v^t$

(b) $\sum_{e \in E} x_e^t = n$

(c) $x^t \geq 0$

2) Set $f_v^t := \frac{1 - \sum_{e \in N(v)} x_e^t}{n}$ for all $v \in V$.

3) Update $w_v^{t+1} := w_v^t(1 - \eta f_v^t)$ for all $v \in V$.

Finally, output

$$x = \frac{1}{T} \sum_{t=0}^{T-1} x^t.$$

6.4.2 Analysis

We would like to call upon the guarantees for the MWU method but this requires us to check some properties.

Lemma 6.4.1
If G has a perfect matching, then

1) x^t always exists and can be found in $\tilde{O}(m)$ time.

2) $\|f^t\|_\infty \leq 1$.

Proof
Define

$$\alpha_e := \sum_{v \sim e} w_v^t$$

$$\sum_v w_v^t := \beta.$$

We would like to find x^t such that

$$\sum_e \alpha_e x_e^t \leq \beta.$$

If G has a perfect matching, say $e^1, \ldots, e^n$, then

$$\sum_{i=1}^n \alpha_{e^i} = \beta.$$

Take

$$e^* := \text{argmin}_e \, \alpha_e$$

so that

$$\begin{aligned} n\alpha_{e^*} &\le n \min_i \alpha_{e^i} \\ &\le \sum_i \alpha_{e^i} \\ &= \beta. \end{aligned}$$

Set $x^t_{e^*} := n$ and $x^t_e = 0$ for all $e \neq e^*$. By construction,

$$\sum_{v \in V} w^t_v \sum_{e \in N(v)} x^t_e \le \sum_{v \in V} w^t_v.$$

In addition,

$$\frac{1-n}{n} \le f^t_v = \frac{1 - \sum_{e \in N(v)} x^t_e}{n} \le 1.$$

We can now proceed to apply the MWU guarantee to show that $x := \frac{1}{T}\sum_{t=0}^{T-1} x^t$ is an ε-approximate fractional perfect matching in G. First, since each x^t satisfies non-negativity and $\sum_v x^t_v = n$, so does the average among x^t's. It remains to check the "difficult" constraints.

Indeed, we have by construction

$$\sum_{v \in V} w^t_v \sum_{e \in N(v)} x^t_e \le \sum_{v \in V} w^t_v$$

$$\sum_{v \in V} w^t_v \left(1 - \sum_{e \in N(v)} x^t_e\right) \ge 0$$

Thus our intermediaries satisfy

$$\langle p^t, f^t \rangle \ge 0.$$

It follows that for $T \ge \frac{\ln n}{\eta^2}$ and any $v \in V$,

$$\begin{aligned} 2\eta &\ge \frac{1}{T}\sum_t \langle p^t, f^t \rangle - \frac{1}{T}\sum_t f^t_v \\ &\ge -\frac{1}{T}\sum_t f^t_v \\ &= -\frac{1}{T}\sum_t \frac{1 - \sum_{e \in N(v)} x^t_e}{n}. \end{aligned}$$

$$\begin{aligned} \sum_{e \in N(v)} \frac{1}{T}\sum_t x^t_e &\le 1 + 2n\eta \\ &\le 1 + \varepsilon. \qquad\qquad \eta := \frac{\varepsilon}{2n} \end{aligned}$$

Theorem 6.4.2
There is an algorithm which takes as input a bipartite graph with a perfect matching on $2n$ vertices, m edges, a parameter $\varepsilon > 0$, and outputs an ε-approximate fractional solution in time

$$O\left(\frac{n^2 m}{\varepsilon^2}\right).$$

Intuitively, the algorithm greedily selects a single edge to add to the perfect matching at every iteration by looking at the "least violated" constraints related to that edge.

Chapter 7

Newton's Method

7.1 Newton's Method

Consider the following simple problem in 1 dimension. Given $g : \mathbb{R}^{\rightarrow}\mathbb{R}$, find r such that $g(r) = 0$.

An intuitive approach is to take a first-order approximation at some x_0, and solve for a root x_1 of the first-order approximation.

$$0 = g(x_0) + (x_1 - x_0)g'(x_0).$$

Theorem 7.1.1
If g is twice continuously differentiable and r is a root of g, then

$$|x_1 - r| \leq M|x_0 - r|^2$$

where

$$M := \sup_{\xi \in (r, x_0)} \left| \frac{g''(\xi)}{2g'(x_0)} \right|.$$

Proof
By the mean value theorem, there is some $\xi \in (r, x_0)$ such that

$$g(r) = g(x_0) + (r - x_0)g'(x_0) + \frac{1}{2}(r - x_0)^2 g''(\xi).$$

Since $0 = g(x_0) + g'(x_0)(x_1 - x_0)$ by construction, it follows that

$$\begin{aligned} 0 &= g(r) \\ &= g'(x_0)(x_0 - x_1) + (r - x_0)g'(x_0) + \frac{1}{2}(r - x_0)^2 g''(\xi) \\ (r - x_1)g'(x_0) &= -\frac{1}{2}(r - x_0)^2 g''(\xi) \\ |x_1 - r| &= |x_0 - r|^2 \cdot \left| \frac{g''(\xi)}{2g'(x_0)} \right| \\ &\leq M|x_0 - r|^2. \end{aligned}$$

7.2 Newton's Method for Optimization

Recall for convex functions, optimization is equivalent to root finding for the derivative. Thus if we wish to optimize $f : \mathbb{R}^d \to \mathbb{R}$ convex, we can attempt to generalize Newton's method for finding x such that $\nabla f(x) = 0$. This requires us to generalize Newton's method for $g : \mathbb{R}^d \to \mathbb{R}^d$.

Moving forward, we assume that f is strictly convex and twice continuously differntiable. The straightforward generalization is thus given by

$$x_1 = x_0 - H(x_0)^{-1} \cdot \nabla f(x_0).$$

Note that strict convexity ensures $H(x_0) \succ 0$ and the inverse always exists.

7.2.1 Interpretation

One interpretation of this method is by considering the second-order approximation and explicitly solving for a root. This is simple and natural.

We will consider a different view which is not as straightforward to derive but yields a nice way to measure the progress when we iteratively apply Newton's method. Specifically, we consider the norm induced by this local inner product

$$\langle u, v \rangle_g := u^T g(x) v.$$

When g is the Hessian of a convex function, this is known as the *Einstein(-Hessian) metric*. It turns out that performing "gradient descent" with respect to $H(x)$, we recover the update rule

$$x_1 = x_0 + n(x_0)$$

where $n(x_0) := -H(x_0)^{-1} \cdot \nabla f(x_0)$ is the *Newton step*.

7.2.2 Analysis

In order to facilitate the analysis, we introduce a non-standard definition.

Definition 7.2.1 (Newton-Local Condition)
We say that $f : \mathbb{R}^d \to \mathbb{R}$ satisfies the *Newton-Local (NL) condition* for parameter $\delta_0 \in (0, 1)$ if for all $\delta \in (0, \delta_0]$, any $x, y \in \mathbb{R}^d$ satisfying $\|x - y\|_x \leq \delta$ also satisfies

$$(1 - 3\delta)H(x) \preceq H(y) \preceq (1 + 3\delta)H(x).$$

Lemma 7.2.1
Assume f is strictly convex and satisfies the NL condition for $\delta_0 = 1/6$. Let $x, y \in \mathbb{R}^n$ be such that $\|x - y\|_x \leq 1/6$. Then for every $u \in \mathbb{R}^n$,

1) $\frac{1}{2}\|u\|_x \leq \|u\|_y \leq 2\|u\|_x$
2) $\frac{1}{2}\|u\|_{H(x)^{-1}} \leq \|u\|_{H(y)^{-1}} \leq 2\|u\|_{H(x)^{-1}}$

Lemma 7.2.2
Suppose $A \succeq 0, B \in \S^d$ satisfies $-\alpha A \preceq B \preceq \alpha A$ for some $\alpha \geq 0$. Then

$$\left\| A^{-\frac{1}{2}} B A^{-\frac{1}{2}} \right\|_{\text{op}} \leq \alpha.$$

Theorem 7.2.3
If f is strictly convex and satisfies the NL condition for $\delta_0 = 1/6$, then

$$\|n(x_1)\|_{x_1} \leq 3\|n(x_0)\|_{x_0}^2$$

for every $x_0 \in \mathbb{R}^n$ such that $\|n(x_0)\|_{x_0} \leq 1/6$. Here $x_1 = x_0 + n(x_0)$.

Proof
Let $x_0 \in \mathbb{R}^n$ be such that $\|n(x_0)\|_{x_0} \leq 1/6$. We wish to show that

$$\|n(x_1)\|_{x_1} \leq 2\|n(x_0)\|_{x_0}^2$$

where $n(x_0) = -H(x_0)^{-1} \cdot \nabla f(x_0)$.

First observe that

$$\begin{aligned}\|n(x_0)\|_{x_0} &= \sqrt{\nabla f(x_0)^T H(x_0)^{-1} H(x_0) H(x_0)^{-1} \nabla f(x_0)} \\ &= \sqrt{\nabla f(x_0)^T H(x_0)^{-1} \nabla f(x_0)} \\ &= \|\nabla f(x_0)\|_{H(x_0)^{-1}}.\end{aligned}$$

We claim that

$$\|\nabla f(x_1)\|_{H^{-1}(x_0)} \leq \frac{3}{2} \|\nabla f(x_0)\|^2_{H(x_0)^{-1}}.$$

If this holds, we can then apply our lemma above to conclude the result.

To show the claim, we first apply the fundamental theorem of calculus to see that

$$\begin{aligned}&\nabla f(x_1) \\ &= \nabla f(x_0) + \int_0^1 H[x_0 + t(x_1 - x_0)](x_1 - x_0) dt \\ &= \left(H(x_0) - \int_0^1 H[x_0 + t(x_1 - x_0)] dt \right) H(x_0)^{-1} \nabla f(x_0) \\ &\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad x_1 - x_0 = H(x_0)^{-1} \nabla f(x_0) \\ &=: M(x_0) H(x_0)^{-1} \nabla f(x_0).\end{aligned}$$

Taking norms yields

$$\begin{aligned}\|\nabla f(x_1)\|_{H(x_0)^{-1}} &= \left\| M(x_0) H(x_0)^{-1} \nabla f(x_0) \right\|_{H(x_0)^{-1}} \\ &\leq \left\| H(x_0)^{-\frac{1}{2}} M(x_0) H(x_0)^{-\frac{1}{2}} \right\|_{\text{op}} \cdot \|\nabla f(x_0)\|_{H(x_0)^{-1}}. \quad \text{Cauchy-Schwartz}\end{aligned}$$

If remains only to show that the operator norm of the first term above is bounded above by $3/2 \|\nabla f(x_0)\|_{H(x_0)^{-1}}$. In order to do so, we would like to call upon the lemma above. This requires demonstrating that

$$\begin{aligned}-\frac{3}{2} \delta H(x_0) \preceq M(x_0) \preceq \frac{3}{2} \delta H(x_0) \\ \delta := \|\nabla f(x_0)\|_{H(x_0)^{-1}}.\end{aligned}$$

Note that $\delta \leq 1/6$ by assumption.

It suffices to demonstrate that for every $t \in [0, 1]$,

$$-3\delta t H(x_0) \preceq H(x_0) - H[x_0 - t(x_1 - x_0)] \preceq 3t\delta H(x_0).$$

We can then integrate with respect to t to conclude the proof. This follows by the NL condition.

Chapter 8

Interior Point Methods

8.1 Linear Programming

Let $A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m, c \in \mathbb{R}^n$ where $m > n$. Define

$$K := \{x \in \mathbb{R}^n : Ax \leq b\}.$$

We wish to minimize $\langle c, x \rangle$ over all $x \in K$. In particular, we assume that K is bounded and full-dimensional (contains an open ball). The condition that $m > n$ essentially encodes the assumption that there are more constraints than variables.

Note that in terms of computation, A, b, c are actually inputs over $\mathbb{Q}$. Thus if L is the number of bits needed to encode the input, a polynomial-time algorithm means poly(L).

Recall from elementary linear programming that there is always an optimal extreme point solution. In particular, there is an optimal solution that can be decribed with n tight inequalities which is $O(L)$ bits.

8.2 Full-Dimensional IPM

8.2.1 Intuition

We wish to turn this contrained optimization problem into an unconstrained one. Consider something of the form

$$\min \eta \langle c, x \rangle + F(x).$$

Definition 8.2.1 (Barrier Function)
Suppose $F : \text{int}(K) \to \mathbb{R}$ is strictly convex and $F(x) \to \infty$ as $x \to \partial K$. F is known as a *barrier function*

Specifically, we can take

$$F(x) := \sum_{i=1}^{m} \log(b_i - \langle a_i, x \rangle)$$

where a_i is the i-th row of A.

Intuitively, as $\eta \to \infty$, the weight of the $\langle c, x \rangle$ in the objective becomes more and more important. Define

$$f_\eta := \eta \langle x, c \rangle + F(x)$$
$$x_\eta^* = \text{argmin}_x \, f_\eta(x).$$

Then x_0^* is *analytic center* of K.

The idea is to start from some $x_0 \approx x_0^*$. Then since $x_0 \approx x_\eta^*$ for η sufficiently small, taking a Newton step will result in x_1 being even closer to x_η^*. Hence we approximately follow the *central path* described by the map $\eta \mapsto x_\eta^*$. At sufficiently small precision, we can terminate the algorithm. Note that in the case of linear programming, once we are sufficiently close, we can round the solution to the nearest vertex to yield an exact optimal solution.

8.2.2 Analysis

First we remind ourselves of the following notation.

$$f_\eta(x) := \eta \langle c, x \rangle + F(x)$$
$$\boldsymbol{\nabla}^2 f_\eta = \boldsymbol{\nabla}^2 F$$
$$x_\eta^* := \text{argmin}_x \, f_\eta(x)$$
$$n_\eta(x) := H^{-1}(x) \boldsymbol{\nabla} f_\eta(x).$$

The algorithm is as follows.

1) Initialize η_0, x_0 such that $\|n_{\eta_0}(x_0)\|_{x_0} \leq 1/6$.
2) Set T such that $\eta_T = \eta_0 \left(1 + \frac{1}{20\sqrt{m}}\right)^T > m/\varepsilon$.
3) For $t = 1, \ldots, T$:
   i) $x_{t+1} \leftarrow x_t + n_{\eta_t}(x_t)$
   ii) $\eta_{t+1} = \eta_t \left(1 + \frac{1}{20\sqrt{m}}\right)$

4) Output $\hat{x}$ which is obtained from x_T by running two more Newton steps.

Lemma 8.2.1
For all x,

$$\|n_{\eta'}(x)\|_x \leq \frac{\eta'}{\eta}\|n_\eta(x)\|_x + \left|\frac{\eta'}{\eta} - 1\right|\sqrt{m}.$$

Proof
By definition,

$$\begin{aligned}
-n_{\eta'}(x) &= H^{-1}(x)\nabla f_{\eta'}(x) \\
&= H^{-1}(x)[\eta' c + g(x)] && g(x) = \nabla F(x) \\
&= \frac{\eta'}{\eta}H^{-1}(x)[\eta c + g(x)] + \left(1 - \frac{\eta'}{\eta}\right)H^{-1}(x)g(x) \\
&= \frac{\eta'}{\eta}H^{-1}(x)\nabla f_\eta(x) + \left(1 - \frac{\eta'}{\eta}\right)H^{-1}(x)g(x).
\end{aligned}$$

Taking norms on both sides and applying the triangle inequality yields

$$\begin{aligned}
\|n_{\eta'}(x)\|_x &\leq \frac{\eta'}{\eta}\left\|H^{-1}(x)\nabla f_\eta(x)\right\|_x + \left|1 - \frac{\eta'}{\eta}\right| \cdot \left\|H^{-1}(x)g(x)\right\|_x \\
&= \frac{\eta'}{\eta}\|n_\eta(x)\|_x + \left|1 - \frac{\eta'}{\eta}\right| \cdot \left\|H^{-1}(x)g(x)\right\|_x.
\end{aligned}$$

If we show that $\|H^{-1}(x)g(x)\|_x \leq \sqrt{m}$ for all $x \in \operatorname{int} K$, we are done.

Indeed,

$$\begin{aligned}
\left\|H^{-1}(x)g(x)\right\|_x^2 &= \|z\|_x^2 \\
&= g(x)^T H^{-1}(x) H(x) H^{-1}(x) g(x) \\
&= g(x)^T H^{-1}(x) g(x) \\
&= \langle z, g(x)\rangle \\
&= \sum_{i=1}^{m} \frac{\langle z, a_i\rangle}{s_i(x)} \\
&= \mathbb{1}^T \left(\langle z, a_i\rangle s_i(x) : i \in [m]\right) \\
&\leq \sqrt{m} \cdot \sqrt{\sum_{i=1}^{m} \frac{\langle z, a_i\rangle^2}{s_i(x)^2}} \\
&= \sqrt{m} \cdot \sqrt{z^T \sum_{i=1}^{m} \frac{a_i a_i^T}{s_i(x)^2} z} \\
&= \sqrt{m} \cdot \|z\|_x.
\end{aligned}$$

Cancelling the $\|z\|_x$ term on both sides concludes the proof.

Lemma 8.2.2
For every $t = 0, 1, \ldots, T$,

$$\|n_{\eta_t}(x_t)\|_{x_t} \leq \frac{1}{6}.$$

Proof
For $t = 0$, the statement holds by assumption. We now argue by induction.

From the previous lecture, we are guaranteed that

$$\|n_{\eta_t}(x_{t+1})\|_{x_{t+1}} \leq 3\left(\frac{1}{6}\right)^2 = \frac{1}{12}$$

assuming that f_{η_t} satisfies the NL condition. Note that we would still need to show that

$$\left\|n_{\eta_{t+1}}(x_{t+1})\right\|_{x_{t+1}} \leq \frac{1}{6}.$$

However, by the choice of η_{t+1}, we have $\eta_{t+1}/\eta_t = 1 + 1/20\sqrt{m}$, so the lemma above applies

to yield

$$\begin{aligned}\left\|n_{\eta_{t+1}}(x_{t+1})\right\|_{x_{t+1}} &\le \left(1+\frac{1}{20\sqrt{m}}\right)\left\|n_{\eta_t}(x_{t+1})\right\|_{x_{t+1}} + \frac{1}{20\sqrt{m}}\sqrt{m} \\ &\le \frac{21}{20}\cdot\frac{1}{12}+\frac{1}{20} \\ &\le \frac{1}{6}.\end{aligned}$$

Now, consider the NL condition for $\nabla^2 f_\eta(x) = \nabla^2 F$. Let x, y be such that $\|x-y\|_x =: \delta \le 1/6$. We have

$$\nabla^2 F(x) = \sum_{i=1}^m \frac{1}{s_i(x)} a_i a_i^T$$

where $s_i(x) := b_i - \langle a_i, x\rangle$ is the i-th slack function.

By computation,

$$\begin{aligned}\delta^2 &= \|x-y\|_x^2 \\ &= (x-y)^T H(x)(x-y) \\ &= (x-y)^T \sum_{i=1}^m \frac{a_i a_i^T}{s_i(x)^2}(x-y) \\ &= \sum_{i=1}^m \frac{\left(\langle a_i, x\rangle - \langle a_i, y\rangle\right)^2}{s_i(x)^2} \\ &= \sum_{i=1}^m \left|\frac{s_i(x)-s_i(y)}{s_i(x)}\right|^2.\end{aligned}$$

Thus for every i,

$$\begin{aligned}\left|\frac{s_i(x)-s_i(y)}{s_i(x)}\right|^2 &\le \delta^2 \\ |s_i(x)-s_i(y)| &\le \delta s_i(x) \\ (1-\delta)s_i(x) \le s_i(y) &\le (1+\delta)s_i(x) \\ \frac{(1-\delta)^2}{s_i(x)^2} \le \frac{1}{s_i(y)^2} &\le \frac{(1+\delta)^2}{s_i(x)^2} \\ \frac{(1-\delta)^2}{s_i(x)^2} a_i a_i^T \le \frac{1}{s_i(y)^2} a_i a_i^T &\le \frac{(1+\delta)^2}{s_i(x)^2} a_i a_i^T.\end{aligned}$$

Summing over all i yields the inequality

$$(1-\delta)^2 H(x) \preceq H(y) \preceq (1+\delta)^2 H(x).$$

This also holds for $(1 \pm 3\delta)$ instead of $(1 \pm \delta)$ assuming that $\delta \le 1/6$.

8.2.3 Initialization

In order for our algorithm to work, we require an initial point x_0 such that

$$\|n_{\eta_0}(x_0)\|_{x_0} \leq \frac{1}{6}.$$

Suppose we have any point x_0. We can construct an objective c' so that x_0 is close to the central path for the objective c'. Then we can "reverse" the algorithm $\eta \to 0$ so that $x'_t \to x_0^*$ and we arrive at a point close to the analytic center.

8.3 Subspace IPM

Recall that reduced a constrained optimization problem to an unconstrained optimization problem through the barrier function.

$$f_\eta(x) := \eta f(x) + F(x).$$

Before, we considered full-dimensional linear programs. We now consider LPs of the form

$$\begin{aligned} &\min\langle c, x\rangle \\ Ax &= b \\ x &\geq 0 \end{aligned}$$

We need a sense of a derivative within a subspace (manifold)

$$E_b := \{x : Ax = b\}.$$

Here, the tangent space, or intuitively, the directions in which we can take an infinitessimal step is

$$E := \{y : Ay = 0\} = \ker A.$$

Recall the *orthogonal projection* Π_E onto E is given by

$$\Pi_E = I - A^T(AA^T)^{-1}A \in \mathbb{R}^{m\times m}.$$

Note that $\Pi_E^2 = \Pi_E$ and $\Pi_E = \Pi_E^T$.

The Newton step analogue is then given by

$$\tilde{n}(x) = -\tilde{H}^{-1}(x)\tilde{\boldsymbol{\nabla}} f_\eta(x)$$

where

$$\tilde{H}(x) = \Pi_E H(x) \Pi_E^T$$
$$\tilde{\boldsymbol{\nabla}} f_\eta(x) = \Pi_E f_\eta(x).$$

The barrier function for non-negativity constraints is simply

$$F(x) = -\sum_{i=1}^{m} \log x_i.$$

Then, using the notation $X := \mathrm{Diag}(x)$,

$$\boldsymbol{\nabla} F(x) = -X^{-1}\mathbb{1}$$
$$\boldsymbol{\nabla}^2 F(x) = X^{-2}.$$

So

$$\begin{aligned} f_\eta(x) &= \eta\langle c, x\rangle + F(x) \\ \tilde{\boldsymbol{\nabla}} f_\eta(x) &= \eta\tilde{\boldsymbol{\nabla}}\langle c, x\rangle + \tilde{\boldsymbol{\nabla}} F(x) \\ &= \Pi_E(\eta c - X^{-1}\mathbb{1}). \end{aligned}$$

Similarly we can compute $\tilde{H}^{-1}(x)$.

Theorem 8.3.1
There is an IPM taking Newton steps in E which minimizes $\langle c, x\rangle$ such that $Ax = b, x \geq 0$.
Given a starting point x' such that $x' > \delta$, the algorithm finds some $\hat{x}$ such that

$$c^T\hat{x} - c^T x^* \leq \varepsilon.$$

Moreover, the number of iterations is

$$\sqrt{m}\,\mathrm{poly}\left(\log m, \log\frac{\|c\|_2}{\varepsilon}, \log D8\right)$$

where D is the ℓ_2-diameter of the polytope.

8.4 Minimum Cost Flow

Suppose we are given a digraph $G = (V, E)$ and source-sink vertices $s \neq t \in V$.

Definition 8.4.1 ($s-t$ Flow)
A unit $s-t$ flow is some $x \in \mathbb{R}_+^m$ such that for all $u \in V$,

$$\sum_{vu \in E} x_{vu} - \sum_{uw \in E} x_{uw} = \begin{cases} 0, & u \neq s,t \\ 1, & u = s \\ -1, & u = t \end{cases}$$

In other words,

$$Bx = \chi_{x,t} = e_s - e_t$$

where B is the vertex-edge incidence matrix.

Problem 1 (Minimum Cost Flow)
Given

(a) a digraph $G = (V, E)$ and $s \neq t \in V$,

(b) capacities $\rho \in \mathbb{R}_+^m$,

(c) and costs $c \in \mathbb{R}^m$,

solve

$$\begin{aligned} &\min \langle c, x \rangle \\ Bx &= \chi_{s,t} \\ x_i &\leq \rho_i && \forall i \in E \\ x_i &\geq 0 && \forall i \in E \end{aligned}$$

We would like to apply the interior point algorithm we developed to solve the minimum cost flow problem. However, we need to efficiently determine a starting point and analyze the complexity per iteration.

8.4.1 Starting Point

We need $\delta \geq 1/\text{poly}(m)$ and $D \leq \text{poly}(m)$.

In our case, $0 \leq x \leq \rho$ and so

$$\begin{aligned} \|x\|_2 &\leq \|\rho\| \\ &\leq \sqrt{m}\|\rho\|_\infty \\ &=: \sqrt{m}U. \end{aligned}$$

For δ, there may be no feasible unit flow, hence add an extremely expensive edge $\hat{e} = s \to t$ with cost $2\sum_{i=1}^m |c_i|$ and capacity $\rho_{\hat{e}} = 2$. Consider some $f^{(i)}$ which is any unit flow using

edge i. If no such flow exists, we may as well throw out that edge. Then

$$x' := \frac{1}{m} \sum_i f^{(i)}$$

satisfies $x' \geq \frac{1}{m}$ for all $i \in E$.

We can also handle the capacity constraints using a similar trick. All in all, there is a

$$\tilde{O}(m^{1.5})$$

time algorithm that solves the minimum cost flow problem.

Chapter 9

Ellipsoid Method

9.1 Separation

Supppose we wish to optimize over 0-1 polytopes of the form

$$P_{\mathcal{F}} := \text{conv}\{\mathbb{1}_S : S \in \mathcal{F}\}$$

for some $\mathcal{F} \subseteq 2^m$. We can formulate matchings, flows, etc, all in this setting.

Unfortunately, there is not always a polynomial size description of $P_{\mathcal{F}}$ and we cannot hope to apply our existing techniques which work with explicit constraints.

Edmonds was the first to show that we separate over the matching polytope.

$$P_M = \left\{ x \in [0,1]^m : \forall S \subseteq [m], 2 \nmid |S|, \sum_{i \in S} x_i \leq \frac{|S|-1}{2} \right\}.$$

Specifically, he derived an algorithm which returns YES if $x \in P$ and a separating hyperplane if $x \notin P$. Moreover, for the case of matchings, the separating hyperplane is one of the (exponentially many) constraints.

9.2 Optimization & Feasibility

Problem 2 (Feasibility)
Given some polytope P, output NO if $P = \varnothing$, otherwise output YES as well as some $x \in P$.

We remark that having a feasibility oracle allows us to optimize by performing binary search on $g \in \mathbb{R}$. In each iteration, we ask if the following polytope is feasible.

$$P' := P \cap \{x : \langle c, x \rangle \geq g\}.$$

We will see how the ellipsoid method allows us to determine feasibility given

(i) a separation oracle,

(ii) a cost vector $c \in \mathbb{R}^m$,

(iii) lower and upper bounds $\ell_0 \leq \text{OPT} \leq u_0$.

Note that we can terminate the binary search when the current bounds $u - \ell \leq \varepsilon$. Thus this requires at most $\log\left({}^{(u_0-\ell_0)}/_{\varepsilon}\right)$ iterations.

9.3 Ellipsoid Method

Suppose there is a polytope P such that

(i) P is full-dimensional,

(ii) $P \subseteq B_R$, a ball of radius R that is given to us,

(iii) $B_r \subseteq P$ contains a ball of radius r that is not provided to us.

The idea is to produce a sequence of convex bodies $B_R = K_0 \supseteq K_1 \supseteq \dots$ where the volume satisfes

$$\text{Vol}(K_{t+1}) \leq \alpha \,\text{Vol}(K_t).$$

Then we need only repeat for

$$\log_{1/\alpha} \left(\frac{R}{r}\right)^m = m \log_{1/\alpha} \frac{R}{r}$$

iterations. For 0-1 polytopes, we know that $R \leq \sqrt{m}$ and in many cases we can lower bound $r \geq {}^1/_{\text{poly}(m)}$. Thus this would yield an efficient algorithm for separation.

We will employ ellipsoids as our convex body. Define

$$E_t(x_t, M_t) := \{x : (x - x_0)^T M^{-1} (x - x_0) \leq 1\}$$

to be the ellipsoid at time t for $x_t \in \mathbb{R}^m, M \succ 0$.

The algorithm proceeds as follows.

1) If $x_t \in P$, return x.

2) Else $x_t \notin P$ and there is a separating hyperplane h_t:

   a) Then construct E_{t+1} to be the minimum volume ellipsoid containing $E_t \cap \text{halfspace}(h)$.

9.3.1 Minimum Volume Ellipsoid

How do we construct E_{t+1} and how much does the volume shrink?

Consider the unit ball and the hyperplane $\{x : x_1 \geq 0\}$. We claim that it suffices to consider this special case since we can transform any ellipsoid with a hyperplane through its center to this case with an affine map.

Define

$$E' = \left\{ x \in \mathbb{R}^m : \left(\frac{m+1}{m}\right)^2 \left(x_1 - \frac{1}{m+1}\right)^2 + \left(\frac{m^2-1}{m^2}\right) \sum_{j=2}^{m} x_j^2 \leq 1 \right\}$$

$$x' = \frac{1}{m+1} e_1$$

$$M' = \text{Diag}\left(\left(\frac{m}{m+1}\right)^2, \frac{m^2}{m^2-1}, \frac{m^2}{m^2-1}, \dots \right).$$

Proposition 9.3.1
The following holds.

1) $E' \supseteq E \cap p\{x : x_1 \geq 0\}$
2) $1/\alpha \approx 1 + \frac{1}{m}$

Proof
1) Let x be such that $\|x\|_2^2 \leq 1$ and $x_1 \geq 0$. We show the stronger statement that

$$\left(\frac{m+1}{m}\right)^2 \left(x_1 - \frac{1}{m+1}\right)^2 + \left(\frac{m^2-1}{m^2}\right)(1 - x_1^2) \leq 1.$$

By taking the second derivative, we can check that the LHS is a convex function of x_1. Hence it suffices to check the inequality at $x_1 = 0, 1$, where is holds by computation.

2) We have

$$\begin{aligned}
\frac{\mathrm{Vol}(E')}{\mathrm{Vol}(E)} &= \sqrt{\det M} \\
&= \left(\frac{m}{m+1}\right) \cdot \left(\frac{m^2}{m^2-1}\right)^{\frac{m-1}{2}} \\
&= \left(1 - \frac{1}{m+1}\right) \cdot \left(1 + \frac{1}{m^2-1}\right)^{\frac{m-1}{2}} \\
&\leq \exp\left(-\frac{1}{m+1}\right) \exp\left(\frac{m-1}{2(m^2-1)}\right) \\
&= \exp\left(-\frac{1}{m+1}\right) \exp\left(\frac{1}{2(m+1)}\right) \\
&= \exp\left(-\frac{1}{2(m+1)}\right) \\
&\approx 1 - \frac{1}{2m}.
\end{aligned}$$

Hence $1/\alpha \approx 1 + 1/m$ and

$$t \approx m^2 \log \frac{R}{r}$$

iterations suffices.