IE 521 – Convex Optimization (Fall 2023)

Roy Dong

August 28, 2023

CONTENTS

PART I

COURSE INTRODUCTION

1
THE PURPOSE OF THIS DOCUMENT

This document was created during my first semester teaching a convex optimization course. Its goals were to do a highly verbose discussion for the interested student with some holes in their mathematical background. As such, it features extended discussions on the role of optimization, as well as more pedantic discussion of mathematical notation and ideas from real analysis and linear algebra. It also spends time drawing intuitions for basic concepts in convex analysis. This can cover anywhere from three lectures to half a semester, depending on the depth in which the instructor wishes to delve on the review. I found it was not necessarily worthwhile to write up notes for the rest of the material in the semester as there were several great resources online that, given a good understanding of the material I cover in detail here, should be quite readable and I found I could add little value in its transcription.

2
OVERVIEW

Hi. Welcome to 'Convex Optimization'.

Broadly speaking and oversimplifying, convex optimization refers to the minimization of convex functions, and convex functions are those that are roughly shaped like bowls. So, this class is the graduate-level study of 'getting to the bottom of bowls'. We'll define these things more formally very shortly, but first let's provide a bit of context for all of this.

Related reading: Boyd and Vandenberghe [2004] Ch. 1.

Why optimization?

Why do we study convex optimization? *Well, optimization shows up pretty much everywhere.* Pretty much in any setting where you want to do something as 'good' as possible subject to some constraints, and you can quantify 'good' with some cost or utility function, then you have an optimization problem. Different utilities or cost functions can model how quickly a task is completed, the cost of the raw materials to produce a good, how satisfied a person is with their decisions, or any balance between these factors. Once you start thinking like an optimization person, it becomes practically impossible to find any task that can't be framed as an optimization problem, and optimization begins to feel like *the* tool to do anything optimally.

Here's a few examples of optimization problems that show up in economics, control theory, and machine learning.

Example 1: Utility maximization subject to a budget

Let's say you have C dollars in your pocket, and the grocery store has n goods. The ith good costs p_i, and we'll call $p = (p_1, \ldots, p_n)$ the *price vector*. Let's considering buying x_i of good i, and let $x = (x_1, \ldots, x_n)$ be our *consumption vector*. In that case, the cost of that consumption vector will be $p^\intercal x = \sum_i p_i x_i$; you'll be able to afford it if $p^\intercal x \leq C$.

Additionally, we associate a *utility* with each consumption vector, denoted $u(x)$. Here, the utility is a real number, and the units for this are usually *utils*. Loosely, you can think of utils as a measure or preference: the more utils associated with x, the more you prefer x. Beyond the idea of 'utils', there's a few common assumptions in economics, which we'll go over here.

One is the principle of *non-satiation*: people always want more. That is to say: if $x \geq y$, then $u(x) \geq u(y)$. Here, $x \geq y$ for vectors in $\mathbb{R}^n$ means that eveny component is weakly greater, i.e. $x_i \geq y_i$ for all i. Non-satiation states that the utility from x is greater than the utility from y if $x \geq y$.

Another common assumption is *diminishing returns*. If you're currently consuming x, the instantaneous change in your utility from getting a little bit more of good i is $\frac{\partial}{\partial x_i} u(x)$. This is called the *marginal utility of i at x*. Diminishing returns says that the utility you get from goods is decreasing as you have more: the utility you get from your first slice of cake is higher than the utility you get from your 100th slice of cake. Formally, this means that $h \mapsto \frac{\partial}{\partial x_i} u(x + he_i)$ is a decreasing function for any x, where e_i is the vector which is 1 at the ith coordinate and zero everywhere else.

So, utility maximization subject to a budget constraint looks like:

$$\begin{aligned} \max_x \quad & u(x) \\ \text{s.t.} \quad & x \geq 0 \\ & p^\intercal x \leq C \end{aligned}$$

We'll talk about how to use these assumptions of non-satiation and diminishing returns down the road. You will be able to see what the consequences of them are, and build an intuition for what happens to common economic models if people *do* get satiated at some point, or if someone's just in love with good i and doesn't experience diminishing returns.

Example 2: Optimal control of a dynamical system

Suppose we have a dynamical system with **state** $x \in \mathbb{R}^n$, **inputs** $u \in \mathbb{R}^m$, and the following **dynamics**:

$$x_{k+1} = f(x_k, u_k)$$

If you haven't seen this before, you can think of the following.

The state captures everything needed to determine how a system will behave. For example, if we consider a single car on the road, the state may be the current position, velocity, heading, and wheel orientation.

The inputs capture all the decisions we can make to influence where the system will go. In a car, the inputs are how depressed the gas pedal and brake pedals are, as well as the steering wheel angle.

The dynamics specify how the state will update: given the current state and the inputs issued, what will the next state be? For a car, consider how the future position, velocity, heading, and wheel orientation are influenced by the current values as well as the steering wheel and pedals.

The dynamics tell us what the system will do *given* the inputs $u = (u_0, u_1, \dots)$. (Note that in this notation, u_k is the input at time k and u is all the inputs across time.) Control is about *finding* the inputs u that make the system do what we want it to do.

We can frame this as an optimization problem by introducing some cost $J(x, u)$. For example, we can set $J(x, u) = +\infty$ if the car crashes at some point. We can also try to get somewhere as quickly as possible, penalizing how long it takes the state x to be in a particular place. We can also penalize uncomfortable rides, making $J(x, u)$ high for trajectories that have a lot of extreme acceleration. Then, optimal control can be written as:

$$\begin{aligned} \min_{x,u} \quad & J(x, u) \\ \text{s.t.} \quad & x_{k+1} = f(x_k, u_k) \text{ for all } k \end{aligned}$$

Some versions of this are easier to solve than others; if f is linear and J is quadratic, then we have the **linear quadratic regulator (LQR)** problem, a fundamental concept in optimal control. LQR is a convex optimization problem! In practice, many roboticists will use LQR in some part of the overall system, even when the dynamics are non-linear and the optimization is non-convex, partially because LQR is so well understood and has very mature methods. It works quite well in practice.

Example 3: Loss minimization in machine learning

We've been tasked with identifying the cats, street signs, and bicycles in a sequence of photos. More generally, there's some randomized function that we don't know:

$$y = f(x, \omega)$$

Here, x is the input and y is the output, and ω is an element of our underlying probability space. (If you're not familiar with probability theory, you can just think of it as $f(x)$ but random, i.e. the value of y does not only depend on x but also some randomness.) For example, x are the JPEGs and y may be the labels.

So we're trying to figure out a good approximation of this mapping $x \mapsto y$. Broadly speaking, we have some family of candidate functions to approximate the true f. We'll call the collection of candidate functions $\mathcal{F}$, and we'll assume this collection is parameterized by some $\theta \in \mathbb{R}^n$, so $\mathcal{F} = \{g(\cdot; \theta) : \theta \in \mathbb{R}^n\}$.

A simple example is when we're doing linear regression, and in this case $g(x, \theta) = \theta^\intercal x$, and the family of candidate functions is all linear functions $x \mapsto \theta^\intercal x$.

If we have a bunch of independent data points $(x_i, y_i)_{i=1}^N$, then we may want to find the function in $\mathcal{F}$ that most closely matches these observations. Suppose we have some **loss** $\ell(y, \hat{y})$ which tells us how much penalty we suffer if the true value of the output is y but we guessed that it is $\hat{y}$.

This framework encapsulates a large amount of machine learning. We can write this as the following optimization:

$$\min_{\theta} \sum_{i=1}^{N} \ell\left(y_i, g\left(x_i; \theta\right)\right)$$

A common loss is $\ell(y, \hat{y}) = |y - \hat{y}|^2$, the squared error. When the loss is the squared error and the family of functions under consideration is linear functions, then this problem is **linear regression**.

If we instead consider deep learning, the candidate function set $\mathcal{F}$ is often the set of all neural networks with a particular structure and the parameter θ is the vector of all the weights connecting neurons in the network. The loss function varies but is very frequently still the mean squared error. Things like the backprop algorithm are methods to try and push θ towards 'better' values.

Yet another set of optimizations that we'll be able to understand and visualize a lot better at the end of this semester.

Why *convex* optimization?

Okay, so optimization itself is useful in a variety of applications. Pretty much in any setting where you want to do something as 'good' as possible subject to some constraints, and you can quantify 'good' with some cost or utility function, then you have an optimization problem. Different utilities or cost functions can refer to how quickly a task is completed, the cost of the raw materials to produce a good, how satisfied a person is with their decisions, or any balance between these factors. Once you start thinking like an optimization person, it becomes practically impossible to find any task that can't be framed as an optimization problem, and optimization begins to feel like **the** tool to do anything as best as possible.

Why *convex* optimization? Well, we'll get into that in more detail, but, in short, convex optimization problems have a lot of nice theoretical properties that also provide insight into non-convex optimization problems in general.

Okay, so hopefully you're sold that optimization is worth learning. These days, the main line drawn between optimization problems is between 'convex' and 'non-convex' problems. If a problem is 'convex', we generally expect to actually find the minimum of the function, which is nowadays often referred to as the **global minimum**. Formally, x^* is a global minimum of f if $f(x^*) \leq$

$f(x)$ for all x in the domain of f. (This coincides exactly with the definition of minimum we gave in the review of real analysis.) This contrasts with **local minima**, which are points that do not minimize the function across the entire domain, but rather in a neighborhood, i.e. x^* is a local minima if there exists some neighborhood U of x^* such that $f(x^*) \leq f(x)$ for all $x \in U$. Note that we're quantifying across the neighborhood U instead of the whole domain now.

Many convex optimization problems are effectively 'solved', in the sense that it is relatively easy to use an off-the-shelf solver to find the global minimum of a function. When convex problems are hard, it's often due to the scale of the problem being very, very large (or, in more rare cases, the curvature being too sharp). At no point do folks consider 'local minima' because, as we'll see, all local minima are global minima in convex optimization.

In contrast, 'non-convex' problems are generally considered difficult. In general, people usually have very little hope of finding global minima of these functions. In these areas, the term 'minima' simply refers to local minima and global minima are rarely discussed.

Even when an underlying problem is non-convex, there are good avenues of research using methods from convex optimization. For example, there is often fruitful research by finding ways to equivalently reformulate a non-convex problem as a convex one, and this success is often viewed as essentially 'solving the problem'. For example, Bose et al. [2015] shows that non-convex problems in optimal power flow are actually equivalent to certain convex relaxations. (This is done by using a larger convex set instead of the original feasible set, and showing that the optimizers of this relaxed problem are actually going to be in the original feasible set.) Alternatively, one can often use convex optimization methods to arrive at satisfactory solutions for non-convex problems. For example, Schulman et al. [2013] solves a sequence of convex optimization problems that approximate a non-convex problem to find feasible robot trajectories that avoid collision.

In summary, it's a very common viewpoint these days that the line between convex and non-convex optimization is one of the major delineations that separate (mostly) easy problems from (mostly) hard problems. However, this was not always the viewpoint. Prior to the 1980s, most people believed that the delineation between 'easy' and 'hard' is the line between 'linear programming' and 'non-linear programming'. The earliest formal argument that 'convexity' should be the fundamental property that determines the ease of an optimization problem was presented in Nemirovsky and Yudin [1983]. This idea of 'convexity' being the main delineation in optimization is explicitly pushed in Rockafellar [1993]. Further bibliographic details can be found in the notes at the end of every chapter in Boyd and Vandenberghe [2004].

3
WHAT TO EXPECT FROM THIS COURSE

This is my first time teaching the course so the contents themselves are still a little TBD.

Broadly speaking, however, this course will cover some basics of optimization (including non-convex optimization), such as optimality conditions and basic algorithms such as gradient descent. This course will also cover concepts from convex analysis such as the definition of a convex set and supporting hyperplane. Then, the course will delve into the notions of Lagrangian duality in convex optimization, from a theoretical perspective. Finally, all these concepts will be combined to understand several algorithms for convex optimization and why they work. Before all of this, though, we will recap some key concepts from real analysis and linear algebra which should be familiar from prerequisites.

The text that follows in this subsection are taken from the syllabus.

The expectations of the students in this course will be quite high. You will be expected to have an understanding of convex optimization from *both* a theoretical and applied perspective.

This course will be theoretical and rigorous, and you will be expected to give valid and detailed mathematical proofs. As this is a class, for pedagogical purposes we will expect more verbosity in proofs (e.g. explaining individual lines in derivations and what properties are invoked) than is typical of, say, a publication.

Additionally, some homework assignments will require programming. You are expected to set up your programming environment prior to the first programming homework.

If your mathematical foundations are a little rocky, it is expected that you will strengthen them through the course of this semester. If you haven't implemented algorithms before, you will be expected to learn enough about programming to be able to code up the algorithms covered in class.

You are **encouraged** to take this class even if your mathematical foundations are a bit rocky, or if you haven't had much experience implementing algorithms before. It's why I'm here as an instructor and you're supposed to learn from the course, not know everything prior to the course. However, if some of these more preliminary ideas outside the scope of the class are a bit foggy, please expect to put in a bit of extra time for 'catch up' and allocate accordingly. Some resources for self-learning will be provided, but if there's some aspects that are not familiar to you, feel free to reach out to me so I can try to find the appropriate resource to help fill in any gaps.

Overall, the goal of this course is to instill an intuition and understanding that will allow you to develop new algorithms, and not simply just implement existing ones. Furthermore, if you have a novel algorithm in hand, you are expected to have to tools to analyze this algorithm to see what properties it has.

How to approach this course

One of the terms I dislike but have failed to find an alternative to is 'mathematical maturity'. Loosely speaking, this is a term that doesn't have a formal definition, but it's more of a 'you know it when you see it' type thing. It essentially means that you have the ability to 'speak mathematics fluently', if you think of mathematics as a language. If you're not sure if you have mathematical maturity, I will provide some references on the course website which should hopefully help with this. You should think of one of this course's objectives as building mathematical maturity, if you are lacking it.

To this end: for every concept we cover, you should be able to **both** *formally* and *intuitively* understand the concept. By intuitively, I mean you should be able to represent the concept in your head and inspect it and just see what properties are true about it. This is often a geometric intuition for things taking place on real vector spaces (i.e. you can close your eyes and picture it in visual space), but people's intuitions may vary. By formally, I mean you should be able to precisely and pedantically write out mathematical symbols to describe your intuition, as well as use logically valid rules of inference to write proofs for true statements. Note that these are not the same thing, nor does the ability to do one imply the other.

You should form this expectation of yourself. When studying any material in the lecture notes, you should both be trying to build an intuition for what the concepts *mean*, but *also* learning how these intuitions show up formally in proofs. For example, you may visualize two disjoint convex sets in your mind, and see that you can always draw a straight line between them; formally, you should be able to verbalize this with a separating hyperplane argument. You should expect *both* of yourself, and study accordingly.

PART II

MATHEMATICAL BACKGROUND

4
QUICK REVIEW OF SET THEORY

In this section, we'll quickly discuss the foundational mathematical concepts in set theory that will be required for this class. In addition, we'll cover the set-theory notation that will be used throughout the entire semester.

You should be comfortable with set theory and its corresponding notation. A set is a mathematical object, which itself holds some collection of mathematical objects. Here is a quick recap of several things you should be comfortable with:

(*a*) For a set A and some mathematical object x, we write $x \in A$ to denote that x is a **member** of A, i.e. A **contains** x. We will write $x \notin A$ to denote that x is *not* a member of A.

(*b*) We use $\emptyset$ to denote the **empty set**, i.e. the set which has no members.

(*c*) Sets can often be defined according to some property its objects have. The notation for this will be as follows: the set of objects in A which satisfy a property $P(\cdot)$ is written $\{x \in A : P(x)\}$. For example, if we consider the set of non-negative real numbers, we would write it as follows: $\{x \in \mathbb{R} : x \geq 0\}$.

Sometimes the set A is omitted, either out of a desire for brevity or laziness. In this case, it'll look like: $\{x : x \geq 0\}$.[1]

(*d*) Given two sets A and B, we say A is a **subset** of B if every element of A is an element of B. This is denoted $A \subseteq B$.

[1]For those either philosophically interested or deeply pedantic, whenever we do mathematical reasoning we have something called the **domain of discourse**. Other terms are the **universe of discourse** or simply the **universe**. This is the set of all mathematical objects under consideration. When nothing is specified, it is assumed that the set we are drawing from is the domain of discourse, and then context will help restrict this further, e.g. $x \geq 0$ tells us that we're only considering mathematical objects where writing $x \geq 0$ makes sense.

To prove $A \subseteq B$, we must take any arbitrary element of A and then show that it is also an element of B. In other words, the proof will start with some sentence like "Let x be any element of A." Some reasoning will follow, and then it will end with "Thus, we have shown $x \in B$. Since x was an arbitrary element of A, we have shown that $A \subseteq B$."

To prove two sets are the same, i.e. $A = B$, we have to show two things: $A \subseteq B$ and $B \subseteq A$. The rules for showing either of these things is given in the previous paragraph.

If $B \subseteq A$, we may sometimes write $A \supseteq B$ to indicate that A is a **superset** of B, which simply means that B is a subset of A.

(e) For two sets A and B, we write $A \cup B$ to denote the **union** of A and B, with the rule that $x \in A \cup B$ if and only if $x \in A$ or $x \in B$. The union can be thought of as the 'OR' operation.

(f) Similarly, for two sets A and B, $A \cap B$ is the **intersection** and can be thought of as the 'AND' operation: it is the set of all mathematical objects in *both* A and B.

(g) For a set A and B such that $A \subset B$, we can define the **complement of A in B** as $B \setminus A = \{x \in B : x \notin A\}$. Sometimes, when the set B is obvious from context, we will simply refer to this as the **complement of** A and denote it A^c.

(h) Given two sets A and B, the **Cartesian product** of A and B is the set of all 2-tuples (a, b) whose first element is in A and whose second element is in B, i.e. $a \in A$ and $b \in B$. This is denoted $A \times B$, and will sometimes be written $\{(a, b) : a \in A, b \in B\}$.

This idea generalizes to $A_1 \times A_2 \times \cdots \times A_n$ as the set of all n-tuples whose value at the ith index comes from A_i.

When all the sets in the Cartesian product are the same, you may sometimes see this written with an exponent, e.g. $\mathbb{R} \times \mathbb{R} \times \mathbb{R} = \mathbb{R}^3$.

(i) A **function** is a mathematical object which assigns elements of its **co-domain** (output space) to elements of its **domain** (input space). For example, consider:

$$f(x) = \begin{bmatrix} x \\ x+1 \end{bmatrix}$$

f here is the function, the domain is $\mathbb{R}$, and the co-domain is $\mathbb{R}^2$. f eats in real numbers and spits out a 2-tuple of real numbers. We will also sometimes refer to x as the **argument** of f.

To specify the domain and co-domain of a function, we may often write $f : X \to Y$. Here, X is the domain and Y is the co-domain, and this notation indicates that f maps X to Y.

Another notation that looks very similar but is different is the $\mapsto$ symbol. It tells you what every element of the domain is assigned to. So, the typical notation you've seen may be something like $f(x) = x^2$. I will often write $f : x \mapsto x^2$ instead. *Note the difference between $\to$ and $\mapsto$.* Here, $f : \mathbb{R} \to \mathbb{R}$ while $f : x \mapsto x^2$. The former tells you the domain and co-domain whereas the latter tells you exactly what assignment is being made. *I will use this notation all the time in this class.*[1]

For two sets X and Y, we will write Y^X as the set of all functions with domain X and co-domain Y, i.e. all the functions that take elements of X and assign elements of Y to them.

We can also define the **image** of f as the elements in the co-domain that are assigned to some element of the domain, i.e. the image of f is $\{y \in Y : y = f(x) \text{ for some } x \in X\}$. We may also write this as $\{f(x) : x \in X\}$, which is just shorthand for the same thing. You will sometimes see this written as $f(X)$, i.e. the function with the domain itself written as the argument.

If we equate 2 with the set $\{0, 1\}$ which has two elements, we can write 2^A to denote the **powerset** of A. Note that every function from A to $\{0, 1\}$ assigns some subset of A to 1. Thus, every subset of A can also equivalently be thought of as a function $A \to \{0, 1\}$, and vice versa.[2]

[1] For those of you who code, you can think of this as analogous to lambda functions in Python, for example.

[2] This is a special case of the **von Neumann** construction of the natural numbers. In this system, the number 0 is represented by the empty set $\emptyset$, and the number 1 is represented by the set containing the empty set $\{\emptyset\}$, and the number 2 is represented by $\{\emptyset, \{\emptyset\}\}$, and so on. The further constructions are recursively defined as $n = \{0, 1, \ldots, n-1\}$: each number is the set of all numbers that came before. Note that each set n contains exactly n elements.

This also gives another motivation for the notation $\mathbb{R}^n$: using the von Neumann construction of the natural numbers, the n-tuples of real numbers can also be thought of as the set of functions which map a domain with n elements into the real numbers.

5
QUICK REVIEW OF REAL ANALYSIS

In this section, we'll quickly discuss the concepts from real analysis you should already be comfortable with from your prerequisites.

As a quick recap of concepts you should be familiar with:

(*a*) You should be comfortable with the ruleset that governs the natural numbers, which we will denote $\mathbb{N}$, and the real numbers, which we will denote $\mathbb{R}$. In this class, we'll consider 0 a natural number, as I am person who often prefers 0-indexing.

(*b*) Two key concepts we'll use a lot throughout this class is the notion of limits and extremizing. First, limits.

For a sequence of real numbers $(a_0, a_1, \dots)$, we will write $\lim_{n\to\infty} a_n$ to denote the **limit of a_n as n goes to infinity**. Mathematically, we say $a^* = \lim_{n\to\infty} a_n$ if the following holds: for any $\epsilon > 0$, there exists an N such that $|a_n - a^*| < \epsilon$ for all $n \geq N$. In words, this states that for any tolerance ϵ, there is some point in time N after which we are within ϵ distance of a^* forever. This has to hold for any $\epsilon > 0$, no matter how small. For short, we may also just write $\lim_n a_n$, or even $\lim a_n$.

Now, let's look at the definition above. The only real structure of the real numbers we used was the norm, and, even then, we only used this to define a distance between a_n and a^*. Thus, putting our mathematician hats on, we can find a natural generalization for this. Framing it more abstractly, we can say the following. Consider any sequence of elements in a metric space $(a_0, a_1, \dots)$. We say $a^* = \lim_{n\to\infty} a_n$ if the following holds: for any $\epsilon > 0$, there exists an N such that $d(a_n, a^*) < \epsilon$ for all $n \geq N$. (Here, $d(\cdot, \cdot)$ denotes the metric of the space.)

We'll try and emphasize this point a bit throughout the semester about how to do more mathematical/theoretical thinking. First, we find an intuitive definition of a concept or proof of a result in a very concrete given space, e.g. the real numbers. Then, we look at this definition or derivation and ask: "What were the key properties we actually used?" Then, we can immediately generalize to the abstract objects that only have the properties used, e.g. metric spaces.

Sometimes, like this case, this generalization will be easy and immediate. Other times, the generalization may not be as obvious. One thing that often happens is that there are multiple definitions that are equivalent in our concrete space but may become different in a more abstract space.

One example is the ϵ-δ definition of continuity. In metric spaces, this is equivalent to the topological definition of continuity. However, the latter generalizes to more abstract spaces. In particular, in this class, we will often task ourselves with generalizing intuitions in one-dimensional spaces to finite-dimensional spaces.

(c) We've discussed the limit of a sequence. A related notion is the limit as an argument approaches a point. We write $\lim_{x \to a} f(x)$ as the **limit of** $f(x)$ **as** x **approaches** a. This is a similar intuition as above but has a slightly different formal definition. We say $y^* = \lim_{x \to a} f(x)$ if the following holds: for any $\epsilon > 0$, there exists a $\delta > 0$ such that $|f(x) - y^*| < \epsilon$ for any x such that $|x - a| < \delta$. Again, in words, this states that for any tolerance ϵ, if the argument is δ close to a, then $f(x)$ is ϵ close to y^*, i.e. every argument in this δ ball around a evaluates to something that is ϵ close to y^*.

This may be defined in settings where f is not defined at a, and you may also see things such as $\lim_{x \to a^+} f(x)$ or $\lim_{x \downarrow a} f(x)$, which means the limit as x approaches a from the right/above.

(d) Next, let's discuss extremizing. Consider any subset of the real numbers $A \subseteq \mathbb{R}$.

An **upper bound** of A is any number $c \in A$ such that $c \geq a$ for every $a \in A$. The **supremum** of A is the **least upper bound** of A, denoted $\sup A$. By convention, if the set A has no upper bound, then we say $\sup A = +\infty$, and the supremum of the empty set is $\sup \emptyset = -\infty$. The existence of the supremum is a real analysis fact that we will not cover here.

How do we use this definition? Well, if $c^* = \sup A$, we know that $c^* \geq x$ for all $x \in A$, since it's an upper bound. Additionally, since it is the *least* upper bound, we know that for any $\epsilon > 0$, there exists an $x \in A$ such that $x > c^* - \epsilon$, i.e. for any $\epsilon > 0$, there's a point x in the set A that is less than ϵ shy of c^*. If such a point didn't exist, then we would have a lesser upper bound $c^* - \epsilon$. As such, we typically assume we have a **supremizing sequence** $(x_0, x_1, x_2, \dots)$ of points in A such that $x_n > (\sup\ A) - 1/n$ for each n.

It's here I want to emphasize that, throughout this course and in most settings, *whenever* we take the supremum, we are just trying to find the least upper bound of a subset of the real numbers. What happens in optimization to make this more and more complicated is the way we describe this subset of real numbers.

Related, we can define the **maximum** of a set A as follows. A point x is the maximum of A if x is an upper bound and $x \in A$. The latter point, that x itself is in A, is *crucial*. If no point in A satisfies this condition, then the maximum does not exist. Note that when the maximum exists, it is equal to the supremum. (The maximum is an upper bound. The least upper bound has to bound all elements of A, including the maximum.)

For example, consider the closed interval $A = [0, 1]$. The supremum is 1, and the maximum is 1. Next, consider the open interval $A = (0, 1)$. The supremum is 1, and the maximum does not exist.

Most frequently, we'll talk about finding the supremum of a function $f : X \to \mathbb{R}$, and, colloquially, we may sometimes refer to this as *maximizing a function*, even when we haven't done the full rigorous proof that a maximum exists rather than just a supremum. This is sometimes denoted $\sup f$, or $\sup_x f(x)$. This is just a shorthand for referring to $\sup \{f(x) : x \in X\}$. That is, we're taking the supremum of the image of f, which is a subset of the real numbers.

Again, it's pretty common convention to refer to this as 'maximizing a function' even when we haven't introduced all the technical assumptions to ensure a maximum exists, so I'll be doing this as well throughout the semester.

Okay, so what does the distinction between supremum and maximum mean for us when we are reasoning about things? Well, things are a lot more straightforward when the maximum exists. Suppose y^* is the maximum of f. Then we can simply say: let x^* be such that $f(x^*) = y^*$. A point in the domain *exists* that *attains* y^*. We call this point x^* an **argmax** of f, or **maximizing argument**. There can be multiple argmaxes. We can reason about x directly.

If we do not know a maximum exists, we have to be more indirect. If y^* is the supremum of f, we can consider a sequence of points in the domain $(x_0, x_1, \dots)$ such that $f(x_n) > y^* - 1/n$. The reasoning will often involve limits or other things, since we can't directly refer to a point x where $f(x) = y^*$. Without other conditions, we don't know much else about the sequence $(x_0, x_1, \dots)$; for example, it may not necessarily converge to anything. (See Figure 1 for an example.)

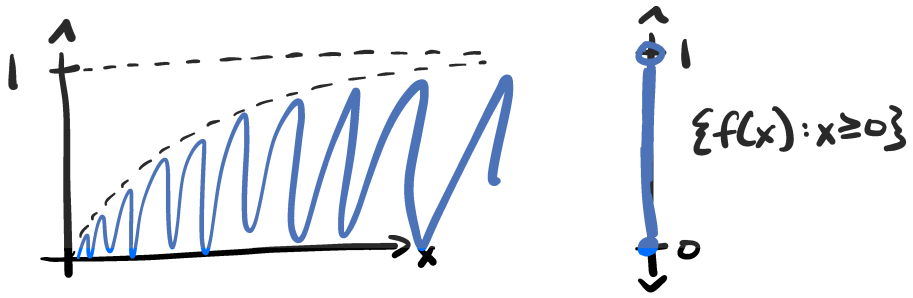

Figure 1: Consider the function $f(x) = (1 - \exp(-x))\,|\sin(x)\,|$, defined for $x \geq 0$. On the right, we see the image of f, the set $\{f(x) : x \geq 0\}$. The infimum and minimum are 0, with argmins $\{0, \pi, 2\pi, \dots\}$. The supremum is $\sup_{x \geq 0} f(x) = 1$ and the maximum does not exist. An example supremizing sequence is $(\pi/2, 3\pi/2, 5\pi/2, \dots)$. Note that this sequence does not converge to any particular point.

Sometimes, you'll see quite a bit of work trying to prove that the maximum exists. Another way to often say this is that the 'supremum is attained', i.e. there is a point which attains the supremum. This point is often the maximum. One of the nice things about this is described in the previous paragraph: we can work with a point rather than a sequence of points.

The same idea carries over when defining **lower bounds**. The **infimum** of a set is the **greatest lower bound**, and denoted $\inf A$. By convention, the infimum is $-\infty$ if there is no lower bound and the infimum of the empty set is $+\infty$. The **minimum** is a lower bound that is in the set, and we will refer to *minimizing a function* even when we may not have proven a minimum exists. If x^* is an argument that attains the infimum of the function, we say x^* is an **argmin** of f, or **minimizing argument**.

One thing we can directly note from these definitions is that $\inf f = -\sup(-f)$: minimizing a function is essentially the same as maximizing its negation.[1] Without loss of generality, we'll typically take infimums when extremizing functions; this is the convention that leads us to refer

[1]The equation here simply states that the values $\inf f$ and $\sup(-f)$ are negations of each other, but there's actually a stronger relationship. If $(x_0, x_1, \dots)$ is an infimizing sequence for f, then it is also a supremizing sequence for $(-f)$. Similarly, the argmins of f and the argmaxes of $(-f)$ are the same points.

to this class as 'convex optimization' rather than 'concave optimization'. (Convex functions have a particularly good structure for minimization, whereas concave functions have a particularly good structure for maximization.)

(*e*) You should be familiar with the topological notions of **open** and **closed** sets. In this class, we'll mostly only consider $\mathbb{R}^n$ and the topology induced by the norm, so we can use the following definitions.

A set $A \subseteq \mathbb{R}^n$ is **open** if the following holds: for every point $x \in A$, there exists an $\epsilon > 0$ such that $y \in A$ for all $|y-x| < \epsilon$. Essentially, for any point in A, you can wiggle around a bit without leaving A. As an example, the interval $(0, 1)$ is open, for any number you choose in this interval, there will be some wiggle room, albeit it could be very, very small. If you take $x = 0.99999999999999$, you can still wiggle by $\epsilon = 0.000000000000005$ without leaving the set.

We'll also define $B_r(x) = \{y \in \mathbb{R}^n : |y - x| < r\}$ as the **open ball of radius** r **centered at** x. So, the definition of open-ness can be written as follows: for every $x \in A$, there exists an $\epsilon > 0$ such that $B_\epsilon(x) \subseteq A$.

One example of where this shows up is when we consider derivatives. Something you may not have realized before is this technical detail. Say $f : [0, 1] \to \mathbb{R}$ is a function we are differentiating. The derivative is only defined on $(0, 1)$. Note that the derivative requires one to look a little bit in both directions to define the instantaneous rate of change. As such, we can't define the derivative at 0 or 1 because we can't look before 0 or after 1 to determine the rate of change. Here, open-ness makes sense: to get an *instantaneous* rate of change, we just have to be able to wiggle a little bit in every direction, no matter how small the wiggle is, we can still get an instantaneous rate of change.

Another term you will see is a **neighborhood** of a point x. A neighborhood of x is any set A such that there exists an $\epsilon > 0$ such that $B_\epsilon(x) \subseteq A$. In words: a neighborhood of x is any set that contains some open ball centered at x.

A set $A \subseteq \mathbb{R}^n$ is **closed** if its complement $A^c = \mathbb{R}^n \setminus A$ is open. This definition hints at the duality between open-ness and closed-ness. A more directly useful definition is that a set A is closed if it contains all its limit points. Formally, for any sequence $(a_0, a_1, \dots)$ of elements in A, if $\lim_n a_n$ exists, then it is also in A.

The basic idea is that you can't take elements of a set to wind up somewhere outside the set. If you consider $(0, 1)$, you could take the sequence $(0.9, 0.99, 0.999, \dots)$ and have every element of the sequence live in the set, whilst the end result leaves the set. Closed-ness assures us that this won't happen: you can safely take limits and not leave the set.

Note that the empty set $\emptyset$ is both open and closed by the definitions. Open-ness states a property that holds for every x in the set, which is trivially satisfied when there are no elements in the set.

(f) A **neighborhood** of x is any open set U that contains x. Basically, U doesn't have to be particularly 'big', but you have to be able to go a little bit in every direction.[1] You can think of a neighborhood as just any set which contains some open ball around x.[2]

(g) You should also be familiar with the definitions of a **convergent sequence**, a **Cauchy sequence**, and a **complete space**.

[1]There's some caveats on the loose informality of this sentence: this intuition only really should be used on metrizable topologies. However, we're not really going to do stuff with non-metrizable topologies in this course so for now we're going to just stick to this intuition. Also, the reason the word 'big' is in scare-quotes is because we don't have a formal definition of 'bigness', which is also a bit outside the scope of this class.

[2]Another technicality for the very pedantic students in the crowd: some textbooks will distinguish between an 'open neighborhood' and a 'neighborhood'. In these books, a 'neighborhood' is any set which contains an open ball around x, and an 'open neighborhood' is a neighborhood which itself is open. Often, the distinction does not matter and many textbooks will just use the term neighborhood for both. The reason the distinction often does not matter is because the definition is often used because of the existence of the open ball around x, not the open-ness of the set itself.

6
SLIGHTLY LONGER BUT STILL QUICK REVIEW OF LINEAR ALGEBRA

Now, we'll quickly recap some linear algebra background which should be familiar to you.

So, what's **matrix multiplication**? Well, let's consider the matrix $A \in \mathbb{R}^{m \times n}$. We can think of A as n columns in $\mathbb{R}^m$:

$$A = \begin{bmatrix} | & | & & | \\ a_1 & a_2 & \dots & a_n \\ | & | & & | \end{bmatrix}$$

If we have a vector $x \in \mathbb{R}^n$, we can define matrix multiplication as x telling us how much of each column to mix in:[1]

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

$$Ax = x_1 \begin{bmatrix} | \\ a_1 \\ | \end{bmatrix} + x_2 \begin{bmatrix} | \\ a_2 \\ | \end{bmatrix} + \cdots + x_n \begin{bmatrix} | \\ a_n \\ | \end{bmatrix}$$

Now, what does it mean to solve a matrix equation? Suppose we have some vector $b \in \mathbb{R}^m$, and we're trying to figure out how to write b as a combination of the columns of A. Basically, given A and b, we're trying to find x such that $Ax = b$:

$$\begin{bmatrix} | \\ b \\ | \end{bmatrix} = x_1 \begin{bmatrix} | \\ a_1 \\ | \end{bmatrix} + x_2 \begin{bmatrix} | \\ a_2 \\ | \end{bmatrix} + \cdots + x_n \begin{bmatrix} | \\ a_n \\ | \end{bmatrix}$$

Put another way, how can we take linear combinations of columns of A to get b? How much of each a_i should we mix into the linear combination?

- **When does such an x exist?** A set of weights $x = (x_1, \dots, x_n)$ for the columns of A exists when b is in the span of the columns of A, or, equivalently, b is in the **range** of A, often denoted $R(A)$.

[1] We're going to treat vectors in $\mathbb{R}^n$ and $\mathbb{R}^{n \times 1}$ interchangably. Most math textbooks do this without any comment, but those who have had to program in NumPy are used to making this distinction between a 1-dimensional vector $\mathbb{R}^n$ (i.e. `x[0]` refers to the first element) and a 2-dimensional vector $\mathbb{R}^{n \times 1}$ (i.e. `x[0][0]` refers to the first element).

- **When does such an x exist for all $b \in \mathbb{R}^m$?** When can we find such weights for any b? Well, when $R(A) = \mathbb{R}^m$.

- **When is x unique?** Okay, so suppose we're given a b and there exists at least one x such that $Ax = b$, i.e. one way to linearly combine the columns of A to get b. When is there only one way to do so? This happens when the columns of A are **linearly independent**, or, equivalently, when the **nullspace** of A is *trivial* (i.e. contains only the zero vector). The nullspace is often denoted $N(A)$ and is sometimes referred to as the **kernel** of A.

  By linearity, if $Ax = b$ has a unique solution for some b, then $Ax = \tilde{b}$ has at most one solution for all $\tilde{b}$.

- **When does a unique x exist for all $b \in \mathbb{R}^m$?** Well, we can combine the last two conditions, so it's when $R(A) = \mathbb{R}^m$ and $N(A) = \{0\}$. Put another way, we need the n columns to be linearly independent *and* span all of $\mathbb{R}^m$; the only way this can happen is when $m = n$ and the columns of A form a **basis** for $\mathbb{R}^n$. When this happens, we say $A \in \mathbb{R}^{n \times n}$ is **invertible**.

This is all from the perspective of looking at Ax as taking a linear combination of columns. Dually, we can look at matrix multiplication as follows. A row vector $\alpha^\intercal \in \mathbb{R}^{1 \times n}$ can be thought of as a **linear functional**: it eats in vectors in $\mathbb{R}^n$ and spits out scalars: $x \mapsto \alpha^\intercal x$.

We can think of a matrix $A \in \mathbb{R}^{m \times n}$ as m row vectors that eat in elements of $\mathbb{R}^n$:

$$A = \begin{bmatrix} - & \alpha_1^\intercal & - \\ - & \alpha_2^\intercal & - \\ & \vdots & \\ - & \alpha_m^\intercal & - \end{bmatrix}$$

$$Ax = \begin{bmatrix} \alpha_1^\intercal x \\ \alpha_2^\intercal x \\ \vdots \\ \alpha_m^\intercal x \end{bmatrix}$$

Each entry of Ax is the result of a linear functional $\alpha_i^\intercal$ acting on x, which we will talk about in the next section on linear forms.

So, we can put these two together to better understand the **inverse matrix**. When A is invertible, $x = AA^{-1}x = A^{-1}Ax$ for all x. What does this mean? Well, let $z = A^{-1}x$ for a second and let $(a_i)_i$ be the columns of A. Then, we have $x = Az$:

$$x = z_1 \begin{bmatrix} | \\ a_1 \\ | \end{bmatrix} + z_2 \begin{bmatrix} | \\ a_2 \\ | \end{bmatrix} + \cdots + z_n \begin{bmatrix} | \\ a_n \\ | \end{bmatrix}$$

Each z_i tells us how much of a_i to put in to get x. Well, how much of a_i should I put in? Let $\tilde{\alpha}_i^\intercal$ denote the ith row of A^{-1}. Then:

$$z_i = \tilde{\alpha}_i^\intercal x$$

So, the ith row of A^{-1} is the linear functional that tells us how much the ith column of A is in x:

$$x = \left(\tilde{\alpha}_1^\intercal x\right) \begin{bmatrix} | \\ a_1 \\ | \end{bmatrix} + \left(\tilde{\alpha}_2^\intercal x\right) \begin{bmatrix} | \\ a_2 \\ | \end{bmatrix} + \cdots + \left(\tilde{\alpha}_n^\intercal x\right) \begin{bmatrix} | \\ a_n \\ | \end{bmatrix}$$

You can think of this as saying: with respect to the basis $(a_i)_i$, there is $\tilde{\alpha}_i^\intercal x$ of the basis vector a_i in x. That's all we mean when we write $x = AA^{-1}x$. Another set of terminology for the same idea is as follows: if A is an invertible matrix, its columns are a **basis** and the rows of A^{-1} are the **dual basis** of the columns of A.

One important thing to note is that we're answering the question: 'How much of a_i is in x with respect to the basis $\left(a_j\right)_j$?' The answer to this question *depends* on the other vectors as well, i.e. *for the same a_i and x, we may have to mix in different amounts based on the directions of the other $\left(a_j\right)_j$ vectors.*

If we try to represent any basis vector a_i with respect to the basis $(a_i)_i$, we can see that a_i is the result of mixing one unit of a_i and zero units of a_j for $j \neq i$. In other words:

$$\tilde{\alpha}_j^\intercal a_i = \begin{cases} 1 & \text{if } j = i \\ 0 & \text{if } j \neq i \end{cases}$$

In matrix form, this is simply $A^{-1}A = I$.

The same relationship holds between the rows of A and the columns of A^{-1} in the equality $x = A^{-1}Ax$.

Eigenvalues and eigenvectors

For a square matrix $A \in \mathbb{R}^{n \times n}$, we say $\lambda \in \mathbb{C}$ is an **eigenvalue** and a non-zero $x \in \mathbb{C}^n$ is an **eigenvector** if $Ax = \lambda x$. Note that the eigenvalues and eigenvectors may be complex, even when A is real. This is because the reals are not *algebraically closed*.[1]

The **determinant** of a square matrix A is the product of its eigenvalues (with repeated multiplication for repeated eigenvalues). A matrix is invertible if and only if zero is not an eigenvalue, i.e. the determinant is non-zero. If 0 is an eigenvalue of A, then the corresponding eigenvector x is in the nullspace of A by definition.

Note that if λ is an eigenvalue of A, then 0 is an eigenvalue of $\lambda I - A$, so $\det(\lambda I - A) = 0$. (Thus, if λ is an eigenvalue of A, then $\det(\lambda I - A) = 0$.) Conversely, if $\det(\lambda I - A) = 0$, then 0 is an eigenvalue of $\lambda I - A$. If x is its corresponding eigenvector, we can see $Ax = \lambda x$, so λ is an eigenvalue of A.

Linear forms

Throughout this course, we're going to do a lot with linear and quadratic forms. We'll start by covering linear forms.

Linear forms are often useful as approximations to a function:

$$f(x) - f(x_0) \approx \nabla f(x_0)^\intercal (x - x_0)$$

Loosely speaking, this approximation is good when $|x - x_0|$ is small. For example, this is the idea behind gradient descent: to try and find a minimum of a function, we move in a direction where the linearization decreases.

Additionally, we will see that linearizations lower-bound convex functions, as visualized in Figure 2. We'll define all of this more formally in the sequel.

[1] A set is algebraically closed if you don't wind up leaving the set due to algebraic operations. Consider $x^2 + 1 = 0$. This is a polynomial with all real coefficients, but when we do the algebraic operation of root finding, we find that there are imaginary roots. We have left the set of real numbers by an algebraic operation, so this set is not closed under algebraic operations. This is literally the same reason real matrices are not algebraically closed, as eigenvalues of a real matrix are simply roots of the polynomial with real coefficients $s \mapsto \det(sI - A)$.

Generally speaking, closure is very important to mathematicians. It is the mathematics equivalent of type-mismatch errors: when programming, you want to make sure your function returns an object of the correct type. In mathematics, without closure, you gotta introduce a lot more stuff to analyze things properly. (In this case, when you're taught the quadratic formula, you also have to spend a lot of time covering imaginary numbers because of this.) As the fable goes, Pythagoreans were so upset to find out the rationals were not closed under geometric operations that $\sqrt{2}$ was discovered, they threw Hippasus off a boat. I've named my Wi-Fi network after Hippasus.

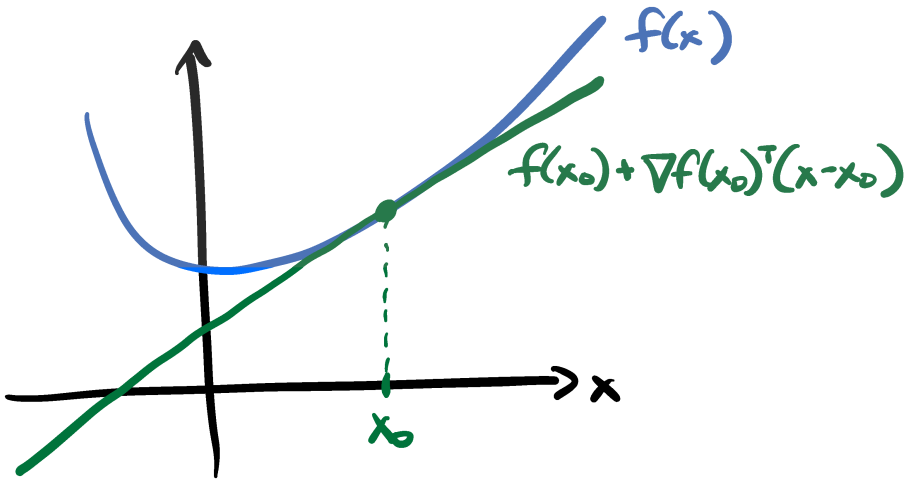

Figure 2: Convex functions are lower bounded by their linearizations at any point (the tangent lines to the graph). Here, we see a convex function f, and its linearization at a point x_0, given by $f(x_0) + \nabla f(x_0)^\intercal (x - x_0)$. This visualization is an example for a function $f : \mathbb{R} \to \mathbb{R}$, but this generalizes to functions $f : \mathbb{R}^n \to \mathbb{R}$. In this case, the graph is in $\mathbb{R}^{n+1}$ and the linearization is a hyperplane that sits below the graph of f. We will cover this in detail once we start formalizing convexity.

There are many other uses for linear forms, but these two will be the most common throughout the semester.

Fix $a \in \mathbb{R}^n$ and let's consider the associated linear form:

$$x \mapsto a^\intercal x$$

This is a pretty important concept so it's definitely worth it to spend some time mulling it over. The vector a points in some direction in $\mathbb{R}^n$. For any other vector x, we can break it down into the component that is in line with a, and a component that is orthogonal to a, as depicted in Figure 3. The inner product between a and x is determined entirely by the component that is in line with a.

Another very important concept we'll use throughout this semester is the notion of hyperplanes and half-spaces.

A **hyperplane** in $\mathbb{R}^n$ is a set of the form:

$$\{x \in \mathbb{R}^n : a^\intercal x = b\}$$

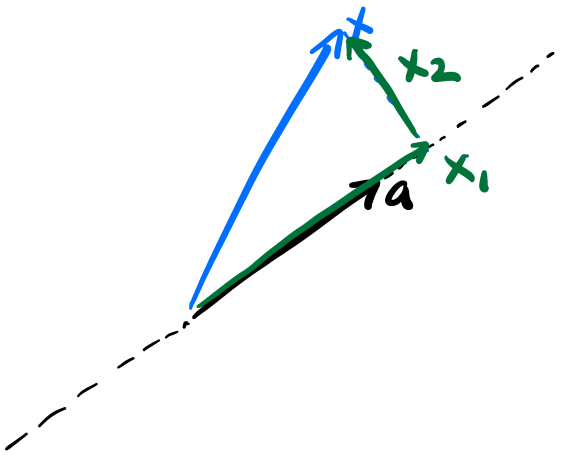

Figure 3: The vector a gives a direction, and x is broken down into the component that is in line with a (denoted x_1) and orthogonal to a (denoted x_2). The inner product between a and x is the same as the inner product between a and x_1: $a^\intercal x = a^\intercal x_1$. Pictures will typically be in ambient dimension $n = 2$ so hyperplanes are often drawn in dimension $n - 1 = 1$, i.e. lines. In $n = 3$ dimensions, hyperplanes will be $n - 1 = 2$ dimensional.

Here, $a \in \mathbb{R}^n$ and $b \in \mathbb{R}$. This is a single constraint, so it is an affine space of dimension $n - 1$. Put another way, it is of *co-dimension* 1: it is defined by 1 linear constraint, so the hyperplane has the dimension of the ambient space minus 1.

If we have some vector x_0 such that $a_\intercal x_0 = b$, we can re-write the hyperplane as:

$$\{x : a^\intercal (x - x_0) = 0\} = x_0 + \mathcal{N}(a^\intercal)$$

The hyperplane can thus be thought as follows: if we start at x_0 and move to x, we should be moving orthogonal to a. We can think of a hyperplane as a single point x_0 plus the nullspace of $a^\intercal$, as in Figure 4.

Each hyperplane splits $\mathbb{R}^n$ into two **halfspaces**, which are sets of the form:

$$\{x \in \mathbb{R}^n : a^\intercal x \leq b\}$$

The **gradient** of the linear form $f(x) = a^\intercal x$ is given by $\nabla f(x) = a$. Intuitively: the direction that most increases $f(x)$ is moving x in the direction a.

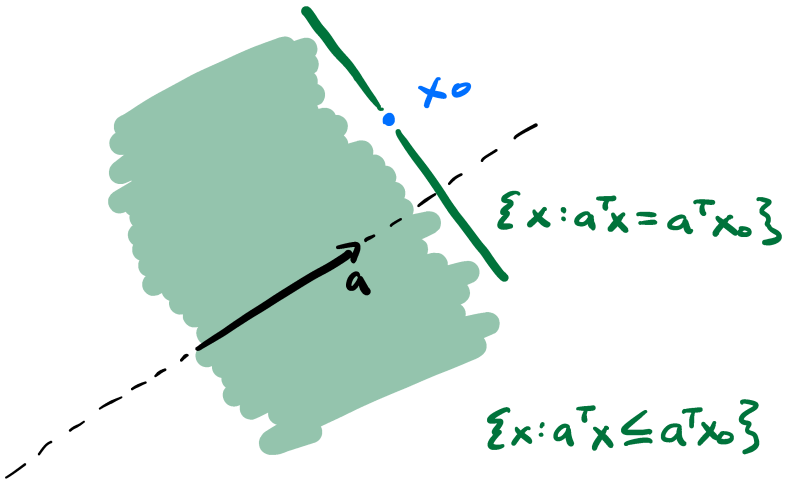

Figure 4: A hyperplane can be thought of as a point x_0 plus any point orthogonal to a. Loosely speaking, a hyperplane cuts the space in 'half', and a halfspace is the set of all points on one 'side' of a hyperplane.

Quadratic forms

Next, let's consider quadratic forms:

$$x \mapsto x^\intercal Q x$$

Sometimes, we'll want to take the second-order approximation of a function:

$$f(x) - f(x_0) \approx \nabla f(x_0)^\intercal (x - x_0) + \frac{1}{2} (x - x_0)^\intercal D^2 f(x_0) (x - x_0)$$

(If you haven't seen this before, $D^2 f(x_0)$ is the **Hessian** of f evaluated at x_0, i.e. the gradient of the gradient; I try to avoid the notation ∇^2 because it has different meanings, such as the Laplacian, and can be ambiguous at times.)

This is more accurate than the linear approximation near x_0, with the error being on the order of $o\left(|x - x_0|^2\right)$ rather than $o(|x - x_0|)$.

Also, in general optimization settings, we'll often find reasons to use upper and lower quadratic bounds: a lower quadratic bound is often provided by an assumption of *strong convexity* and an upper quadratic bound is often provided by an assumption of *smoothness*.

Before we get there though, we should cover some other linear algebra concepts.

First, the **transpose** of a matrix $A \in \mathbb{R}^{m \times n}$, denoted $A^\intercal \in \mathbb{R}^{n \times m}$, is the result of swapping the rows and columns of A, i.e. if a_i is the ith column in A, then $a_i^\intercal$ is the ith row of $A^\intercal$. We can think of the transpose as taking each column a_i of A and turning it into the linear operator $a_i^\intercal$.

A matrix is **symmetric** if $A = A^\intercal$ and it is **skew-symmetric** if $A = -A^\intercal$.

A symmetric matrix has all purely real eigenvalues, and a skew-symmetric matrix has all purely imaginary eigenvalues. Every matrix can be broken down into its symmetric and skew-symmetric part, as follows:

$$A = \left(\frac{A + A^\intercal}{2}\right) + \left(\frac{A - A^\intercal}{2}\right)$$

Loosely speaking, this is the n-dimensional generalization of the idea that any complex number can be broken up into a real part and an imaginary part, i.e. $x = a + ib$.[1]

Now, note with quadratic forms, for any skew-symmetric matrix A, the quadratic form is identically zero: $x^\intercal Ax = 0$ for all x. (I suggest you prove this as an exercise!) So, **without loss of generality**, we only consider quadratic forms where Q is symmetric in $x \mapsto x^\intercal Qx$.

Now, let's recall the intuition we were building before: we noted that $A \in \mathbb{R}^{n\times n}$ being invertible is the same thing as the columns of A forming a basis for $\mathbb{R}^n$. Additionally, we noted that the rows of A^{-1} were its corresponding dual basis. With the notation a_i for the columns of A and $\tilde{\alpha}_i^\intercal$ for the rows of A^{-1}, we discussed the interpretation that $\tilde{\alpha}_i^\intercal$ was a linear operator that stated 'how much' of a_i was in x, i.e.

$$x = \left(\tilde{\alpha}_1^\intercal x\right)\begin{bmatrix}|\\ a_1\\ |\end{bmatrix} + \left(\tilde{\alpha}_2^\intercal x\right)\begin{bmatrix}|\\ a_2\\ |\end{bmatrix} + \cdots + \left(\tilde{\alpha}_n^\intercal x\right)\begin{bmatrix}|\\ a_n\\ |\end{bmatrix}$$

Now, we say a basis $(a_i)_{i=1}^n$ is **orthonormal** if it is both orthogonal and normal. A basis is **orthogonal** if $a_i^\intercal a_j = 0$ for all $i \neq j$. A basis is **normal** if $a_i^\intercal a_i = 1$. So, they are unit norm and all point in orthogonal directions.

A nice thing to note about an orthonormal basis is that it is **self-dual**, i.e. the dual basis of $(a_i)_i$ is $\left(a_i^\intercal\right)_i$ when $(a_i)_i$ is orthonormal. Why is this? Well, let's apply the above intuition for inverse matrices and dual bases from above.

We can visualize the setting where the a_i are orthogonal and unit norm in Figure 5. Given these are the basis vectors, when we ask 'how much of a_i is in x?', we can get the answer by taking $a_i^\intercal x$: we project x onto a_i and this is our answer.

[1]The reason why this is loosely speaking is because we're really only looking at square matrices in $\mathbb{R}^{n\times n}$. The actual generalization would require us to look at matrices in $\mathbb{C}^{n\times n}$, where we would say every complex matrix can be broken up into a Hermitian and skew-Hermitian matrix. We won't be doing much with complex matrices in this class, and you'll rarely see things in the optimization literature dealing with complex matrices.

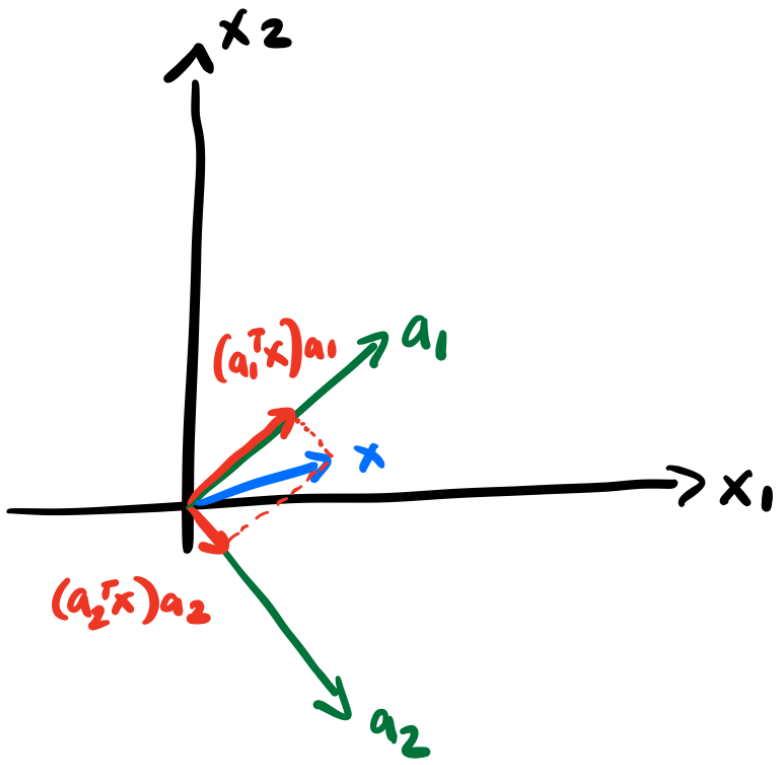

Figure 5: When (a_1, a_2) are orthonormal, for any vector x, we can write $x = \left(a_1^\intercal x\right) a_1 + \left(a_2^\intercal x\right) a_2$. Put in other words, there is $a_1^\intercal x$ of a_1 in x, and similarly a_2.

The basis being orthonormal is essential for this. If the basis vectors are not unit norm, then the inner product $a_i^\intercal x$ is not the projection onto the direction a_i, but rather a re-scaled version of it. If the basis vectors are not orthogonal, then x is not equal to the sum of the projections anymore, as visualized in Figure 6. With a basis that is not orthogonal, 'how much of a_1 is in x?' *depends* on the direction of a_2 in a complicated way.

Expressing the same facts with matrix terminology, what this means is if U is a matrix whose columns are orthonormal, then $U^{-1} = U^\intercal$, i.e. the inverse of U is its transpose. We say U is an **orthogonal** matrix if its columns are an orthonormal basis.

So, another thing that is nice about symmetric matrices is that there always exists an orthonormal basis of eigenvectors, or, equivalently, any symmetric Q can be written as $Q = U\Lambda U^\intercal$, where U is an orthogonal basis (whose columns are the eigenvectors of Q) and Λ is a diagonal matrix whose entries are the eigenvalues of Q.

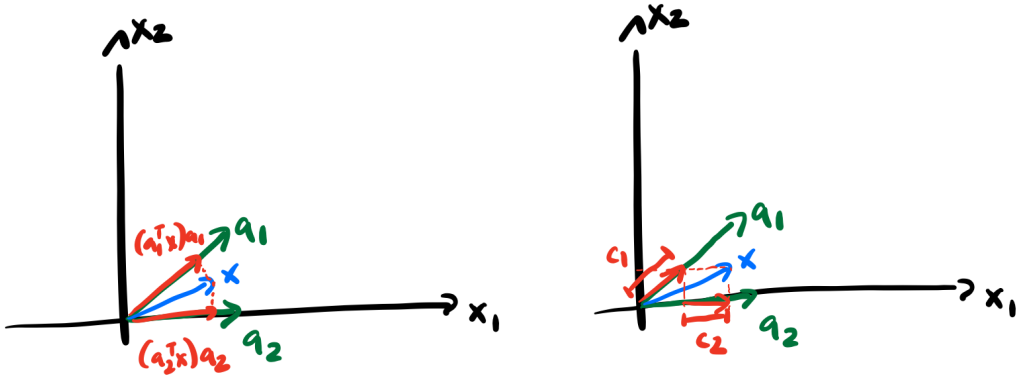

Figure 6: When (a_1, a_2) are not orthogonal, we can still project x onto the directions a_1 and a_2. However, x is no longer the sum of the two projections. We need to use a different method to find the weights c such that $x = c_1a_1 + c_2a_2$. This is why inverting a non-orthogonal matrix is much harder than inverting an orthogonal one. On the right, we see how much of a_1 and a_2 are actually in the linear combination $x = c_1a_1 + c_2a_2$.

So, what does this newfound intuition tell us about the action of Q? Well, let's break apart $Qx = U\Lambda U^\intercal x$ now. We start with a vector x. The first multiplication gives:

$$U^\intercal x = \begin{bmatrix} u_1^\intercal x \\ \vdots \\ u_n^\intercal x \end{bmatrix}$$

So, $U^\intercal x$ is a vector whose ith entry says 'what's the projection of x in the direction u_i?', which, for an orthonormal basis is the same as 'how much of u_i is in x with respect to the basis $\left(u_j\right)_j$?'

Next, we multiply by Λ:

$$\Lambda U^\intercal x = \begin{bmatrix} \lambda_1 u_1^\intercal x \\ \vdots \\ \lambda_n u_n^\intercal x \end{bmatrix}$$

So, Λ basically says: once you figure out how much of u_i is in x, scale that amount by λ_i.

Finally, the last multiplication tells us to take these as weights for how much to mix each of the basis vectors $(u_i)_i$:

$$Qx = U\Lambda U^\intercal x = \lambda_1 u_1^\intercal x \begin{bmatrix} | \\ u_1 \\ | \end{bmatrix} + \cdots + \lambda_n u_n^\intercal x \begin{bmatrix} | \\ u_n \\ | \end{bmatrix}$$

So, intuitively, we can think of Q acting on x as doing the following. We first calculate how much of each basis vector u_i is in x, giving us the scalar $u_i^\intercal x$. We scale that amount by λ_i, giving us the scalar $\lambda_i u_i^\intercal x$. Then, we mix together that amount of the basis vector u_i, for each i.

Similarly, we can also analyze the quadratic form:

$$x^\intercal Q x = (U^\intercal x)^\intercal \Lambda (U^\intercal x) = \sum_{i=1}^{n} \lambda_i \left(u_i^\intercal x\right)^2$$

The quadratic form evaluated at x says: for each u_i, calculate how much x points in the direction u_i, square it, scale it by λ_i, and then sum the results across i.

For a symmetric matrix Q, we say that Q is **positive definite** if all its eigenvalues are strictly positive, and **positive semi-definite** if all its eigenvalues are weakly positive. We will use the notation $Q \succ 0$ and $Q \succeq 0$ for this. Note that we can talk about positiveness of the eigenvalues because we already know they will be real. (It's not well-defined to say whether or a complex number is positive or not.)

When Q is positive definite, the level sets of $x \mapsto x^\intercal Q x$ are **ellipsoids**:

$$\{x : x^\intercal Q x = c\}$$

Here, $c > 0$ is some constant.

The gradient of the quadratic form $x \mapsto x^\intercal Q x$ is $2Qx$, which is visualized in Figure 7. Sometimes, folks use a convention where the quadratic form is $\frac{1}{2} x^\intercal Q x$ so the gradient is simply Qx.

Linear programs

With this, we can do a quick review of linear programs (LPs) as well.

A linear program is an optimization problem where we wish to choose $x \in \mathbb{R}^n$ to minimize $c^\intercal x$, subject to a bunch of linear inequality constraints, $a_i^\intercal x \geq b_i$ for $i = 1, \ldots, m$.

Well, we're now in a great spot to visualize LPs. We want to go as far as possible in the $-c$ direction, but have to lie in a bunch of given halfspaces. The intersection of a finite number of halfspaces is called a **polyhedron**. Looking at Figure 8, we can get an intuition for why optimizers of LPs sit in the corners of the polyhedron. The one situation where this can behave strangely is when the vector $-c$ points right at a face of the polyhedron, where all points on the face are optimizers and all have the same value for $c^\intercal x$. Even in this setting, there will be optimizers which sit in the corner of the polyhedron.

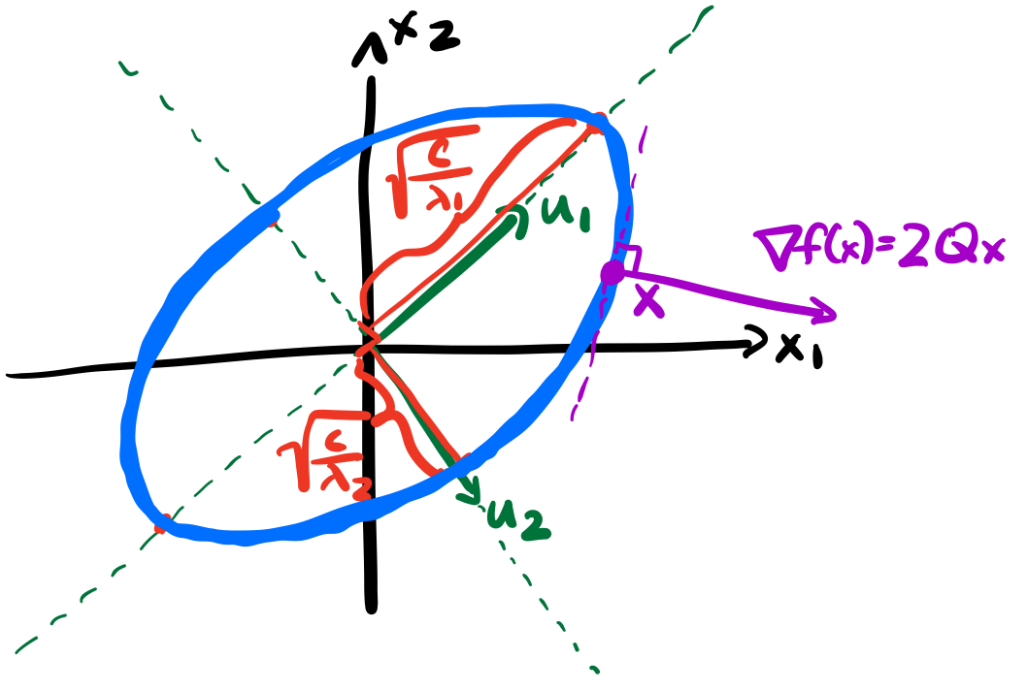

Figure 7: A visualization of the ellipsoid $\{x : x^\intercal Qx = c\}$, where λ_1 and λ_2 are eigenvalues with orthonormal eigenvectors u_1 and u_2, respectively. We can see that the basis (u_1, u_2) effectively just rotates our axes, and we can 'ignore' x_1 and x_2 in this visualization once we rotate to the (u_1, u_2) axes. The gradient of $x \mapsto x^\intercal Qx$ is also visualized here, and is the vector that is the normal to the level set, pointing in the direction of greatest increase for the quadratic form.

This motivates **simplex methods**, which are algorithms that try to find the 'best' next corner to check for optimality. (Duality allows us to certify a corner is optimal when we do find it.)

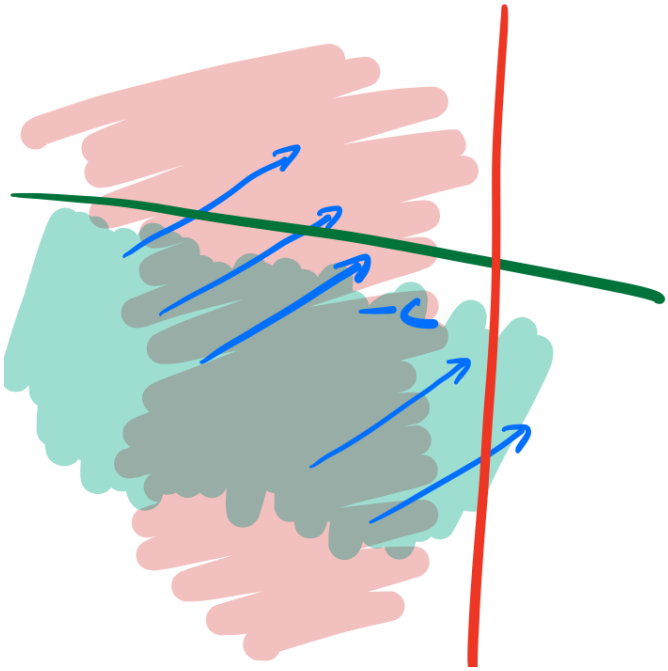

Figure 8: Visually, the blue $-c$ lines tell us which direction we want to push towards. The green and red lines represent two linear inequality constraints, stating that we must lie a particular side of the hyperplane.

7 OPTIMIZATION

Unconstrained optimization

As another review, let's consider the problem of unconstrained optimization, and review some familiar concepts.

Let $f : \mathbb{R}^n \to \mathbb{R}$. We wish to find the minimum of $\{f(x) : x \in \mathbb{R}^n\} \subseteq \mathbb{R}$, often denoted:

$$\min_x f(x)$$

The following should be familiar from your pre-requisites, so I won't prove this in formal detail; however, it's a good exercise to try and prove this rigorously, using the formal definitions of limits we gave in the real analysis review.

Proposition 1. *If x is a local minimum of f and f is differentiable at x, then $\nabla f(x) = 0$.*

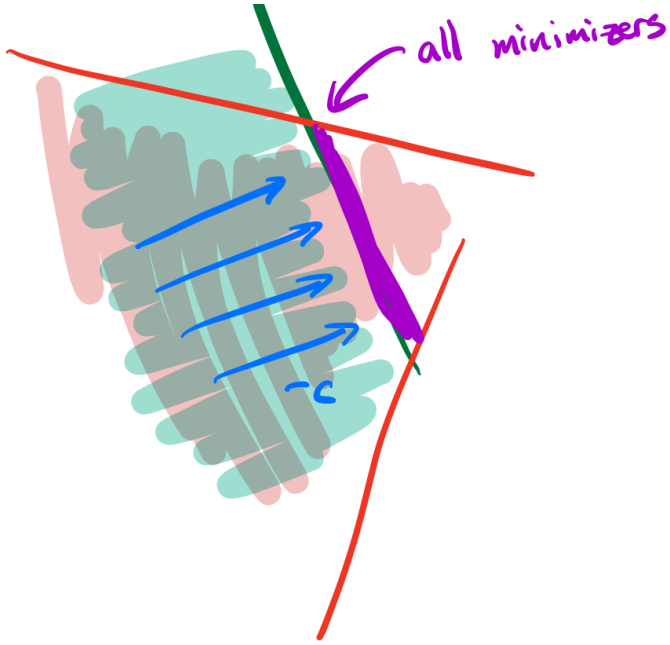

Figure 9: If one of the linear inequality constraints lines up exactly with the cost vector, there may be optimizers which are not in the corners of the polyhedron; in this case, there is an entire face of optimizers and the value $c^\intercal x$ is the same for all points on the face.

This is a *necessary* condition of optimality, i.e. if x is a minimum, then it necessarily must satisfy this property. Put another way: the set of optimal points is a subset of the points that satisfy the necessary conditions. (Knowing x is an optimal point allows us to conclude the necessary conditions.) In practice, one of the ways we can use this is we can restrict our search for optimum points to looking among the set of points that satisfy the necessary conditions. For example, if we have a function f and we are able to calculate $\nabla f(x)$, sometimes we can just check all the points x such that $\nabla f(x) = 0$.

The converse would be a *sufficient* condition for optimality, i.e. if we know these conditions are true, then we can conclude that x is a minimum. There's sufficient information to conclude optimality. Put another way: the set of points that satisfy the sufficient condition is a subset of the optimal points. (Knowing we're in the sufficient condition set is enough to conclude we're in the set of optimal points.) In practice, we can use sufficient conditions to certify that a point is optimal.

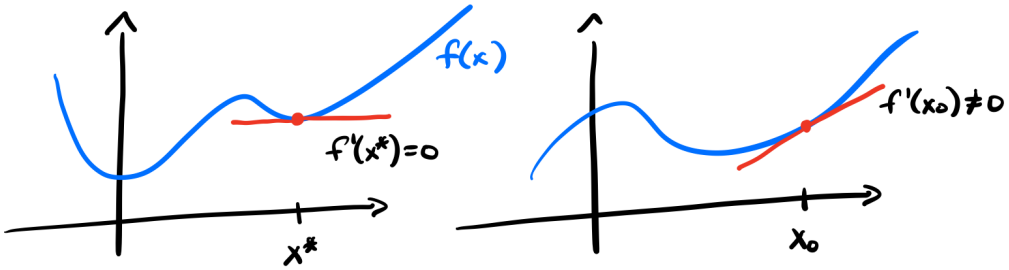

Figure 10: Consider $f : \mathbb{R} \to \mathbb{R}$. We've pictured the graph of f, which is the set of points $\{(x, f(x)) : x \in \mathbb{R}\}$. The slope of the line tangent to the graph at a point x is the derivative at x. If the derivative is positive, we can reduce $f(x)$ by moving slightly to the left; if the derivative is negative, we can reduce $f(x)$ by moving slightly to the right. For general $f : \mathbb{R}^n \to \mathbb{R}$, if we move in the direction of $-\nabla f(x)$ we can reduce $f(x)$.

Quadratic costs

Suppose we're trying to find $x \in \mathbb{R}^n$ which minimizes:

$$x^\intercal Q x + c^\intercal x + b$$

Here, $Q \succ 0$, $c \in \mathbb{R}^n$, and $b \in \mathbb{R}$ are all given. We can simply set the gradient to zero to get the solution:

$$2Qx + c = 0$$

Since Q is positive definite, we can invert it to find the answer:

$$x = -\frac{1}{2} Q^{-1} c$$

Geometrically, we can also think of this as finding an x such that $2Qx = -c$, described in Figure 11.

Now, suppose we want to minimize a quadratic function subject to a linear equality constraint:

$$\begin{aligned} \min_x \quad & x^\intercal Q x \\ \text{s.t.} \quad & c^\intercal x = b \end{aligned}$$

Let $f(x) = x^\intercal Q x$ and $h(x) = c^\intercal x - b$, so the problem is to minimize $f(x)$ subject to $h(x) = 0$.

The smallest level set we can get to on the hyperplane is tangent to the hyperplane, i.e. the gradient of $x^\intercal Q x$ is a scaled version of c, as shown in Figure 12. Formally, this means there exists constant $\lambda \in \mathbb{R}$ such that $\nabla f(x) + \lambda \nabla h(x) = 0$.

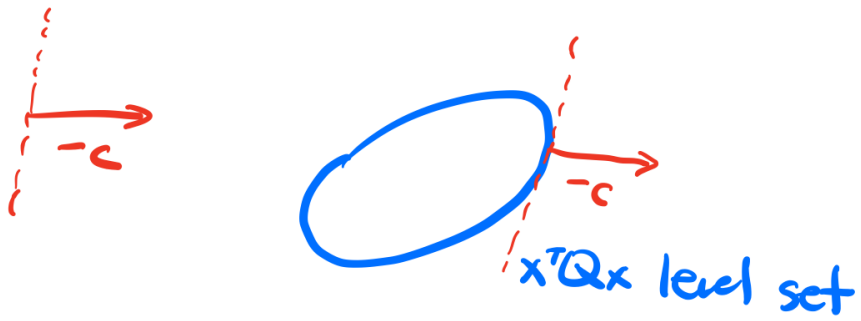

Figure 11: When minimizing $x^\intercal Qx + c^\intercal x + b$, we can think of taking an ellipsoid $\{x : x^\intercal Qx = b\}$ and finding which point on the ellipsoid has $-c$ as its normal vector, so that $2Qx$ and $-c$ are pointing in the same direction. Then we can scale x accordingly such that $2Qx = -c$.

Now, suppose we want to minimize a quadratic function subject to a linear inequality constraint:

$$\begin{aligned} \min_x \ & x^\intercal Qx \\ \text{s.t. } & c^\intercal x \leq b \end{aligned}$$

Let $f(x) = x^\intercal Qx$ and $g(x) = c^\intercal x - b$, so the problem is to minimize $f(x)$ subject to $g(x) \leq 0$.

Geometrically, if zero is possible (i.e. $g(0) \leq 0$), then it will be an optimal solution, in which case $\nabla f(x) = 0$. If zero is not a feasible point, then the smallest level set we can get to is the one that has $-c$ as a normal vector. Thus, there exists some constant $\lambda \geq 0$ such that $\nabla f(x) = -\lambda \nabla g(x)$, or $\nabla f(x) + \lambda \nabla g(x) = 0$.

We'll develop this further but it should build some intuition for the notion of Kalman-Kuhn-Tucker conditions for optimality, which we will discuss in greater detail when we cover Lagrangian duality.

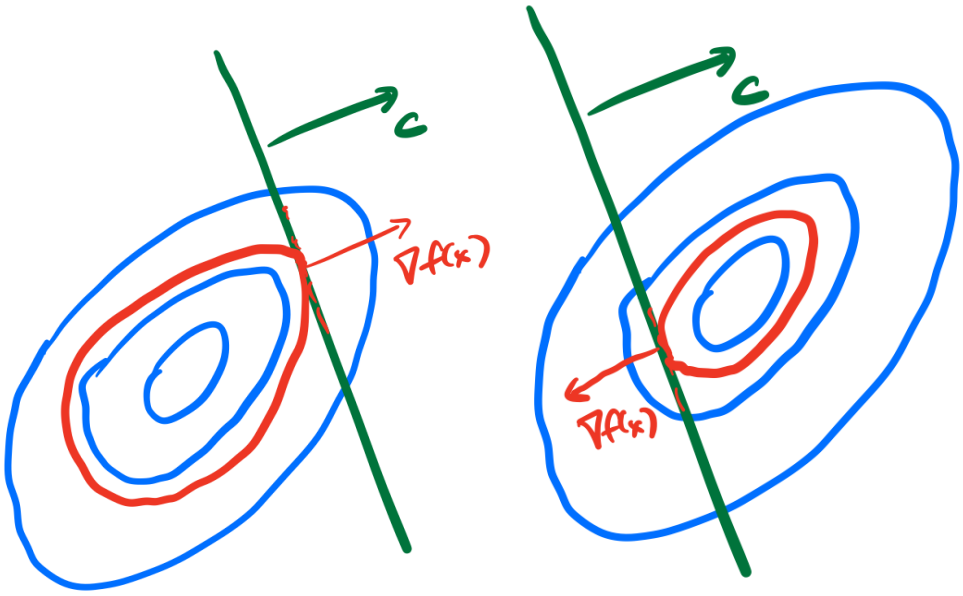

Figure 12: The level sets of $f(x) = x^\intercal Q x$ and the hyperplane $h(x) = c^\intercal x - b = 0$. At optimum, the level set of f will be tangent to the hyperplane. Note that λ can be positive or negative.

PART III

BASIC NOTIONS IN CONVEX ANALYSIS AND OPTIMIZATION

8
CONVEX SETS, CONVEX HULLS, AND CONVEX COMBINATIONS

Related reading: Boyd and Vandenberghe [2004] Ch. 2-3.

We'll be living in $\mathbb{R}^n$ for a lot of this discussion. This section will be a barrage of definitions, but we're introducing the cast of characters that we'll be working with for the rest of the semester.

Given two points x and y, the **line segment between x and y** is the set of all points:

$$\{\theta x + (1-\theta)\, y : \theta \in [0,1]\}$$

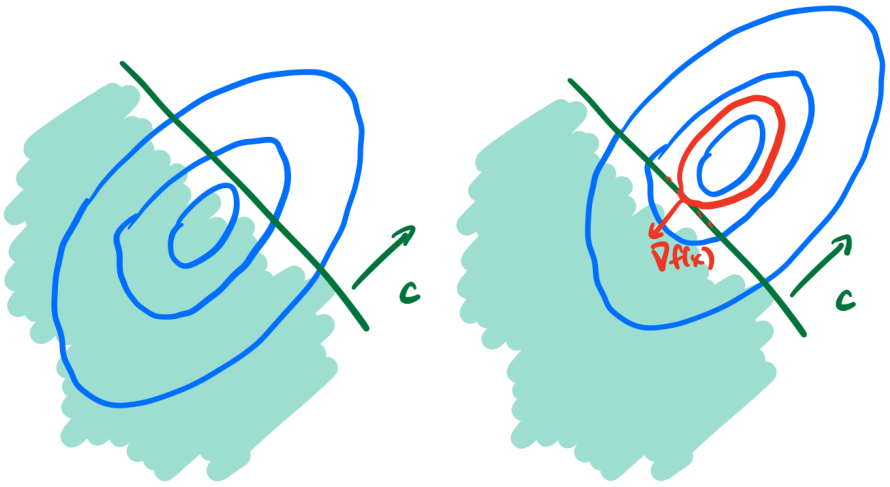

Figure 13: The level sets of $f(x) = x^\intercal Q x$ and the halfspace $g(x) = c^\intercal x - b \leq 0$. If zero is feasible, it is optimal, and if it is not, then the boundary hyperplane of the halfspace is tangent to the gradient of f.

Looking at the expression $\theta x + (1-\theta)\, y$, we can see that at $\theta = 0$ we're at y and at $\theta = 1$ we're at x, and all the in-between values of θ put us somewhere in between the two points.

Another common way to write the same expression is $y + \theta\,(x - y)$, which more clearly shows that we start at y at $\theta = 0$ and go in the direction $x - y$ as θ increases.

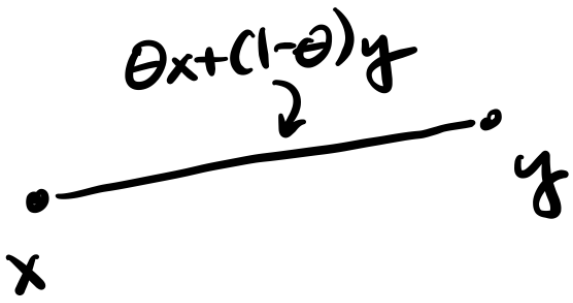

Figure 14: A line segment connecting x and y.

We say C is a **convex set** if the line segment connecting x and y is in C for all $x, y \in C$. If I take any two points in C, I can go from one to another in a straight line without leaving the set C. In a sense, it means we can 'see' every point in the set from any other point.

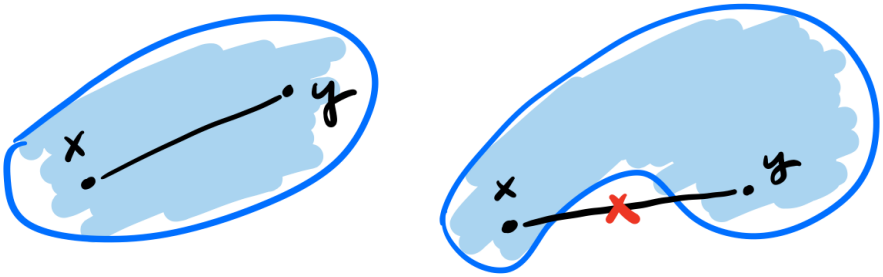

Figure 15: On the left, a convex set: if we draw the line segment between any two points, we remain in the set. On the right, a non-convex set: as an example, for these two points shown, the line segment is not contained within the set.

This is where we may get really pedantic about little details, such as whether or not the boundary is included. For example, see Figure 16.

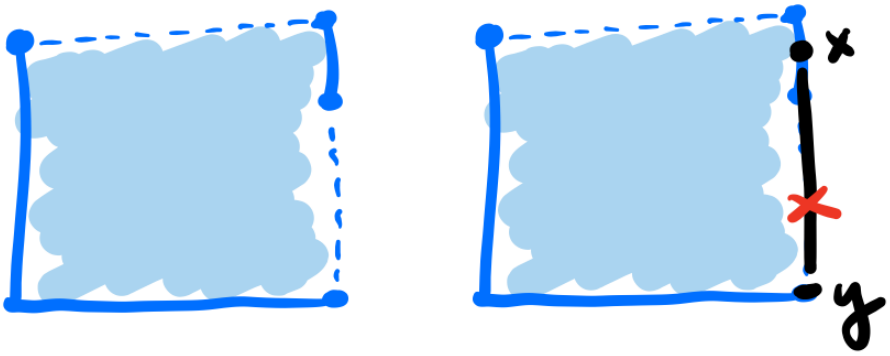

Figure 16: This is an example of a set that is not convex just because of issues with the boundary. It only contains some points of its boundary, and, as such, if we draw the line depicted on the right, we technically leave the set, despite being arbitrarily close to actually being in the set.

Note that the empty set is always convex, the entire space is always convex, and any singleton set is always convex.

So, for emphasis, we currently should have both an intuitive understanding of convex sets (it looks round-ish and can't have any nooks in it) as well as a formal understanding (if I take any two points x and y in the set I have a property they satisfy).

Let's use these two understandings to prove an important fact about convex sets.

Proposition 2. *The intersection of an arbitrary set of convex sets is convex.*

If we try and draw this idea out, basically if you put a bunch of round-ish shapes down and look at where they overlap, then you'll still wind up with something round-ish. Hopefully you can kind of picture it; Figure 17 shows one example.

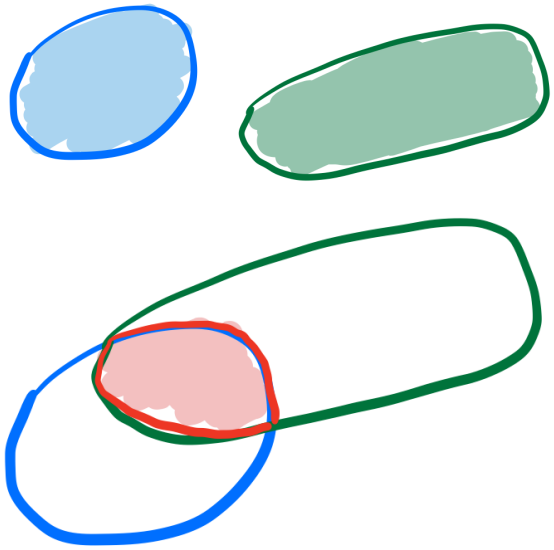

Figure 17: The intersection of convex sets is convex.

Now, we can reason about it formally as well. Let I be an arbitrary index set, with C_i a convex set for every $i \in I$. We wish to show that $C = \cap_{i \in I} C_i$ is convex.

What is the definition of convexity? Well, for any points x and y in C, we want to show the line segment connecting them is in C.

So, let's take any x and y in C and try to show that. Since x and y are in C, this means x and y are in C_i for every i, by definition of the intersection.

Now, fix any $i \in I$. Since C_i is convex, this means the line segment between x and y is in C_i. This held for arbitrary i, so we've shown that the line segment containing x and y are in C_i for all $i \in I$. Thus, again by the definition of the intersection, that means the line segment is in $\cap_{i \in I} C_i$. QED.

So to recap the proof, we wanted to show $\forall x, y \in C = \cap_{i \in I}, \forall \theta \in [0, 1], \theta x + (1 - \theta) y \in C$. We first fixed an arbitrary pair of $x, y \in C$ and then needed to show $\forall \theta \in [0, 1], \theta x + (1 - \theta) y \in C$. To show that these points were in C for all $\theta \in [0, 1]$, we needed to use the definition of C. Since C is the intersection

of a bunch of sets, we need to show that the points were in C_i for all $i \in I$. Invoking convexity for each individual $i \in I$ let us conclude that it was in C_i for that particular i, and since we can do this for every i, we conclude it's in C_i for all i.[1]

This is... *much* more verbose than you will typically see in a paper. You may see something as succinct as "Since the line segment is in each of the C_i it will be in C." If you're already comfortable about mathematical reasoning and nested quantifiers and such, feel free to be this succinct. However, if you have any uncertainties, or if you ever look at a proof and are not entirely sure if the reasoning is valid, I encourage you to be overly verbose and pedantic at the beginning, in order to build up that mathematical intuition and maturity. I'll only do a few more proofs in this overly verbose way as a guidepost, but after that, you'll be expected to more or less pursue the development of rigorous sensibilities on your own.

Some examples of convex sets include:

(a) Any normed ball $\{x : \|x - x_0\| \leq r\}$.

(b) Ellipsoids.

(c) The **norm cone**, given by $\{(x, t) : \|x\| \leq t\}$.

(d) Polyhedra.

Given some set $A \subseteq \mathbb{R}^n$, we can define the **convex hull of** A, denoted $\text{conv}\,(A)$, as the 'smallest' convex set containing A. Here, I've put 'smallest' in scare-quotes because we need to rigorously define the notion of smallest. There are *several* mathematical notions of 'smallness' which do not necessarily align. In this context, we think of a set A as 'smaller' than B if $A \subseteq B$, i.e. A contains weakly fewer elements than B.

Using this notion of 'smallness', we can define the convex hull of A is the intersection of all convex sets containing A, sometimes denoted:

$$\text{conv}\,(A) = \bigcap_{C \supseteq A : C \text{ convex}} C$$

Since the intersection of convex sets is convex, we know that the convex hull will be convex. Intuitively, you can visualize the convex hull as 'adding all the line segments between points'.

[1]For a more detailed introduction to proof structures, please see the 'how to math' document I've included with the course materials.

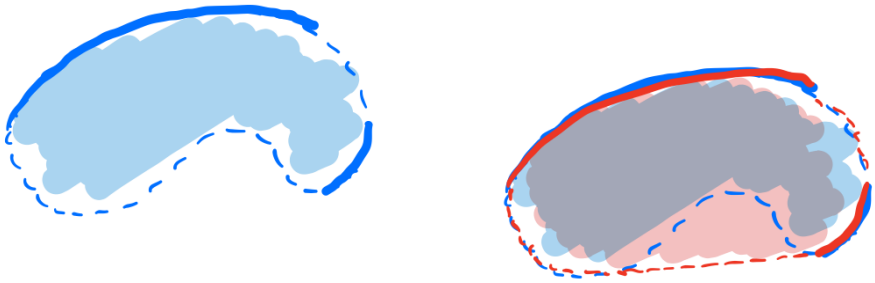

Figure 18: On the right, we see the convex hull of the blue set on the left.

You'll often use the concept of convex hull when you want to 'make a set convex'. So, for example, if you have an optimization problem where your cost function is convex but your feasible set is not convex, you may define a relaxed version of your original problem where the feasible set is replaced with the convex hull of the feasible set. This relaxed version will be a convex optimization problem, and usually easier to solve. This is a technique used frequently in research, and, when one does this, the 'hard work' consists in showing the relaxation is useful in some way. For example, one can sometimes show that, under certain conditions, the optimal solution is actually in the original feasible set, in which case you have actually solved the original non-convex optimization problem. (This is a bit of foreshadowing; if you are not familiar with the terms 'feasible set' or such, please bookmark this paragraph and come back to it after we cover these terms.)

Another important concept is the notion of convex combinations. For a finite set of points $x_1, \ldots, x_n$, we define a **convex combination of** $(x_1, \ldots, x_n)$ as $\lambda_1 x_1 + \cdots + \lambda_n x_n$ for some $\lambda \geq 0$ and $\sum_{i=1}^n \lambda_i = 1$.[1] So, we're combining non-negative amounts of our n ingredients together, using a total amount of 1 unit of all of them. Visually, you can think of this as all the points 'in between' the set of points you're mixing. This is a generalization of the $\theta x + (1-\theta)\, y$ expression we saw before; the latter is just the convex combination of 2 points.

Okay, so let's use this definition to prove some stuff. (Proving small things like this with a new definition is like the mathematician's version of playing with a new toy.) Let's show the following:

[1]This is the first time in the notes I've used the notation $\lambda \geq 0$, where $\lambda \in \mathbb{R}^n$ is a vector. Throughout this course, unless otherwise noted, this simply means that every component of λ is non-negative, i.e. $\lambda_i \geq 0$ for all $i \in \{1, \ldots, n\}$.

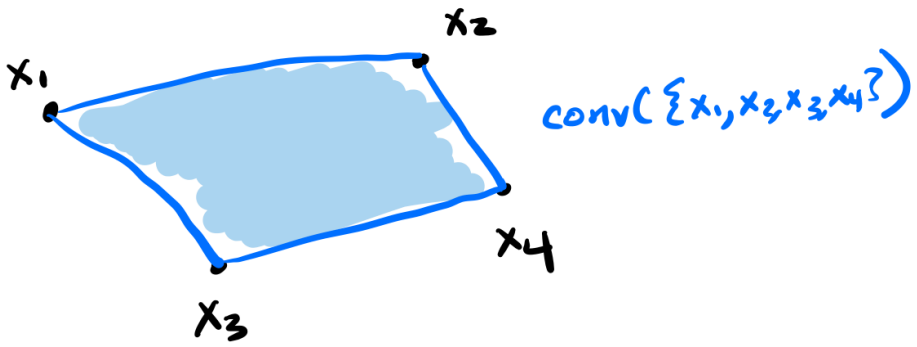

Figure 19: The convex hull can be visualized as all the points 'between' a set of points.

Proposition 3. *If C is a convex set, then it contains all convex combinations of points in C, i.e. for any finite number of points n, points $x_1 \in C, \ldots, x_n \in C$, and valid weights λ, the point $x = \sum_{i=1}^{n} \lambda_i x_i$ is in C.*

What's the intuition here? Well, if we take a convex combination of points, we're looking at something that's in the 'middle' of all of them. In contrast, the condition that defines a convex set looks at line segments between two points at a time, i.e. stuff that's in the 'middle' of two points. To show that a convex set actually has convex combinations, we want to show that a convex combination of n points can be recreated by sequentially taking points on the line segment connecting them.

For example, suppose we have a convex combination of 3 points in C: $\lambda_1 x_1 + \lambda_2 x_2 + \lambda_3 x_3$. All the points on the line segment connecting x_1 and x_2 are in C, by the definition of convexity. So, we know that the following point is in C:

$$y_2 = \frac{\lambda_1}{\lambda_1 + \lambda_2} x_1 + \frac{\lambda_2}{\lambda_1 + \lambda_2} x_2 \in C$$

Since that point is in C, we know the line segment between y_2 and x_3 is in C. Note that $\theta = \lambda_1 + \lambda_2$ and $1 - \theta = \lambda_3$ gives us a valid set of weights for two points. So the following point must be in C:

$$\lambda_1 x_1 + \lambda_2 x_2 + \lambda_3 x_3 = (\lambda_1 + \lambda_2)\, y_2 + \lambda_3 x_3 \in C$$

This is what we wanted to show: the convex combination of points itself is actually in C. The argument here is visualized in Figure 20. Note what we were doing here: we were given a convex combination of 3 points and we showed it could be recreated by taking convex combinations of 2 points sequentially. I'll leave it as an exercise to show the general case for n points: you can do so using a *proof by induction*.

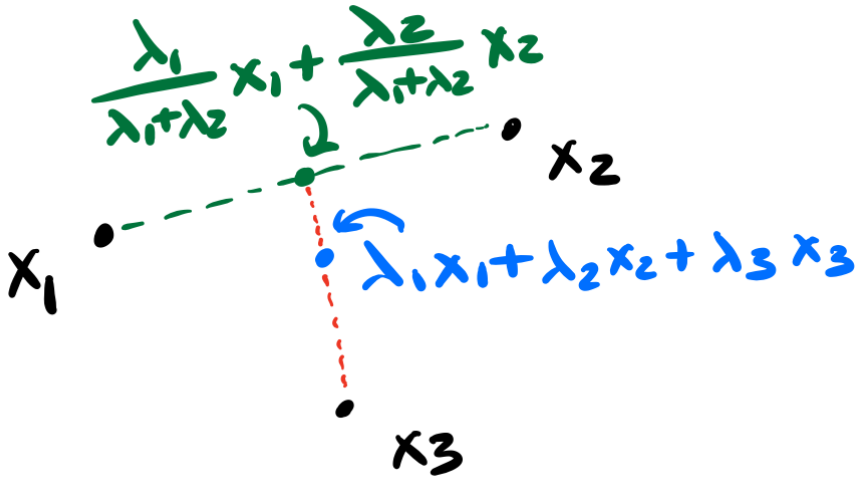

Figure 20: The blue dot represents a convex combination of x_1, x_2, and x_3. We can recreate this point by first taking a convex combination of x_1 and x_2, and then taking a convex combination of the result with x_3.

Great, so not only did we get some exercise with this new definition, we actually proved a pretty useful fact!

Next, let's use this definition again and do another proof with it. Let's prove the following:

Proposition 4. *The convex hull of A (defined as the intersection of all convex sets containing A) is equal to the set of all convex combinations of points in A.*

When I say the convex combination of all points in A, I mean the points x such that $x = \sum_{i=1}^{n} \lambda_i x_i$ for some number n, some valid weights λ, and points $x_1 \in A, \ldots, x_n \in A$. Note that n isn't a fixed number here, i.e. we're not just looking at all convex combinations of n points, but we're looking at all convex combinations, of any arbitrary **finite** number of points.

This will be the last overly pedantic proof I do in these notes. Afterward, I will start to use common conventions and shorthand you'll see in proofs. If you find it useful, it is worth the time investment to keep re-writing these proofs in a very verbose and pedantic way. It's the mathematical thinking equivalent of practicing your scales on an instrument: after a while, it becomes second nature and you can start to comfortably express yourself through the notes. However, before that, it can be mechanical and feel a little bit like a grind.

Before we begin, let's note why this proof may be hard to show formally. First, when we define the convex hull, it is the arbitrary intersection of convex sets: we are intersecting an uncountably infinite collection of sets, and each set itself often contains uncountably infinite many points. When we consider convex combinations, we're only taking finitely many points out at a time. The thing that's not trivial and makes this proof hard is that we somehow have to take the finiteness of convex combinations and conclude the infiniteness of the convex hull.

Okay, let's do this. Let $\widehat{C}$ denote the set of all convex combinations of elements of A. We're being asked to show two sets are equal, i.e. $\operatorname{conv}(A) = \widehat{C}$, so we have to show two inclusions: we need to show that the convex hull is contained in the set of all convex combinations, i.e. $\operatorname{conv}(A) \subseteq \widehat{C}$, and vice versa, i.e. $\widehat{C} \subseteq \operatorname{conv}(A)$.

As hinted by the note above, one direction is easy. Let $x \in \widehat{C}$ be any convex combination of points in A, i.e. $x = \sum_{i=1}^{n} \lambda_i x_i$. We want to show that x is on the convex hull of A, which is defined as the intersection of all convex sets containing A. Thus, we want to show that x is in every convex set containing A. Let C be any convex set containing A. Since C contains A, it contains each point $x_i \in A$. So, x is a convex combination of points in C. As we just showed, a convex set contains all convex combinations of its points, so $x \in C$. This held for any convex $C \supseteq A$, so it is in $\operatorname{conv}(A) = \cap_{C \subseteq A : C \text{ convex}} C$. Thus, $\widehat{C} \subseteq \operatorname{conv}(A)$.

Now, the other direction. We're trying to show $\operatorname{conv}(A) \subseteq \widehat{C}$, but note that $\operatorname{conv}(A)$ is defined as an intersection. This means for any convex set C containing A, we have $\operatorname{conv}(A) \subseteq C$. So, it will suffice to show that there exists some convex set C containing A such that $C \subseteq \widehat{C}$, the set of all convex combinations of points in A, since this would give us the chain $\operatorname{conv}(A) \subseteq C \subseteq \widehat{C}$.

In fact, we'll actually be able to show that $\widehat{C}$ itself is a convex set, which, by the argument above, is enough to show that $\operatorname{conv}(A) \subseteq \widehat{C}$. What is the definition of convexity? Well, we need to take any points $x_1, x_2 \in \widehat{C}$ and show that any convex combination of them is in $\widehat{C}$.

Okay, so let x_1, x_2 be any two points in $\widehat{C}$. Since $\widehat{C}$ is defined as the set of convex combinations of elements of A, this means that there exists points $y_1^1, \ldots, y_m^1$ in A and valid weights λ^i such that $x_1 = \sum_{i=1}^{m} \lambda_i^1 y_i^1$ and similarly points $y_1^2, \ldots, y_n^2$ and weights λ^2 such that $x_2 = \sum_{i=1}^{n} \lambda_i^2 y_i^2$. For any convex combination of x_1 and x_2, we can write:

$$\theta x_1 + (1-\theta) x_2 = \theta \sum_{i=1}^{m} \lambda_i^1 y_i^1 + (1-\theta) \sum_{i=1}^{n} \lambda_i^2 y_i^2$$

Note that the weights $\left(\theta\lambda_1^1, \ldots, \theta\lambda_m^1, (1-\theta)\,\lambda_1^2, \ldots, (1-\theta)\,\lambda_n^2\right)$ give a valid set of weights for a convex combination of $m+n$ points, so we've written $\theta x_1 + (1-\theta)\,x_2$ as a convex combination of elements of A. Thus, we've shown that for any $x_1, x_2 \in \widehat{C}$, $\theta x_1 + (1-\theta)\,x_2 \in \widehat{C}$ for all $\theta \in [0,1]$, i.e. $\widehat{C}$ is convex. This implies that $\text{conv}\,(A) \subseteq \widehat{C}$.

And that's it: we've shown $\text{conv}\,(A) = \widehat{C}$ through both inclusions. QED.

Okay, so we've defined convex sets, convex hulls, and convex combinations. We can use these concepts to define a **convex function**. Simply put, a convex function is a function whose epigraph is a convex set.

For a function $f : A \to B$, the **graph of** f is:

$$\text{graph} f = \{(x,y) : x \in A, y \in B, y = f\,(x)\} \subseteq A \times B$$

When $A = \mathbb{R}$ and $B = \mathbb{R}$, the graph is in $\mathbb{R}^2$, and that's what you typically think of when you draw x- and y-axes, and draw a line through all the points such that $y = f\,(x)$.

The **epigraph of** f is:

$$\text{epi} f = \{(x,y) : x \in A, y \in B, f\,(x) \leq y\} \subseteq A \times B$$

The epigraph is simply all the points above a graph. In this class, we will typically consider the domains that are a subset of $\mathbb{R}^n$, i.e. $A \subseteq \mathbb{R}^n$ and co-domains that are $\mathbb{R}$. As such, the epigraph will live in the space $\mathbb{R}^{n+1}$.

Note that:

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

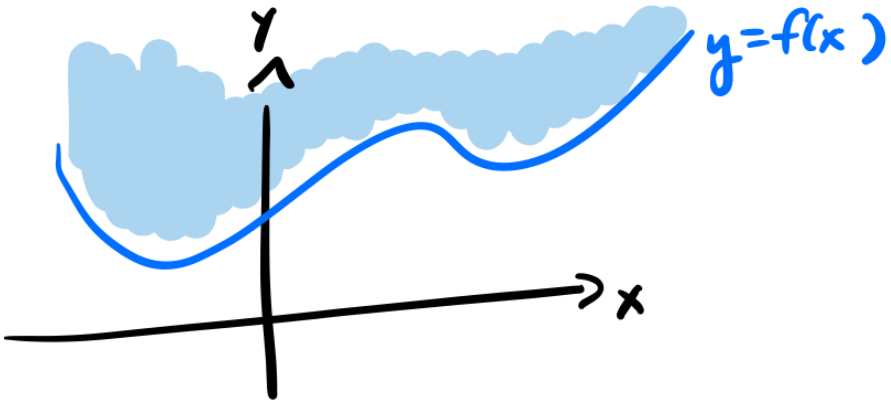

Figure 21: A visualization of the graph (dark blue line) and the epigraph (dark blue line and light blue shading) of a function $f : \mathbb{R} \to \mathbb{R}$.

Suppose $f : A \to \mathbb{R}$, where $A \subseteq \mathbb{R}^n$. We say f is a **convex function** if epi$f$ is a convex set.

You may have seen other definitions of convexity. In your homework, you'll be asked to show the following.

Proposition 5. *If $f : A \to \mathbb{R}$ is a convex function (as defined by the epigraph being a convex set), then the domain A is a convex set, and the following holds for all $x_1, x_2 \in A$ and $\lambda \in [0, 1]$.*

$$\lambda f(x_1) + (1 - \lambda)(f(x_2)) \geq f(\lambda x_1 + (1 - \lambda) x_2)$$

We can also define a **concave function**: f is concave if $-f$ is convex.

Let's move on to the star of the show in convex analysis: duality.

9
HYPERPLANES IN CONVEX ANALYSIS

Let's begin to take our first look at duality. For convex analysis, the duality often comes from hyperplanes, so a strong intuition in this will help a lot for understanding that which is to come.

Proposition 6. *A closed, convex set is the intersection of all the halfspaces that contain it.*

Formally, let $S \subseteq \mathbb{R}^n$ be a closed, convex set. Let $\mathcal{H} = \{H \subseteq \mathbb{R}^n : H \text{ is a halfspace}, S \subseteq H\} \subseteq 2^{\mathbb{R}^n}$ be the set of halfspaces which contain S. Then:

$$S = \bigcap_{H \in \mathcal{H}} H$$

Note that this claim is that one set equals the intersection of another set, which tells us how to structure the proof.

One direction is easy. Suppose $x \in S$. Then $x \in H$ for every $H \in \mathcal{H}$, since $S \subseteq H$ for every $H \in \mathcal{H}$. Thus, x is in the intersection. So $S \subseteq \cap_{H \in \mathcal{H}} H$ is easy to show.

The other direction can be shown with a separating hyperplane argument. Before we formally go into those details, let's talk about why this result is useful to begin with.

The key thing here is that the two sets are *equal*. What that means is that we can think of closed, convex sets in two equivalent ways. On one hand, we can think of S as the set of points in it, e.g. $S = \{x : x \in S\}$. On the other hand, can think of S as the hyperplanes that it sits squarely on one side of.

This may superficially seem not that interesting, but we'll spend a lot of time digging into this to see the surprising consequences of it.

As a preview, let's apply Proposition 6 to the epigraph of a convex function. As we defined before, a function $f : A \to \mathbb{R}$ (where $A \subseteq \mathbb{R}^n$) is a convex function if its epigraph $\text{epi} f \subseteq \mathbb{R}^{n+1}$ is a convex set. Furthermore, f is a **closed convex function** if the epigraph is a closed, convex set. For this section, we'll assume f is a closed convex function.[1]

[1]Throughout this semester, I'm going to be ignoring some of the more pedantic details you may find in textbooks such as Rockafellar [1993]. Interested students can look up the definition of a *proper convex function*. Much like Boyd and Vandenberghe [2004], I'm going to ignore degenerate cases such as where the domain is empty.

Loosely speaking, every halfspace containing the epigraph has to be 'pointed up' in its last coordinate, since the epigraph is all the points above the graph. When f is continuously differentiable, the hyperplanes that push up to the edge of the epigraph are tangent to the epigraph, and are actually the linear approximations, giving the equation $f(x) \geq f(x_0) + \nabla f(x_0)^\intercal (x - x_0)$. (This isn't a proof of this statement, but just an intuition for it.)

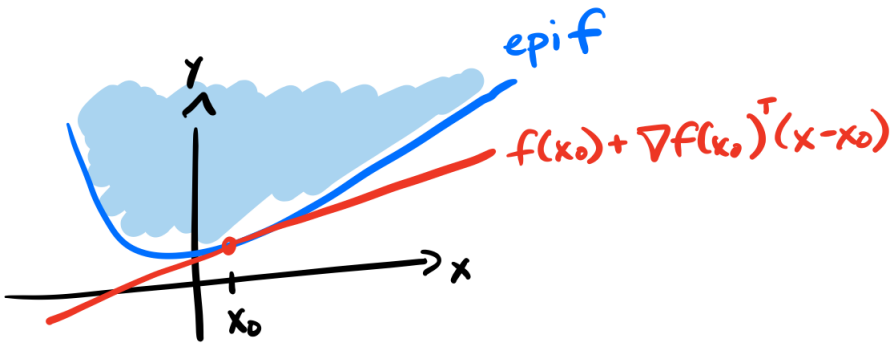

Figure 22: When f is continuously differentiable, the hyperplanes that are tangent to the epigraph are actually the linear approximations. The fact that the epigraph lies on one side of this hyperplane states that the convex function sits above its linear approximation.

When we apply Proposition 6 to epigraphs, we get the following for closed convex functions f: f is equal to the supremum of all linear functions that lower bound f. Formally, let $\mathcal{F} = \{(c, b) : c^\intercal x + b \leq f(x) \text{ for all } x \in A\}$, and then:

$$f(x) = \sup_{(c,b)\in\mathcal{F}} c^\intercal x + b \tag{1}$$

This is the idea underlying the **convex conjugate** (or **Fenchel conjugate** or **Legendre transform**): we start to treat slopes as the domain rather than points. Intuitively, the conjugates ask: 'Given a slope, how far up can I push the hyperplane until it touches the epigraph?' This is a dual way of thinking to the typical differentiation, which asks: 'Given a point, what is the slope?' I'm getting a little ahead of myself, but we'll cover this soon.

So, let's talk about one way in which this insight is useful for actual algorithms: **cutting planes methods**. When applied to minimizing a convex function, the idea is to approximate the supremum in Equation (1) with a finite number of linear functions:

$$f(x) \approx \max_i c_i^\intercal x + b_i$$

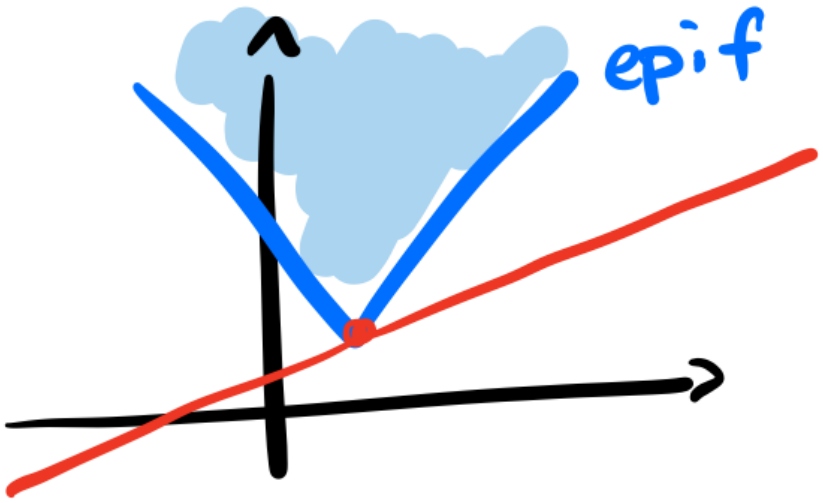

Figure 23: Things get a little weirder when f is convex but not differentiable, as visualized here. The generalization of gradients to convex functions are *subgradients*, which we'll talk about later in the semester.

If we are attempting to minimize f, we have an approximation of $\min f$ with:

$$\begin{aligned} \min_{(x,t)} \quad & t \\ \text{s.t.} \quad & t \geq c_i^\intercal x + b_i \text{ for all } i \end{aligned}$$

This gives us a linear program, which we have tools to solve already. After each iteration, we intelligently add some more linear under-approximations of f to the maximum and repeat, getting better and better at approximating f with the maximum of a finite number of functions. In this setting, cutting planes methods vary based on how additional linear under-approximations are chosen. However, as I already hinted at, sometimes insights are more clear in convex analysis when we think of things in terms of sets rather than functions (such as the epigraph), so we'll present cutting planes methods more generally in a bit.

Before that, let's discuss separating hyperplanes. From real analysis, let us recall the definition of a distance between two sets:[1]

$$d(A, B) = \inf\{\|x - y\| : x \in A, y \in B\}$$

As per our previous discussion, this infimum may or may not be a minimum (e.g. there may or may not exist a pair (x, y) in $A \times B$ that attains the infimum). The distance between two disjoint sets may be zero, even if they are both closed, as shown in Figure 24.

Now, let's show some separating hyperplane theorems. First, let's look at case that assumes away the aforementioned issues with the infimum.

Proposition 7. *Let A and B be two convex sets in $\mathbb{R}^n$ with $d(A, B) > 0$, and suppose there exists points $x \in A$ and $y \in B$ such that $\|x - y\| = d(A, B)$.*

Then, there exists $c \in \mathbb{R}^n$ and $b \in \mathbb{R}$ such that $c^\intercal x + b \leq 0$ for all $x \in A$ and $c^\intercal y + b \geq 0$ for all $y \in B$.

Okay, let's draw a picture to build some intuition for this. See Figure 25.

What's the equation for the hyperplane in Figure 25? Well, we can take $c = y - x$, which is the vector from x to y. At y, we would have $c^\intercal y = \|y\|^2 - x^\intercal y$ and at x we would have $c^\intercal x = x^\intercal y - \|x\|^2$. The halfway point between x and y would be $b = \frac{\|y\|^2 - \|x\|^2}{2}$. So, we have an equation for our candidate hyperplane: $z \mapsto c^\intercal z + b$. Note that this hyperplane can also be written as $z \mapsto (y - x)^\intercal \left(z - \frac{y+x}{2}\right)$, viewing this as first shifted z to the midpoint of the hyperplane. This also can be written as $z \mapsto c^\intercal (z - [x + c/2])$. Let's show that this actually separates A and B as claimed by the proposition.

[1]Here's a minor note on notation. What you'll typically see amongst engineers is that $|\cdot|$ is the absolute value of a scalar, $\|\cdot\|$ is a norm of a vector, be it an finite- or infinite-dimensional underlying vector space. Mathematicians typically use $|\cdot|$ to denote the norm of a vector in a finite-dimensional space (including scalars), and $\|\cdot\|$ is reserved only for infinite-dimensional spaces. For this class, I'll try to use the former convention but I may lapse and accidentally use the latter. In general, you should be alert as to what the arguments are for any norm and be aware of what is the case.

My best guess as to why the communities differ in convention is as follows. For many real applications, the choice of norm matters. If you minimize with respect to the ℓ_1 norm or the Euclidean norm, you'll often get different answers. (This sort of stuff is crucial for this class!) However, all norms in finite-dimensional spaces are equivalent, in the sense that they define the same open sets. Since all norms are topologically equivalent, mathematicians often don't bother to distinguish between norms in finite-dimensional spaces.

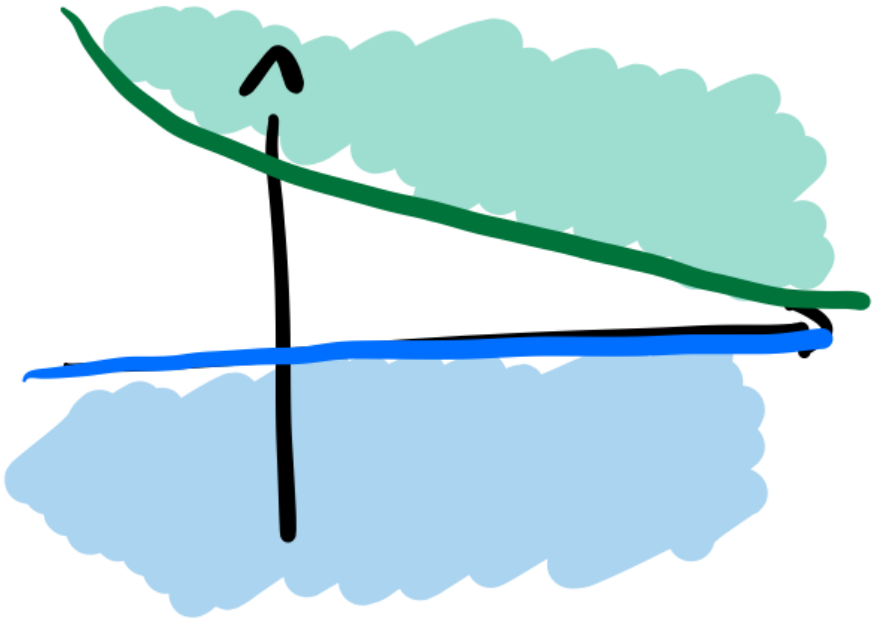

Figure 24: The distance between these two closed, disjoint sets is zero. As they are disjoint, no pair of points from the two sets actually attains this infimum.

The relevant properties are: 1) x and y attain the minimum distance between A and B, and 2) A and B are convex. For the sake of contradiction, suppose this hyperplane does *not* separate A and B. Let's say there's some $x_1 \in A$ such that $c^\intercal x_1 + b > 0$. (The argument when there is a $y_1 \in B$ such that $c^\intercal y_1 + b < 0$ is exactly the same.) What would that look like? Well, let's see Figure 26. In light of the insight from this figure, let's try and show that contradiction.

x_1 itself may not be closer to B than x, but if we start at x and move a little bit in the direction x_1, we will be closer to B.

First, let's note that $(x_1 - x)^\intercal (y - x) > 0$, which visually makes sense. Moving from x to x_1, we have to move in the direction of the hyperplane boundary for x_1 to be on the other side of it. To formally show this, note that $c^\intercal (x_1 - [x + c/2]) > 0$. (This was one of the ways to write the hyperplane we noted above.) Re-arranging, we get:

$$c^\intercal (x_1 - x) > \frac{\|c\|^2}{2} > 0$$

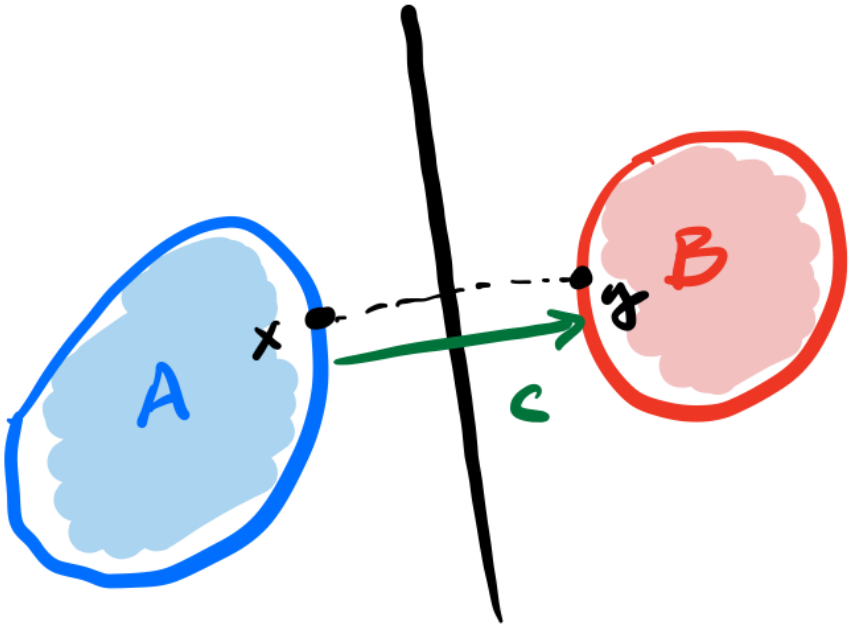

Figure 25: A visualization for the proof of Proposition 7. A and B are convex sets with points $x \in A$ and $y \in B$ that attain the minimum distance between sets. A reasonable candidate for a separating hyperplane would be one that's orthogonal to the line between x and y, placed halfway between the two.

(Recall that $c = y - x$.) So, as we visually already saw, $c^\intercal (x_1 - x) > 0$.

Let's formalize this. Consider the mapping $g(t) = \|y - [x + t(x_1 - x)]\|^2$. We're starting at x and moving in the direction x_1, and checking our (squared) distance to y as we do so. Since A is convex and x and x_1 are in A, all points in between will be in A, i.e. $x + t(x_1 - x) \in A$ for all $t \in [0, 1]$. At $t = 0$, this is simply $g(0) = \|y - x\|^2 = d(A, B)^2$. We can expand this out:

$$g(t) = \|c - t(x_1 - x)\|^2 = \|c\|^2 - 2tc^\intercal (x_1 - x) + t^2 \|x - x_1\|^2$$

Considering the derivative of this mapping:

$$\frac{d}{dt} g : t \mapsto -2c^\intercal (x_1 - x) + 2t\|x - x_1\|^2$$

Evaluating this derivative at zero yields:

$$\left.\frac{d}{dt} g\right|_{t=0} = -2c^\intercal (x_1 - x) < 0$$

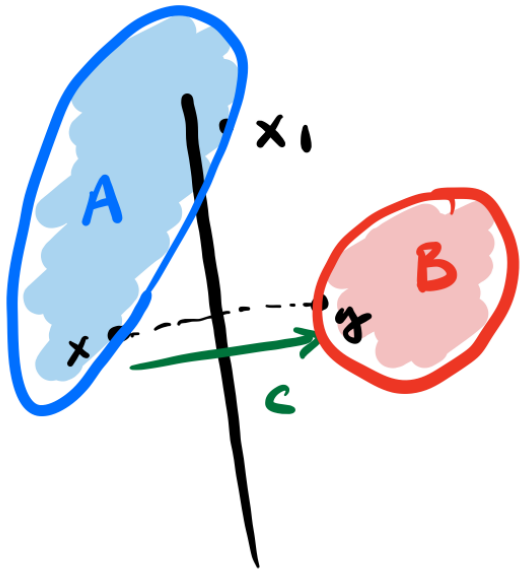

Figure 26: Suppose that $x_1 \in A$ is on the other side of the hyperplane constructed before. Drawing this out, we can see why a contradiction would arise: it seems like x and y would not be a pair of distance-minimizing points anymore.

Since the derivative is negative, there exists some point $t \in (0, 1]$ where $g(t) < g(0)$, meaning the point $x + t(x_1 - x)$ is closer to y than x itself, contradicting the fact that (x, y) attain the minimum distance. Thus, we conclude that such an x_1 cannot exist. QED.

Proposition 7 was proved by explicitly using the x and y that attain the minimum distance. However, this will allow us to prove the more general result.

Proposition 8. *Let A and B be two disjoint convex sets in $\mathbb{R}^n$. Then there exists $c \in \mathbb{R}^n$ and $b \in \mathbb{R}$ such that $c^\intercal x + b \leq 0$ for all $x \in A$ and $c^\intercal y + b \geq 0$ for all $y \in B$.*

I'll leave the proof of this as an exercise. You can first show that $A - B = \{x - y : x \in A, y \in B\}$ is a convex set, and then separate $A - B$ from $\{0\}$. Reformulating the problem this way makes it easier, as now we are considering the distance between a single point and a set; even if this distance is not attained, rather than infimizing sequences (x_n, y_n) with $x \in A, y \in B$, we can take an infimizing sequence (z_n) with $z \in A - B$ and look at how far it is from zero.

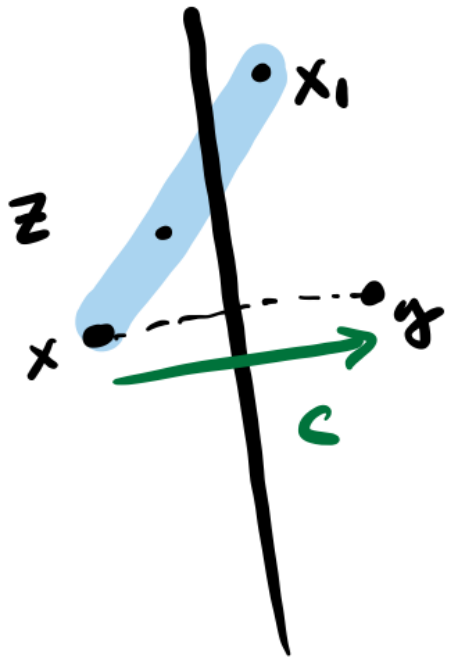

Figure 27: Since x_1 is on the other side of the hyperplane from x, we have that $x_1 - x$ has to have some part that's pointing in line with $c = y - x$. Intuitively, going from x to x_1 has to move positively in the c direction to get to the other side.

Bibliography

Subhonmesh Bose, Steven H. Low, Thanchanok Teeraratkul, and Babak Hassibi. Equivalent relaxations of optimal power flow. *IEEE Transactions on Automatic Control*, 60(3):729–742, March 2015. ISSN 1558-2523. doi: 10.1109/TAC.2014.2357112.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

A. S. Nemirovsky and D.B. Yudin. *Problem Complexity and Method Efficiency in Optimization*. Wiley-Interscience, 1983.

R. Tyrrell Rockafellar. Lagrange multipliers and optimality. *SIAM Review*, 35 (2):183–238, 1993. doi: 10.1137/1035044. URL https://doi.org/10.1137/1035044.

John Schulman, Jonathan Ho, Alex Lee, Ibrahim Awwal, Henry Bradlow, and Pieter Abbeel. Finding locally optimal, collision-free trajectories with sequential convex optimization. In *Proceedings of Robotics: Science and Systems*, Berlin, Germany, June 2013. doi: 10.15607/RSS.2013.IX.031.