MATH 164, OPTIMIZATION, FALL 2016

STEVEN HEILMAN

Contents

1. Introduction

Nothing takes place in the world whose meaning is not that of some maximum or minimum.

Leonhard Euler

Date: November 30, 2016

Optimization is the search for local and global maxima and minima of functions.

Optimization problems are encountered in various disciplines in mathematics, statistics, physics, computer science, and so on. There are some general theories of optimization, but unfortunately, there is no single theory that tells us how to optimize any quantity that we can encounter. So, in this course, we will focus on some theories of optimization, and we will also focus on specific examples. Some examples we will consider include:

- Find the maximum of a continuous function $f\colon [0,1] \to \mathbb{R}$.
- Find the largest eigenvalue of a large square matrix.
- Maximize the volume of a rectangular box of constant surface area.
- If n people want to order a pizza, and each person has some constraints on which toppings they do or do not want, find the set of pizza toppings that maximizes the number of satisfied people.
- Find the shortest path between two points in the plane.

In each of these problems, we consider if we can answer the problem theoretically, and if a computer can answer each problem in a reasonable amount of time. For example, finding the largest eigenvalue of a large square matrix seems impossible for a person to do in general, but perhaps a computer could answer this problem in an amount of time that is a polynomial of the size of the matrix.

2. Review of Calculus and Linear Algebra

2.1. Optimization on the Line.

Definition 2.1 (**Limits**). Let $f\colon \mathbb{R} \to \mathbb{R}$ be a function. Let $x, L \in \mathbb{R}$. We say that f has **limit** L at x if and only if: $\forall\ \varepsilon > 0$, $\exists\ \delta = \delta(\varepsilon) > 0$ such that, for any $y \in \mathbb{R}$ such that $0 < |x - y| < \delta$, we have

$$|f(y) - L| < \varepsilon.$$

If f has limit L at x, we denote this by writing

$$\lim_{y \to x} f(y) = L.$$

Similarly, if $x^{(0)}, x^{(1)}, \ldots$ is a sequence of real numbers, we say the sequence $x^{(0)}, x^{(1)}, \ldots$ has **limit** L as $j \to \infty$ if and only if: $\forall\ \varepsilon > 0$, $\exists\ j = j(\varepsilon) > 0$ such that, for any $k \in \mathbb{R}$ such that $k > j$, we have

$$|x^{(k)} - L| < \varepsilon.$$

If $x^{(0)}, x^{(1)}, \ldots$ has limit L as $j \to \infty$, we denote this by writing

$$\lim_{j \to \infty} x^{(j)} = L.$$

Definition 2.2 (**Continuous Function**). Let $a < b$. Let $f\colon (a,b) \to \mathbb{R}$ be a function. Let $x \in (a,b)$. We say that f is **continuous** at x if and only if

$$\lim_{y \to x} f(y) = f(x).$$

We say that f is **continuous on** (a,b) (or we just say that f is **continuous**) if and only if f is continuous at x for every $x \in (a,b)$. We say that f is **discontinuous** at x if and only if f is not continuous at x.

Definition 2.3 (**Derivative**)**.** Let $a < b$. Let $f\colon (a,b) \to \mathbb{R}$ be a function. Let $x \in (a,b)$. We define the **derivative** $f'(x)$ of f at x to be the following limit, if it exists:

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}.$$

If $f'(x)$ exists, we say that f is **differentiable** at x. We say that f is **differentiable on** (a,b) (or we just say f is **differentiable**) if and only if f is differentiable at x for every $x \in (a,b)$.

Definition 2.4 (**Critical Point**)**.** Let $a < b$. Let $f\colon (a,b) \to \mathbb{R}$. We say that f has a **critical point** at the point $x \in (a,b)$ if $f'(x) = 0$ or $f'(x)$ does not exist.

Definition 2.5 (**Local Extremum**)**.** Let $a < b$. Let $f\colon (a,b) \to \mathbb{R}$. We say that f has a **local maximum** at the point $x \in (a,b)$ if there exists $\varepsilon > 0$ such that $f(x) \geq f(y)$ for all $y \in (x - \varepsilon, x + \varepsilon)$. We say that f has a **local minimum** at the point $x \in \mathbb{R}$ if the function $-f$ has a local maximum at x. If f has either a local maximum or a local minimum at $x \in (a,b)$, we say that x is a **local extremum** of f.

Definition 2.6 (**Global Extrema**)**.** Let $f\colon (a,b) \to \mathbb{R}$. We say that f has a **global maximum** at the point $x \in (a,b)$ if $f(x) \geq f(y)$ for all $y \in (a,b)$. We say that f has a **global minimum** at the point $x \in \mathbb{R}$ if the function $-f$ has a global maximum at x.

Proposition 2.7 (**First Derivative Test**)**.** *Let $c \in (a,b)$ be a critical point for a continuous function $f\colon (a,b) \to \mathbb{R}$. Assume that f is differentiable on (a,c) and on (c,b).*

- *If $f'(y) > 0 \ \forall \ y \in (a,c)$, and if $f'(y) < 0 \ \forall \ y \in (c,b)$, then f has a local maximum at x.*
- *If $f'(y) < 0 \ \forall \ y \in (a,c)$, and if $f'(y) > 0 \ \forall \ y \in (c,b)$, then f has a local minimum at x.*
- *If $f'(y) > 0 \ \forall \ y \in (a,c) \cup (c,b)$, or if $f'(y) < 0 \ \forall \ y \in (a,c) \cup (c,b)$, then f does not have a local maximum or a local minimum at x.*

Exercise 2.8. Find a continuous function $f\colon \mathbb{R} \to \mathbb{R}$ such that f has a global maximum at $x = 0$, but f is not differentiable at 0.

Proposition 2.9 (**Second Derivative Test**)**.** *Let $f\colon (a,b) \to \mathbb{R}$. Let $c \in (a,b)$. Assume that $f'(c)$ and $f''(c)$ exist. Assume also that $f'(c) = 0$, and that f'' is continuous on (a,b).*

1. *If $f''(c) > 0$, then f has a local minimum at c.*
2. *If $f''(c) < 0$, then f has a local maximum at c.*
3. *If $f''(c) = 0$, then f may or may not have a local extremum at c.*

Below is a procedure for finding the extreme values of a continuous function $f\colon [a,b] \to \mathbb{R}$.

Algorithm 2.10.

- Find the critical points of f in (a,b).
- Compute the value of f at the critical points of f, and at the endpoints a and b.
- Choose the largest and smallest values of f from these points.

Exercise 2.11. Let $f\colon [-1,2] \to \mathbb{R}$ be defined by $f(x) = x^3 - 3x + 2$. Find all local and global extrema of f.

Algorithm 2.10 actually finds the extreme values of f, due to the following theorem.

Theorem 2.12 (**Extreme Value Theorem**). *Let $a < b$. Let $f\colon [a,b] \to \mathbb{R}$ be a continuous function. Then there exist $x, y \in [a,b]$ such that $f(x) \leq f(z) \leq f(y)$ for all $z \in [a,b]$.*

Exercise 2.13. Find a continuous function $f\colon (0,1) \to \mathbb{R}$ such that there does not exist $x \in (0,1)$ such that $f(x) \leq f(z)$ for all $z \in (0,1)$.

Remark 2.14. Since maximizing a function f is equivalent to minimizing the function $-f$, when we discuss optimization below, we will typically focus on maximizing a function f.

2.2. **Linear Algebra.**

Let $x = (x_1, \ldots, x_n)$ be a vector in $\mathbb{R}^n$. Unless otherwise stated, a vector $x \in \mathbb{R}^n$ will always denote a column vector. Also, we let I denote the $n \times n$ identity matrix.

Let A be a real $n \times n$ matrix. For any $i, j \in \{1, \ldots, n\}$, we let A_{ij} denote the entry in the i^{th} row and j^{th} column of A. We say that A is a **diagonal** matrix if $A_{ij} = 0$ for all $i, j \in \{1, \ldots, n\}$ with $i \neq j$.

Remark 2.15. We use the symbol 0 to denote the scalar zero. We also use the symbol 0 to denote the zero vector.

Definition 2.16. Let A be a real $n \times n$ matrix. We say that A is **symmetric** if $A = A^T$. We say A is **orthogonal** if $AA^T = A^T A = I$. That is, A is orthogonal when $A^T = A^{-1}$.

Definition 2.17. Let A be a real $n \times n$ matrix. Let $\lambda \in \mathbb{C}$. We say that λ is an **eigenvalue** of A if $\det(A - \lambda I) = 0$. We say that $x \in \mathbb{R}^n$ is an **eigenvector** of A with eigenvalue λ if $Ax = \lambda x$ and if $x \neq 0$.

Exercise 2.18. Find all eigenvalues and eigenvectors of the matrix $A = \begin{pmatrix} 1 & 2 \\ 0 & 2 \end{pmatrix}$.

Find all eigenvalues and eigenvectors of the matrix $A = \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix}$.

Exercise 2.19. Prove that a real $n \times n$ matrix has at least one eigenvalue.

Definition 2.20. Let A be an $n \times n$ real symmetric matrix. We say that A is **positive definite** if

$$x^T A x > 0, \qquad \forall\, x \in \mathbb{R}^n,\ x \neq 0.$$

We say that A is **positive semidefinite**, which we denote by $A \geq 0$, if

$$x^T A x \geq 0, \qquad \forall\, x \in \mathbb{R}^n.$$

Exercise 2.21. Let A be an $n \times n$ real symmetric matrix. Show that the following three conditions are equivalent:

- A is positive semidefinite
- All eigenvalues of A are nonnegative.
- There exists a real $n \times n$ matrix B such that $A = BB^T$.

(Hint: you should probably use the Spectral Theorem for Symmetric Matrices.)

Theorem 2.22 (**Spectral Theorem for Symmetric Matrices**). *Let A be an $n \times n$ real symmetric matrix. Then there exists an orthogonal $n \times n$ matrix Q whose columns are each eigenvectors of A, and there exists a real diagonal matrix D whose diagonal entries are the eigenvalues of A such that $Q^{-1}AQ = D$. That is, $A = QDQ^{-1}$.*

Equivalently, if $\lambda_1, \ldots, \lambda_n \in \mathbb{C}$ are the eigenvalues of A (where some eigenvalues are allowed to be the same), then $\lambda_1, \ldots, \lambda_n \in \mathbb{R}$, and there exist vectors $v_1, \ldots, v_n \in \mathbb{R}^n$ which are an orthonormal basis of $\mathbb{R}^n$ such that $A = \sum_{i=1}^n \lambda_i v_i v_i^T$.

Exercise 2.23. Show that the two different versions of the Spectral Theorem for Symmetric Matrices are equivalent.

Exercise 2.24. Let A be an $n \times n$ real symmetric matrix. Let $\lambda_1 \geq \cdots \geq \lambda_n \in \mathbb{R}$ be the eigenvalues of A, ordered according to their size. Let $x \in \mathbb{R}^n$. Show that

$$\lambda_1 \|x\|^2 \geq x^T A x \geq \lambda_n \|x\|^2 .$$

Exercise 2.25. Prove the **Cauchy-Schwarz inequality**: For any $x, y \in \mathbb{R}^n$, we have

$$|\langle x, y\rangle| \leq \|x\| \|y\| .$$

(Hint: subtract the projection of y onto x. That is, if $x \neq 0$, let $v := \frac{\langle x,y\rangle}{\langle x,x\rangle} x$, and expand out the inequality $\|y - v\|^2 \geq 0$.)

Exercise 2.26. Prove the **triangle inequality**: For any $x, y, z \in \mathbb{R}^n$,

$$\|x - y\| \leq \|x - z\| + \|z - y\| .$$

(Hint: it may be conceptually easier to show $\|x + y\| \leq \|x\| + \|y\|$. To show this inequality, square both sides, and use the Cauchy-Schwarz inequality.) Then, deduce the **reverse triangle inequality**:

$$\|x - y\| \geq |\, \|x\| - \|y\| \,| .$$

Exercise 2.27 (The Power Method). This exercise gives an algorithm for finding the eigenvectors and eigenvalues of a symmetric matrix. The Power Method described below is not the best algorithm for this task, but it is perhaps the easiest to describe and analyze.

Let A be an $n \times n$ real symmetric matrix. Let $\lambda_1 \geq \cdots \geq \lambda_n$ be the (unknown) eigenvalues of A, and let $v_1, \ldots, v_n \in \mathbb{R}^n$ be the corresponding (unknown) eigenvectors of A such that $\|v_i\| = 1$ and such that $Av_i = \lambda_i v_i$ for all $1 \leq i \leq n$.

Given A, our first goal is to find v_1 and λ_1. For simplicity, assume that $1/2 < \lambda_1 < 1$, and $0 \leq \lambda_n \leq \cdots \leq \lambda_2 < 1/4$. Suppose we have found a vector $v \in \mathbb{R}^n$ such that $\|v\| = 1$ and $|\langle v, v_1\rangle| > 1/n$. (A randomly chosen v will satisfy $|\langle v, v_1\rangle| > 1/(10\sqrt{n})$, [which is a nice optional exercise for those who have taken 170A], so this assumption is valid in practice.) Let k be a positive integer. Show that

$$A^k v$$

approximates v_1 well as k becomes large. More specifically, show that for all $k \geq 1$,

$$\left\|A^k v - \langle v, v_1\rangle \lambda_1^k v_1\right\|^2 \leq \frac{n-1}{16^k}.$$

Since $|\langle v, v_1\rangle| \lambda_1^k > 2^{-k}/n$, this inequality implies that $A^k v$ is approximately an eigenvector of A with eigenvalue λ_1. That is, by the triangle inequality,

$$\left\|A(A^k v) - \lambda_1(A^k v)\right\| \leq \left\|A^{k+1} v - \langle v, v_1\rangle \lambda_1^{k+1} v_1\right\| + \lambda_1 \left\|\langle v, v_1\rangle \lambda_1^k v_1 - A^k v\right\| \leq 2\frac{\sqrt{n-1}}{4^k}.$$

Moreover, by the reverse triangle inequality,

$$\left\|A^k v\right\| = \left\|A^k v - \langle v, v_1\rangle \lambda_1^k v_1 + \langle v, v_1\rangle \lambda_1^k v_1\right\| \geq \frac{1}{n} 2^{-k} - \frac{\sqrt{n-1}}{4^k}.$$

In conclusion, if we take k to be large (say $k > 10\log n$), and if we define $z := A^k v$, then z is approximately an eigenvector of A, that is

$$\left\| A\frac{A^k v}{\|A^k v\|} - \lambda_1 \frac{A^k v}{\|A^k v\|}\right\| \leq 4n^{3/2}2^{-k} \leq 4n^{-4}.$$

And to approximately find the first eigenvalue λ_1, we simply compute

$$\frac{z^T A z}{z^T z}.$$

That is, we have approximately found the first eigenvector and eigenvalue of A.

Remarks. To find the second eigenvector and eigenvalue, we can repeat the above procedure, where we start by choosing v such that $\langle v, v_1\rangle = 0$, $\|v\| = 1$ and $|\langle v, v_2\rangle| > 1/(10\sqrt{n})$. To find the third eigenvector and eigenvalue, we can repeat the above procedure, where we start by choosing v such that $\langle v, v_1\rangle = \langle v, v_2\rangle = 0$, $\|v\| = 1$ and $|\langle v, v_3\rangle| > 1/(10\sqrt{n})$. And so on.

Google's PageRank algorithm uses the power method to rank websites very rapidly. In particular, they let n be the number of websites on the internet (so that n is roughly 10^9). They then define an $n \times n$ matrix C where $C_{ij} = 1$ if there is a hyperlink between websites i and j, and $C_{ij} = 0$ otherwise. Then, they let B be an $n \times n$ matrix such that B_{ij} is 1 divided by the number of 1's in the i^{th} row of C, if $C_{ij} = 1$, and $B_{ij} = 0$ otherwise. Finally, they define

$$A = (.85)B + (.15)D/n$$

where D is an $n \times n$ matrix all of whose entries are 1.

The power method finds the eigenvector v_1 of A, and the size of the i^{th} entry of v_1 is proportional to the "rank" of website i.

Exercise 2.28 (This exercise is optional; any exercise in this course involving programming is optional)**.** Write a program in Matlab that computes the first, second, and third eigenvectors and eigenvalues of a symmetric matrix A of arbitrary size. Compare your results with the Matlab programs `eigs` and `eig`.

Then, under the assumptions of the previous exercise ($1/2 < \lambda_1 < 1$, and $0 \leq \lambda_n \leq \cdots \leq \lambda_2 < 1/4$), provide an upper bound on the number of arithmetic operations that are required to compute the first three decimal places of the first eigenvalue λ_1. Your upper bound could involve either the size n of the matrix A, or the number m of nonzero entries of A.

Note that the power method iteratively applies the matrix to a vector, instead of multiplying matrices together. The latter operation can require many more arithmetic operations than the former.

2.3. **Convex Geometry, Convex Functions.**

Definition 2.29. A set $K \subseteq \mathbb{R}^n$ is **convex** if, for any two points $x, y \in K$, the line segment between them

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

also lies in K.

Example 2.30. The unit disc $\{(x_1, x_2) \in \mathbb{R}^2 : x_1^2 + x_2^2 \leq 1\}$ is convex. The unit circle $C = \{(x_1, x_2) \in \mathbb{R}^2 : x_1^2 + x_2^2 = 1\}$ is not convex, since $(1,0) \in C$, $(-1,0) \in C$, $(0,0) \notin C$, while $(0,0) = (1/2)(1,0) + (1/2)(-1,0)$. So, if C were convex, then $(0,0)$ would have to be in C.

Exercise 2.31. Show that the intersection of two convex sets is convex.

Definition 2.32. A function $f\colon \mathbb{R}^n \to \mathbb{R}$ is **convex** if, for any two points $x, y \in \mathbb{R}^n$, and for any $t \in (0,1)$,

$$f(tx + (1-t)y) \leq tf(x) + (1-t)f(y).$$

A function $f\colon \mathbb{R}^n \to \mathbb{R}$ is **strictly convex** if, for any two points $x, y \in \mathbb{R}^n$, and for any $t \in (0,1)$,

$$f(tx + (1-t)y) < tf(x) + (1-t)f(y).$$

Exercise 2.33. Let $f\colon \mathbb{R} \to \mathbb{R}$ so that $f(x) = x^2$. Show that f is convex.

Exercise 2.34. Let $f\colon \mathbb{R} \to \mathbb{R}$ be a function with three continuous derivatives. Show that f is convex if and only if $f''(x) \geq 0$ for all $x \in \mathbb{R}$. (Hint: for the reverse implication, you may need to use Taylor's Theorem with integral remainder.)

Theorem 2.35 (Taylor's Theorem with Integral Remainder, Dimension 1). *Let $k > 0$. Let $f\colon \mathbb{R} \to \mathbb{R}$ be a function with $k+1$ continuous derivatives. Then, for any $x, y \in \mathbb{R}$,*

$$f(x) = f(y) + \sum_{i=1}^{k-1} \frac{(x-y)^i}{i!} f^{(i)}(y) + \frac{1}{(k-1)!} \int_y^x f^{(k)}(t)(x-t)^{k-1} dt$$

Remark 2.36. The case $k = 1$ of Taylor's Theorem is the Fundamental Theorem of Calculus.

Exercise 2.37. Prove Taylor's Theorem with Integral Remainder when $k = 2$:

Let $f\colon \mathbb{R} \to \mathbb{R}$ be a function with three continuous derivatives. Then, for any $x, y \in \mathbb{R}$,

$$f(x) = f(y) + (x-y)f'(y) + \int_y^x f''(t)(x-t)dt.$$

(Hint: integrate by parts.)

Exercise 2.38. Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be a strictly convex function. Show that f has at most one global minimum.

Then, find a convex set $K \subseteq \mathbb{R}$ and a strictly convex function $f\colon K \to \mathbb{R}$ such that f does not have a global minimum.

Exercise 2.39. Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be a convex function. Let $x \in \mathbb{R}^n$ be a local minimum of f. Show that x is in fact a global minimum of f.

Now suppose additionally that $f \in C^1$, and $x \in \mathbb{R}^n$ satisfies $\nabla f(x) = 0$. Show that x is a global minimum of f.

Exercise 2.40. In statistics and other applications, we can be presented with data points $(x_1, y_1), \ldots, (x_n, y_n)$. We would like to find the line $y = mx + b$ which lies "closest" to all of these data points. Such a line is known as a **linear regression**. There are many ways to define the "closest" such line. The standard method is to use **least squares minimization**. A line which lies close to all of the data points should make the quantities $(y_i - mx_i - b)$ all very small. We would like to find numbers m, b such that the following quantity is minimized:

$$f(m,b) = \sum_{i=1}^{n} (y_i - mx_i - b)^2.$$

Show that the global minimum value of f is achieved when

$$m = \frac{\left(\sum_{i=1}^{n} x_i\right)\left(\sum_{j=1}^{n} y_j\right) - n\left(\sum_{k=1}^{n} x_k y_k\right)}{\left(\sum_{i=1}^{n} x_i\right)^2 - n\left(\sum_{j=1}^{n} x_j^2\right)}.$$

$$b = \frac{1}{n}\left(\sum_{i=1}^{n} y_i - m\sum_{j=1}^{n} x_j\right).$$

(In probabilistic terminology, $-m$ is a covariance divided by a variance.)

2.4. **Lagrange Multipliers.**

Definition 2.41 (**Standard Inner Product**)**.** Let $x = (x_1, \ldots, x_n), y = (y_1, \ldots, y_n) \in \mathbb{R}^n$. We define the **(standard) inner product** of x and y, denoted $\langle x, y\rangle$, to be

$$\langle x, y\rangle := x^T y = \sum_{i=1}^{n} x_i y_i.$$

We define the 2**-norm** of x, denoted $\|x\|$, to be

$$\|x\| := \sqrt{\langle x, x\rangle} = (\sum_{i=1}^{n} x_i^2)^{1/2}.$$

Definition 2.42 (**Limits**)**.** Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be a function. Let $x \in \mathbb{R}^n$, $L \in \mathbb{R}$. We say that f has **limit** L at x if and only if: $\forall\ \varepsilon > 0$, $\exists\ \delta = \delta(\varepsilon) > 0$ such that, for any $y \in \mathbb{R}^n$ such that $0 < \|x - y\| < \delta$, we have

$$|f(y) - L| < \varepsilon.$$

If f has limit L at x, we denote this by writing

$$\lim_{y \to x} f(y) = L.$$

Definition 2.43 (**Continuous Function**)**.** Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be a function. Let $x \in \mathbb{R}^n$. We say that f is **continuous** at x if and only if

$$\lim_{y \to x} f(y) = f(x).$$

We say that f is **continuous on** $\mathbb{R}^n$ (or we just say that f is **continuous**) if and only if f is continuous at x for every $x \in \mathbb{R}^n$. We say that f is **discontinuous** at x if and only if f is not continuous at x.

Definition 2.44 (**Derivatives**)**.** Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be a function. Let $x \in \mathbb{R}^n$. Let $i \in \{1, \ldots, n\}$. We define the **partial derivative** $\frac{\partial f}{\partial x_i}(x)$ of f at x in the x_i direction to be the following limit, if it exists:

$$\frac{\partial f}{\partial x_i}(x) = \frac{\partial}{\partial x_i} f(x) := \lim_{h \to 0} \frac{f(x_1, \ldots, x_{i-1}, x_i + h, x_{i+1}, \ldots, x_n) - f(x)}{h}.$$

If $i, j \in \{1, \ldots, n\}$, we then define

$$\frac{\partial^2 f}{\partial x_i \partial x_j}(x) := \frac{\partial}{\partial x_i}\left(\frac{\partial f}{\partial x_j}(x)\right).$$

Similarly, If $k > 0$, and if $i_1, \ldots, i_k \in \{1, \ldots, n\}$, then we can define the iterated partial derivatives of f by induction:

$$\frac{\partial^k f}{\partial x_{i_1} \cdots \partial x_{i_k}}(x) := \frac{\partial}{\partial x_{i_1}} \left(\frac{\partial^{k-1} f}{\partial x_{i_2} \cdots \partial x_{i_k}}(x) \right),$$

and we say that $\frac{\partial^k f}{\partial x_{i_1} \cdots \partial x_{i_k}}(x)$ is an **iterated partial derivative of order** k of f.

Remark 2.45. These definitions easily extend to vector-valued functions. For example, we say that $s\colon \mathbb{R} \to \mathbb{R}^n$ is differentiable if each coordinate of s is a differentiable function.

Definition 2.46 (**C^k Function**)**.** Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be a function and let k be a positive integer. If all partial derivatives of order k of f exist and are continuous, we say that f is a C^k **function**, which we denote by writing $f \in C^k$.

Definition 2.47 (**Gradient, Directional Derivative**)**.** Let $f \in C^1$ and let $x \in \mathbb{R}^n$. The **gradient** of f at x is the following column vector

$$\nabla f(x) := \left(\frac{\partial f}{\partial x_1}(x), \ldots, \frac{\partial f}{\partial x_n}(x) \right)^T.$$

Let $v \in \mathbb{R}^n$. The **directional derivative** of f at x in the direction v, denoted $D_v f(x)$, is

$$D_v f(x) := \langle \nabla f(x), v \rangle.$$

Remark 2.48. We also use the notation $Df(x) = \left(\frac{\partial f}{\partial x_1}(x), \ldots, \frac{\partial f}{\partial x_n}(x) \right)$ for the row vector of partial derivatives at x.

Definition 2.49 (**Critical Point**)**.** Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be a function. We say that f has a **critical point** at the point $x \in \mathbb{R}^n$ if $\nabla f(x) = 0$, or if any partial derivative of f does not exist at x.

Definition 2.50 (**Local Extremum**)**.** Let $D \subseteq \mathbb{R}^n$. Let $f\colon D \to \mathbb{R}$ be a function. We say that f has a **local maximum** at the point $x \in D$ if there exists $\varepsilon > 0$ such that $f(x) \geq f(y)$ for all $y \in D$ such that $\|y - x\| < \varepsilon$. We say that f has a **local minimum** at the point $x \in D$ if the function $-f$ has a local maximum at x. If f has either a local maximum or a local minimum at $x \in D$, we say that x is a **local extremum** of f.

Definition 2.51 (**Global Extrema**)**.** Let $D \subseteq \mathbb{R}^n$. Let $f\colon D \to \mathbb{R}$. We say that f has a **global maximum** at the point $x \in D$ if $f(x) \geq f(y)$ for all $y \in D$. We say that f has a **global minimum** at the point $x \in D$ if the function $-f$ has a global maximum at x.

Exercise 2.52. Find a function $f\colon \mathbb{R} \to \mathbb{R}$ such that no local or global maximum of f exists, and no local or global minimum of f exists.

Definition 2.53 (**Level Surface**)**.** Let $f\colon \mathbb{R}^n \to \mathbb{R}$. Let $c \in \mathbb{R}$. A **level surface** of f is any set of the form

$$\{x \in \mathbb{R}^n \colon f(x) = c\}.$$

Exercise 2.54. Let $f\colon \mathbb{R}^n \to \mathbb{R}$. A version of Taylor's Theorem for functions on $\mathbb{R}^n$ follows from Taylor's Theorem for functions on $\mathbb{R}$ in the following way. (For simplicity, we look at the Taylor expansion of f at $x = 0$.) Let $y \in \mathbb{R}^n$, let $t \in \mathbb{R}$, and define $g\colon \mathbb{R} \to \mathbb{R}$ by $g(t) = f(ty)$. Then Taylor's Theorem for g holds. Using the Chain rule, what are the first two or three terms in the Taylor expansion of g, in terms of derivatives of f at $x = 0$?

Proposition 2.55 (**Geometric Interpretation of the Gradient**). *Let $n \geq 2$. Let $f \in C^1$. Let $x \in \mathbb{R}^n$. Assume $\nabla f(x) \neq 0$.*

(i) *$\nabla f(x)$ points in the direction of greatest increase of f. That is, if $w := \frac{\nabla f(x)}{\|\nabla f(x)\|}$, then*
$$D_w f(x) = \max_{v \in \mathbb{R}^n \colon \|v\|=1} D_v f(x).$$

(ii) *$\nabla f(x)$ is orthogonal to the level surface of f that passes through x.*

Proof. Let $v \in \mathbb{R}^n$ with $\|v\| = 1$. The Cauchy-Schwarz inequality, Exercise 2.25, implies
$$D_v f(x) = \langle \nabla f(x), v \rangle \leq \|\nabla f(x)\| = D_w f(x).$$
Item (i) is proven.

We now prove item (ii). Let $c \in \mathbb{R}$ and let $S := \{y \in \mathbb{R}^n \colon f(y) = c\}$ be a level surface of f such that $x \in S$. Let $s \colon \mathbb{R} \to \mathbb{R}^n$ be a path such that $s(0) = x$ and such that $f(s(t))$ is constant in $t \in \mathbb{R}$. From the Chain Rule,
$$0 = \frac{d}{dt}|_{t=0} f(s(t)) = \langle \nabla f(s(0)), s'(0) \rangle.$$
Let T be the span of all vectors $s'(0)$ with $s(0) = x$. From Lemma 2.56 below, T has dimension $n-1$. And each vector $s'(0)$ is tangent to the level surface S. Since $\nabla f(s(0))$, is perpendicular to T, we conclude that $\nabla f(s(0))$ is perpendicular to the level surface. □

Lemma 2.56. *Let $f \in C^1$. Let $x \in \mathbb{R}^n$ with $\nabla f(x) \neq 0$. Let $c \in \mathbb{R}$, and let $S = \{y \in \mathbb{R}^n \colon f(y) = c\}$. Let $T := \mathrm{span}\{s'(0) \colon s(0) = x,\ s \colon \mathbb{R} \to S$ is differentiable$\}$. Then T has dimension $n-1$.*

Proof. Without loss of generality, $\partial f(x)/\partial x_n \neq 0$ and $x = 0$. We freely use the Implicit Function Theorem. This theorem says: $\exists\, \varepsilon > 0$ such that $S \cap \{y \in \mathbb{R}^n \colon \|y - x\| < \varepsilon\}$ is the graph of a function g. That is, $\exists\, \delta > 0$, $\exists\, U \subseteq \mathbb{R}^{n-1}$ such that $\{u \in \mathbb{R}^{n-1} \|u - (x_1, \ldots, x_{n-1})\| < \delta\} \subseteq U$, and $\exists$ a C^1 function $g \colon \mathbb{R}^{n-1} \to \mathbb{R}$ such that
$$S \cap \{y \in \mathbb{R}^n \colon \|y - x\| < \varepsilon\} = \{(z, g(z)) \colon z \in U\}. \qquad (*)$$
We show that $(*)$ implies T is an $(n-1)$-dimensional subspace. Let $1 \leq i \leq n-1$ and let $e_i \in \mathbb{R}^{n-1}$ be the vector with a 1 in the i^{th} entry and zeros in all other entries. Then for any $1 \leq i \leq n-1$, define $s_i \colon \mathbb{R} \to \mathbb{R}^n$ by $s_i(t) := (te_i, g(te_i))$, where we restrict the domain of s so that t satisfies $te_i \in U$. Then $s(0) = 0$, and $s_1'(0), \ldots, s_{n-1}'(0)$ are linearly independent, so T has dimension at least $n-1$. Also, T has dimension at most $n-1$ by the right side of $(*)$. Therefore, T has dimension $n-1$, as desired. □

Proposition 2.57 (**Lagrange Multipliers**). *Let $f, g \in C^1$. Let $c \in \mathbb{R}$. Let $x \in \mathbb{R}^n$ so that $g(x) = c$ and such that x is a local maximum of f on the set $\{x \in \mathbb{R}^n \colon g(x) = c\}$. Assume $\nabla g(x) \neq 0$. Then $\exists\, \lambda \in \mathbb{R}$ such that*
$$\nabla f(x) = \lambda \nabla g(x).$$

Proof. Let $s \colon \mathbb{R} \to \mathbb{R}^n$ be a differentiable path such that $g(s(t)) = c$ for all $t \in [-1, 1]$, and $s(0) = x$. By assumption, $f(s(t))$ is a function of the real variable t which has a local maximum at $t = 0$. So, the Chain Rule implies
$$0 = \frac{d}{dt}|_{t=0} f(s(t)) = \langle \nabla f(s(0)), s'(0) \rangle.$$

Let T be the span of all vectors $s'(0)$ resulting from differentiable paths $s\colon \mathbb{R} \to \mathbb{R}^n$ with $s(0) = x$. Lemma 2.56 implies that T is an $(n-1)$-dimensional subspace. In summary, $\nabla f(x)$ is perpendicular to T, and also $\nabla g(x)$ is perpendicular to T, by Proposition 2.55. Since T is $(n-1)$-dimensional, we conclude that $\nabla f(x) = \lambda \nabla g(x)$ for some $\lambda \in \mathbb{R}$. □

Exercise 2.58. Maximize $f(x, y) = x^2 + y^2$ subject to the constraint $x^2 + 2y^2 = 1$.

Remark 2.59. As we know from calculus class, the level set of g might have a boundary, in which case we need to check the boundary of the level set in order to optimize f on the level set.

Exercise 2.60. Suppose that we have a probability distribution on the set $\{1, \ldots, n\}$, i.e. a sequence $p = (p_1, \ldots, p_n)$ of probabilities in the set $\overline{\mathcal{P}_n}$, where

$$\overline{\mathcal{P}_n} := \left\{ p \in [0,1]^n \colon \sum_{i=1}^{n} p_i = 1 \right\}, \qquad \mathcal{P}_n := \left\{ p \in (0,1)^n \colon \sum_{i=1}^{n} p_i = 1 \right\}.$$

A fundamental quantity for a probability distribution p is its *entropy*

$$S(p) := -\sum_{i=1}^{n} p_i \log p_i \,.$$

(We extend the function $x \log x$ to 0 by continuity, so that $0 \log 0 := 0$.) The entropy of p measures the disorder or lack of information in p.

(i) Using Lagrange multipliers, find the local maximum q of S on the set $\mathcal{P}_n$. Compute the value of S at q.
(ii) Prove that S reaches its maximum on $\overline{\mathcal{P}_n}$ at q.

Exercise 2.61. Let A be a real symmetric positive definite $n \times n$ matrix. Let $b \in \mathbb{R}^n$. Define $f\colon \mathbb{R}^n \to \mathbb{R}$ so that, for any $y \in \mathbb{R}^n$,

$$f(y) := \frac{1}{2} y^T A y - b^T y$$

Show that f is strictly convex. Conclude that f has exactly one global minimum. (Recall that strict convexity alone does not guarantee that a global minimum exists.)

More generally, let $1 \leq k \leq n-1$, let $H \subseteq \mathbb{R}^n$ be a k-dimensional subspace of $\mathbb{R}^n$, let $x^{(0)} \in \mathbb{R}^n$ and let

$$K := \{x^{(0)} + h \colon h \in H\}.$$

Let $f_K\colon K \to \mathbb{R}$ by $f_K(y) := \frac{1}{2} y^T A y - b^T y$, $\forall\, y \in K$. Then f_K also has exactly one global minimum $x_K \in K$. Moreover, $\nabla f(x_K) = A x_K - b$ is orthogonal to H. Conversely, if $x_K \in K$ satisfies $A x_K - b$ is orthogonal to H, then x_K is the unique global minimum of f on K.

2.5. Second Derivative Test.

Definition 2.62. Let $f\colon \mathbb{R}^n \to \mathbb{R}$. We define the **Hessian matrix** of f at $x \in \mathbb{R}^n$, denoted $D^2 f(x)$, to be the following $n \times n$ matrix (if it exists):

$$D^2 f(x) := \begin{pmatrix} \frac{\partial^2 f}{\partial x_1^2}(x) & \cdots & \frac{\partial^2 f}{\partial x_n \partial x_1}(x) \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_1 \partial x_n}(x) & \cdots & \frac{\partial^2 f}{\partial x_n^2}(x) \end{pmatrix}.$$

Remark 2.63. Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be a function such that all of its second order partial derivatives exists and are continuous. Then $\frac{\partial^2 f}{\partial x_i \partial x_j} = \frac{\partial^2 f}{\partial x_j \partial x_i}$ for all $i, j \in \{1, \ldots, n\}$ (this statement is known as Clairaut's Theorem, and it is proven in 131B). Consequently, the Hessian of f is a symmetric matrix, so the Spectral Theorem, Theorem 2.22, implies that all of its eigenvalues are real.

Proposition 2.64 (**Second Derivative Test**). *Let $f\colon \mathbb{R}^n \to \mathbb{R}$ with $f \in C^3$. Let $x \in \mathbb{R}^n$ be a critical point of f.*

- *If all eigenvalues of $D^2 f(x)$ are positive (i.e. if $D^2 f(x)$ is positive definite), then x is a local minimum of f.*
- *If all eigenvalues of $D^2 f(x)$ are negative (i.e. if $D^2 f(x)$ is negative definite), then x is a local maximum of f.*
- *If $D^2 f(x)$ has one positive and one negative eigenvalue, then x is called a **saddle point** of f.*

Proof. We prove the first assertion. We write f in its second-order Taylor expansion. Then there exists $\varepsilon > 0, C > 0$ such that, for all $y \in \mathbb{R}^n$ with $\|y - x\| < \varepsilon$,

$$\left| f(y) - \left(f(x) + \langle y - x, \nabla f(x) \rangle + \frac{1}{2}(y - x)^T (D^2 f(x))(y - x) \right) \right| \leq C \|y - x\|^3 .$$

In particular,

$$f(y) \geq f(x) + \langle y - x, \nabla f(x) \rangle + \frac{1}{2}(y - x)^T (D^2 f(x))(y - x) - C \|y - x\|^3 .$$

Let $c > 0$ be the smallest eigenvalue of $D^2 f(x)$. Using $\nabla f(x) = 0$ and Exercise 2.24,

$$f(y) \geq f(x) + c \|y - x\|^2 - C \|y - x\|^3 = f(x) + \|y - x\|^2 (c - C \|y - x\|).$$

So, if $y \in \mathbb{R}^n$ satisfies $0 < \|y - x\| < \min(\varepsilon, c/C)$, we get

$$f(y) > f(x).$$

That is, x is a local minimum of f □

If $\nabla f(x) = 0$ and if $D^2 f(x)$ has all nonnegative eigenvalues, then the point x may or may not be a local extremum of f.

Exercise 2.65. Give an example of a function $f\colon \mathbb{R}^2 \to \mathbb{R}$ such that $\nabla f(0) = 0$, all eigenvalues of $D^2 f(0)$ are nonnegative, but f does not have a local minimum at 0.

Exercise 2.66. Let $f\colon \mathbb{R}^2 \to \mathbb{R}$ so that $f(x, y) = x^2 + y^2(1 + x)^3$. Show that f has one critical point which is a local minimum, but f has no global maximum, and f has no global minimum.

That is, having only one critical point which is a local minimum does not imply that this point is a global minimum.

Exercise 2.67. Let $f \in C^2$. Assume that f has a local maximum at $x \in \mathbb{R}^n$. Conclude that $\nabla f(x) = 0$.

Exercise 2.68. Prove the following partial converse to Proposition 2.64.

Let $f \in C^3$. Assume that f has a local minimum at $x \in \mathbb{R}^n$. Then $\nabla f(x) = 0$ and all eigenvalues of $D^2 f(x)$ are nonnegative.

3. Optimization of Real Functions

3.1. **Gradient Ascent/Descent.** Suppose we have a function $f\colon \mathbb{R}^n \to \mathbb{R}$. From the Geometric Interpretation of the Gradient, Proposition 2.55, we might expect that we can maximize f by moving in the direction of increase of f.

Algorithm 3.1 (Gradient Ascent Algorithm).

- Start at any point $x^{(0)} \in \mathbb{R}^n$.
- Move in the direction $\nabla f(x^{(0)})$. That is, let $\varepsilon > 0$ be small, and define $x^{(1)} = x^{(0)} + \varepsilon \nabla f(x^{(0)})$. More generally, if $n \geq 1$ and if we are given $x^{(n)}$, define
$$x^{(n+1)} := x^{(n)} + \varepsilon \nabla f(x^{(n)}).$$
(To instead minimize the function f, define $x^{(n+1)} := x^{(n)} - \varepsilon \nabla f(x^{(n)})$.)
- Repeat the previous step several times.

Note that even if the sequence of points $x^{(0)}, x^{(1)}, \ldots$ converges to some $x \in \mathbb{R}$, then x may only be a local maximum of f. That is, x may not be a global maximum.

Example 3.2. Let $f\colon \mathbb{R} \to \mathbb{R}$ so that $f(x) = (x^2-1)^2$. Then f has no global maximum, and f has a local maximum at $x = 0$. However, if we use the initial guess $x^{(0)} = 1/2$ with $\varepsilon = 1/100$, then the sequence of points $x^{(0)}, x^{(1)}, \ldots$ from the Gradient Descent algorithm will converge to the local maximum at 0. For example, we have
$$x^{(n)} = x^{(n-1)} + (1/100)4x^{(n-1)}((x^{(n-1)})^2 - 1), \qquad \forall\, n \geq 1.$$
So, using a computer,
$$x^{(1)} = .485, \qquad x^{(2)} = .4702, \quad \ldots \quad x^{(100)} = .0098, \quad \ldots \quad x^{(200)} = .000165.$$

Remark 3.3. Already in the above example, we have encountered the main problem with using computers to perform arithmetic. As a general rule, multiplication, addition and division of positive numbers can be done with negligible errors on a computer. However, subtracting two numbers on a computer will introduce numerical error, if those two numbers are close to each other. Consider for example the quantity
$$((1+2^{-53})-1)2^{53}.$$
This quantity is equal to 1. However, if we write `((1+2^(-53))-1)*2^(53)` in Matlab, we get 0 instead!

For a less extreme but still important example, consider the quantity
$$((1+10^{-9})-1)10^{9}$$
This quantity is equal to 1, but in Matlab, this expression evaluates to about $1 + 8 \cdot 10^{-8}$. So, the error is somewhat small, but especially with an iterative algorithm such as Gradient Descent, one could worry that the errors accumulate as more steps are used in the algorithm.

In both arithmetic examples, the issue with performing computer arithmetic is how the number is represented in the computer. In double-precision floating-point arithmetic, which is the standard way to represent numbers in computers, a real number is stored as a 53-digit binary number with an 11-digit binary exponent. That is, a real number on a computer is stored in the form
$$\pm 1.b_1 b_2 \ldots b_{52} \times 2^{c_1 c_2 \ldots c_{11} - 2^{10} + 1},$$

where $b_1, \ldots, b_{52}, c_1, \ldots, c_{11} \in \{0,1\}$, and we interpret the decimal and the exponent as binary numbers.

For example, when Matlab computes $1 + 2^{-53}$, it is performing the following addition

$$\Big(1.0\ldots0 \times 2^0\Big) + \Big(1.0\ldots0 \times 2^{-53}\Big).$$

Now, in order to add the numbers, the computer tries to represent the smaller number so that its exponent is 2^0, matching the larger number's exponent. But since only 52 binary digits of the number 2^{-53} are stored, the addition becomes

$$\Big(1.0\ldots0 \times 2^0\Big) + \Big(0.0\ldots0 \times 2^0\Big) = 1.$$

Finally, subtracting 1 from this expression gives the result of 0 for the expression $(1+2^{-53})-1$.

Exercise 3.4. The Gradient Descent algorithm can behave quite badly for nontrivial reasons. Let $f\colon \mathbb{R} \to \mathbb{R}$ be defined by $f(x) = -x^{1000}$. Suppose we want to maximize f, starting at the initial guess $x^{(0)} = 1$ and using the parameter $\varepsilon = 1/100$. Show that the sequence of points $x^{(0)}, x^{(1)}, \ldots$ does not converge to 0. In fact, show that this sequence of points diverges!

On the other hand, choose a smaller ε to use in the Gradient Descent Algorithm such that the points $x^{(0)}, x^{(1)}, \ldots$ do converge to 0. (For the latter result you can freely use the following fact from analysis: if a sequence of nonnegative numbers $x^{(0)}, x^{(1)}, \ldots$ is strictly decreasing, i.e. if $x^{(i+1)} < x^{(i)}$ for all $i \geq 1$, then the sequence $x^{(0)}, x^{(1)}, \ldots$ converges to a nonnegative real number.)

In this way, a less naive version of the Gradient Descent algorithm will use a small value of ε exactly when the derivative of f is large. That is, the algorithm will adjust ε to depend on $\|\nabla f(x^{(n)})\|$.

3.2. **Newton's Method.** If we want to maximize $g\colon \mathbb{R} \to \mathbb{R}$ with $g \in C^2$, it suffices to solve the equation $g'(x) = 0$ for $x \in \mathbb{R}$. That is, if $f(x) := g'(x)$ for any $x \in \mathbb{R}$, then optimizing g reduces to seeking the zeros of f.

Algorithm 3.5. Newton's Method, a general way to find the roots of a differentiable function $f\colon \mathbb{R} \to \mathbb{R}$.

(1) Choose any point $x^{(0)} \in \mathbb{R}$.
(2) Compute the tangent line of f at $x^{(0)}$: $y(x) = f'(x^{(0)})(x - x^{(0)}) + f(x^{(0)})$.
(3) Find $x^{(1)}$ such that $y(x^{(1)}) = 0$. This is the intersection of the tangent line $y(x)$ with the x-axis. Note that $x^{(1)}$ satisfies

$$x^{(1)} = x^{(0)} - \frac{f(x^{(0)})}{f'(x^{(0)})}.$$

(4) Return to step (2), but replace $x^{(0)}$ with $x^{(1)}$. More generally, at the n^{th} iteration of the algorithm, compute the tangent line of f at $x^{(n)}$ in step (2), and then find an $x^{(n+1)}$ in step (3) which is a zero of the tangent line. So, in general we iterate the following equation.

$$x^{(n+1)} = x^{(n)} - \frac{f(x^{(n)})}{f'(x^{(n)})}.$$

Example 3.6 (**The Babylonian Square Root Algorithm**)**.** Let $M > 0$ be a fixed constant. Let $f(x) = x^2 - M$ for any $x \in \mathbb{R}$. Let $x^{(0)} = M$. Then Newton's Method gives the recursion.

$$x^{(n+1)} = x^{(n)} - \frac{f(x^{(n)})}{f'(x^{(n)})} = x^{(n)} - \frac{((x^{(n)})^2 - M)}{2x^{(n)}} = \frac{1}{2}\left(x^{(n)} + \frac{M}{x^{(n)}}\right).$$

Then the sequence $x^{(0)}, x^{(1)}, \ldots$ converges to the positive square root of M. (We will not prove that convergence occurs.) This recursion was known to the Babylonians. And it is still used today by our computers to calculate square roots.

For example, if $M = 2$, we have

$$x^{(1)} = \frac{1}{2}(1 + 2) = 1.5.$$

$$x^{(2)} = \frac{1}{2}(3/2 + 4/3) = 17/12 \approx 1.4167$$

$$x^{(3)} = \frac{1}{2}(17/12 + 24/17) = 577/408 \approx 1.414216$$

So, we can see that this algorithm converges to the square root rather rapidly.

Exercise 3.7 (Optional)**.** To see an illustration of Newton's Method, see the Applet, Newton Example. In many examples, it only takes a few iterations of the algorithm to get a good approximation for a zero of f.

Write a Matlab program that plots the first few iterates of Newton's method, as in the above applet. (Or, use a different programming language if you prefer it.) You can test your program using, e.g. the function $f(x) = x^2 - 1$ with initial guess $x_0 = 2$.

Exercise 3.8. Recall that Newton's method is an algorithm for finding zeros of a function f. It consists in iterating the map

$$\varphi(x) := x - (f'(x))^{-1} f(x).$$

Thus, we start with some given $x^{(0)}$ and define $x^{(1)} := \varphi(x^{(0)})$, $x^{(2)} := \varphi(x^{(1)})$, etc.

This problem is devoted to an analysis of the convergence of Newton's method. For simplicity, we work in one dimension, i.e. we set $n = 1$. Without loss of generality, we assume that the zero of f we are interested in is at the origin: $f(0) = 0$. We shall show that, assuming $f'(0)$ is invertible and $f \in C^2$, the sequence $x^{(0)}, x^{(1)}, \ldots$ converges to 0 provided $x^{(0)}$ is close enough to 0.

Let $K > 1$, $R > 0$ and suppose that

$$K^{-1} \le |f'(x)| \le K, \qquad |f''(x)| \le K, \qquad \forall\, |x| \le R. \tag{1}$$

(i) Using Taylor's Theorem, show that there exists $\varepsilon > 0$ such that, if $|x| < \varepsilon$, then

$$|f(0) - (f(x) - xf'(x))| \le K\,|x|^2.$$

(ii) Suppose $|x^{(0)}| < \varepsilon$. Then from part (i), $|f(x^{(0)}) - x^{(0)} f'(x^{(0)}))| \le K|x^{(0)}|^2$. Using the definition of $x^{(1)}$, deduce that

$$|x^{(1)}| \le K|f(x^{(0)}) - x^{(0)} f'(x^{(0)}))| \le K^2 |x^{(0)}|^2.$$

(iii) Let $n \ge 1$. Suppose $|x^{(n)}| < \varepsilon$. Show that

$$|x^{(n+1)}| \le K^2 |x^{(n)}|^2.$$

(iv) Conclude that $\lim_{n\to\infty} |x^{(n)}| = 0$, as desired.

In the case that we are looking for a local extremum of a function $g\colon \mathbb{R} \to \mathbb{R}$, we applied Newton's method to $f := g'$. If we rewrite the iterative equation of Newton's method in terms of f, we get

$$x^{(n+1)} = x^{(n)} - \frac{g'(x^{(n)})}{g''(x^{(n)})}, \qquad \forall n \geq 1.$$

This formula can be generalized to a function of n variables as follows

Algorithm 3.9. Newton's Method Version 2, a general way to find the local extrema of a function $g\colon \mathbb{R}^n \to \mathbb{R}$.

(1) Choose any point $x^{(0)} \in \mathbb{R}$.
(2) Inductively define $x^{(0)}, x^{(1)}, \ldots$ via the following recursion:

$$x^{(n+1)} = x^{(n)} - [D^2 g(x^{(n)})]^{-1} \nabla g(x^{(n)}).$$

Remark 3.10. Newton's Method implicitly assumes that we can compute all first and second order partial derivatives of the function f. In practice, this is not always possible. Therefore, there are various modifications to Newton's method which only require computation of the first derivatives of f. Nevertheless, as we have discussed, gradient descent methods may only find a local extremum of a function. That is, these methods may not be helpful for general optimization problems, where we seek a global extremum. For this reason, we will not discuss Newton's Method or gradient descent methods further.

Exercise 3.11 (Optional)**.** Write a Matlab program to implement Newton's Method for functions of two variables. Test your implementation on Rosenbrock's function: $f(x, y) = 100(y - x^2)^2 + (1 - x)^2$.

3.3. **Conjugate Gradient Methods.**

Let A be an $n \times n$ symmetric positive definite matrix. Let $b \in \mathbb{R}^n$ with $b \neq 0$. A general problem in linear algebra is to find $x \in \mathbb{R}^n$ such that

$$Ax = b.$$

Note that since A has no zero eigenvalues, the rank of A is n, so a solution x exists for the equation $Ax = b$. (Recall that a solution may not exist e.g. if the rank of A is less than n.)

Since A is invertible, we could just apply A^{-1} to both sides. However, computing A^{-1} can be computationally expensive. Moreover, if many eigenvalues of A are small, it can be difficult to compute A^{-1} accurately. In linear algebra class, we learned how to solve $Ax = b$ using Gaussian elimination. However, that method can also be computationally expensive, requiring on the order of n^3 arithmetic operations. (Gaussian elimination consists of n steps, and each step requires roughly n^2 arithmetic operations, so there are roughly n^3 arithmetic operations performed in total.)

Thankfully, there are ways to solve $Ax = b$ with far fewer arithmetic operations. One such method is called the Conjugate Gradient Method.

Exercise 3.12. Let A be an $n \times n$ symmetric positive definite matrix. Let $y, z \in \mathbb{R}^n$. Define a function $\langle \cdot, \cdot \rangle_A \colon \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ by

$$\langle y, z \rangle_A := y^T A z.$$

Show that $\langle \cdot, \cdot \rangle_A$ is an inner product on $\mathbb{R}^n$. That is, show:

- For any $y \in \mathbb{R}^n$ with $y \neq 0$, $\langle y, y\rangle_A > 0$.
- For any $y, z \in \mathbb{R}^n$, $\langle y, z\rangle_A = \langle z, y\rangle_A$.
- For any $y, z \in \mathbb{R}^n, \lambda \in \mathbb{R}$, $\langle \lambda y, z\rangle_A = \lambda\langle y, z\rangle_A$.
- For any $w, y, z \in \mathbb{R}^n, \lambda \in \mathbb{R}$, $\langle w + y, z\rangle_A = \langle w, z\rangle_A + \langle y, z\rangle_A$.

Then, a standard fact from linear algebra implies that the following function is a norm on $\mathbb{R}^n$.

$$\|y\|_A := \sqrt{\langle y, y\rangle_A} = \sqrt{y^T A y}.$$

Suppose $x \in \mathbb{R}^n$ satisfies $Ax = b$. We would like to find $y \in \mathbb{R}^n$ that is as close to x as possible. One way of formalizing the previous sentence is to find $y \in \mathbb{R}^n$ such that $\|y - x\|_A$ is as small as possible. Note that

$$\|y - x\|_A^2 = y^T A y - 2x^T A y + x^T A x = y^T A y - 2b^T y + x^T A x.$$

So, minimizing $\|y - x\|_A$ is equivalent to minimizing the following quadratic function of y:

$$f(y) := \frac{1}{2} y^T A y - b^T y.$$

The rough idea of the Conjugate Gradient method is to minimize $f(y)$ by selecting a particular basis of $\mathbb{R}^n$. Suppose $d^{(1)}, \ldots, d^{(n)}$ is a basis of $\mathbb{R}^n$. (This set of vectors may not be a basis in the algorithm, but for now let's pretend that it is a basis.) Then there exist $c_1, \ldots, c_n \in \mathbb{R}$ such that $y = \sum_{i=1}^n c_i d^{(i)}$, and

$$\begin{aligned} f(y) &= \frac{1}{2}\left(\sum_{i=1}^n c_i d^{(i)}\right)^T A \left(\sum_{i=1}^n c_i d^{(i)}\right) - b^T \sum_{i=1}^n c_i d^{(i)} \\ &= \frac{1}{2}\sum_{i=1}^n c_i^2 (d^{(i)})^T A d^{(i)} - \sum_{i=1}^n c_i b^T d^{(i)} + \frac{1}{2}\sum_{i \neq j} c_i c_j (d^{(i)})^T A d^{(j)}. \end{aligned}$$

The Conjugate Gradient Algorithm chooses the vectors $d^{(1)}, \ldots, d^{(n)}$ such that $\langle d^{(i)}, d^{(j)}\rangle_A = 0$ if $i, j \in \{1, \ldots, n\}$ with $i \neq j$. We then have

$$f(y) = \sum_{i=1}^n \left(\frac{1}{2} c_i^2 (d^{(i)})^T A d^{(i)} - c_i b^T d^{(i)}\right).$$

So minimizing $f(y)$ is equivalent to minimizing each term in the sum separately. To minimize each term in the sum separately, we can choose y (i.e. choose the c_i) so the derivative with respect to c_i is zero. That is, we choose the c_i such that

$$c_i = \frac{\langle b, d^{(i)}\rangle}{\langle d^{(i)}, d^{(i)}\rangle_A}.$$

Algorithm 3.13 (**Basic Version of Conjugate Gradient Algorithm**)**.** An algorithm for solving the matrix equation $Ax = b$ for $x \in \mathbb{R}^n$, where A is an $n \times n$ symmetric positive definite matrix, and $b \in \mathbb{R}^n$, $b \neq 0$.

- Let $x^{(1)} := 0$
- Let $r^{(1)} := Ax^{(1)} - b$. Let $d^{(1)} := b - Ax^{(1)}$.

- For every $1 \leq i \leq n-1$, recursively define

$$\alpha_i := -\frac{\langle r^{(i)}, d^{(i)}\rangle}{\langle d^{(i)}, d^{(i)}\rangle_A}. \tag{2}$$

$$x^{(i+1)} := x^{(i)} + \alpha_i d^{(i)}. \tag{3}$$

$$r^{(i+1)} := Ax^{(i+1)} - b. \tag{4}$$

$$\beta_{i+1} := \frac{\langle d^{(i)}, r^{(i+1)}\rangle_A}{\langle d^{(i)}, d^{(i)}\rangle_A}. \tag{5}$$

$$d^{(i+1)} := -r^{(i+1)} + \beta_{i+1} d^{(i)}. \tag{6}$$

For any $2 \leq i \leq n$, if we ever have $r^{(i)} = 0$, the algorithm terminates and outputs $x^{(i)}$.

If $r^{(i)} \neq 0$ for all $1 \leq i \leq n$, the algorithm outputs $x^{(n+1)}$.

Remark 3.14. If $r^{(i)} = 0$ for some $1 \leq i \leq n$, then $x^{(i)}$ satisfies $Ax^{(i)} = b$, so we have solved the system $Ax = b$.

Since $r^{(i)}$ measures the error between $Ax^{(i)}$ and b, $r^{(i)}$ is called the i^{th} **residual** of the Conjugate Gradient algorithm.

Lemma 3.15. *The vectors produced by Algorithm 3.13 satisfy the following properties. Let $1 \leq k \leq n$. Assume that $\langle d^{(i)}, d^{(j)}\rangle_A = 0$ for any $1 \leq i, j \leq k-1$ with $i \neq j$. Then*

$$\langle r^{(k)}, d^{(i)}\rangle = 0, \qquad \forall\, 1 \leq i < k.$$

Also, $x^{(k+1)}$ is the unique minimum of the function $f(y) := \frac{1}{2}y^T Ay - b^T y$ over the set

$$\{x^{(1)} + z \colon z \in \text{span}(d^{(1)}, \ldots, d^{(k)})\}.$$

Proof. From Exercise 2.61, y is the unique minimum of f on the set

$$\{x^{(1)} + z \colon z \in \text{span}(d^{(1)}, \ldots, d^{(k)})\}$$

if and only if $\nabla f(y) = Ay - b$ is orthogonal to $\text{span}(d^{(1)}, \ldots, d^{(k)})$. So by (4), to complete the proof, it suffices to prove the either assertion by induction on k.

In the case $k = 1$, we have $x^{(2)} = x^{(1)} + \alpha_1 d^{(1)}$ where α_1 is chosen to minimize $f(x^{(1)} + \alpha_1 d^{(1)})$. To see this, note that

$$\frac{d}{d\alpha_1} f(x^{(1)} + \alpha_1 d^{(1)}) = \langle \nabla f(x^{(1)} + \alpha_1 d^{(1)}), d^{(1)}\rangle = \langle Ax^{(1)} + \alpha_1 Ad^{(1)} - b, d^{(1)}\rangle.$$

So, solving for α_1 in the equation $\frac{d}{d\alpha_1} f(x^{(1)} + \alpha_1 d^{(1)}) = 0$ gives

$$\alpha_1 = \frac{\langle b - Ax^{(1)}, d^{(1)}\rangle}{\langle d^{(1)}, d^{(1)}\rangle_A} = -\frac{\langle r^{(1)}, d^{(1)}\rangle}{\langle d^{(1)}, d^{(1)}\rangle_A}.$$

That is, $x^{(2)}$ is chosen to minimize f over the set of all possible $\alpha^{(1)} \in \mathbb{R}$. So, the base case of the induction holds.

We now prove the inductive step. Assume $\langle r^{(k-1)}, d^{(i)}\rangle = 0 \;\forall\; 1 \leq i < k-1$. We then consider the case when $k-1$ becomes k.

$$r^{(k)} \overset{(4)}{=} A(x^{(k)} - x^{(k-1)}) + Ax^{(k-1)} - b \overset{(4)\wedge(3)}{=} \alpha_{k-1} Ad^{(k-1)} + r^{(k-1)}. \tag{$*$}$$

Taking the inner product of both sides with $d^{(k-1)}$,

$$\langle r^{(k)}, d^{(k-1)}\rangle = \langle r^{(k-1)}, d^{(k-1)}\rangle + \alpha_{k-1}\langle d^{(k-1)}, d^{(k-1)}\rangle_A \overset{(2)}{=} 0.$$

And if $1 \le i \le k-2$, we have

$$\langle r^{(k)}, d^{(i)}\rangle \overset{(*)}{=} \langle r^{(k-1)}, d^{(i)}\rangle + \alpha_{k-1}\langle d^{(k-1)}, d^{(i)}\rangle_A = 0.$$

Here the first term in the sum is zero by the inductive hypothesis, and the second term is zero by the first assumption of the lemma. This completes the inductive step, thereby completing the proof. $\square$

Lemma 3.16. *Fix $1 \le k \le n$. Assume that $r_i \neq 0$ for all $1 \le i \le k$. Then the vectors produced by Algorithm 3.13 satisfy the following properties.*

(i) $\operatorname{span}(r^{(1)}, \dots, r^{(k)}) = \operatorname{span}(r^{(1)}, Ar^{(1)} \dots, A^{k-1}r^{(1)})$.
(ii) $\operatorname{span}(d^{(1)}, \dots, d^{(k)}) = \operatorname{span}(r^{(1)}, Ar^{(1)} \dots, A^{k-1}r^{(1)})$.
(iii) $\langle d^{(i)}, d^{(k)}\rangle_A = 0$ *for all* $1 \le i \le k-1$.
(iv) $\langle r^{(i)}, r^{(k)}\rangle = 0$ *for all* $1 \le i \le k-1$.

Proof. We prove all assertions simultaneously by induction on k. The case $k = 1$ is clear. Assume (i)-(iv) hold for the case k. We prove that (i)-(iv) hold for $k+1$.
Step 1. Inductive Step of (i). By the inductive hypothesis,

$$r^{(k)} \in \operatorname{span}(r^{(1)}, Ar^{(1)} \dots, A^{k-1}r^{(1)}), \qquad d^{(k)} \in \operatorname{span}(r^{(1)}, Ar^{(1)} \dots, A^{k-1}r^{(1)}).$$

Applying A to the second expression, $Ad^{(k)} \in \operatorname{span}(Ar^{(1)}, \dots, A^k r^{(1)})$. Then

$$r^{(k+1)} \overset{(4)}{=} A(x^{(k+1)} - x^{(k)}) + Ax^{(k)} - b \overset{(4)\wedge(3)}{=} \alpha_k A d^{(k)} + r^{(k)}. \qquad (*)$$

Since $r^{(k)} \in \operatorname{span}(r^{(1)}, Ar^{(1)} \dots, A^{k-1}r^{(1)})$, we get $r^{(k+1)} \in \operatorname{span}(r^{(1)}, Ar^{(1)}, \dots, A^k r^{(1)})$ from $(*)$. So, using the inductive hypothesis for (i), we get

$$\operatorname{span}(r^{(1)}, \dots, r^{(k+1)}) \subseteq \operatorname{span}(r^{(1)}, Ar^{(1)} \dots, A^k r^{(1)}).$$

We now prove the reverse inclusion. Applying the inductive hypothesis of (ii) and (i),

$$A^k r^{(1)} = A(A^{(k-1)} r^{(1)}) \in \operatorname{span}(Ad^{(1)}, \dots, Ad^{(k)}) \subseteq \operatorname{span}(r^{(1)}, \dots, r^{(k)}, Ad^{(k)}).$$

From $(*)$, $Ad^{(k)} = (r^{(k+1)} - r^{(k)})/\alpha_k$. (Applying (6) iteratively and using the inductive hypothesis of (iv), note that $\langle d^{(k)}, r^{(k)}\rangle = -\langle r^{(k)}, r^{(k)}\rangle \neq 0$, so we have not divided by zero since $\alpha_k \neq 0$. Moreover, division by zero does not occur in (2) or (5) for this reason.) So,

$$A^k r^{(1)} \in \operatorname{span}(r^{(1)}, \dots, r^{(k+1)}).$$

Combining this with the inductive hypothesis for (i),

$$\operatorname{span}(r^{(1)}, Ar^{(1)} \dots, A^k r^{(1)}) \subseteq \operatorname{span}(r^{(1)}, \dots, r^{(k+1)}).$$

In conclusion, we verified that (i) holds for $k+1$, completing the inductive step for (i).

Step 2. Inductive Step of (ii).

$$\begin{aligned}
&\operatorname{span}(d^{(1)},\dots,d^{(k+1)})\\
&\quad=\operatorname{span}(d^{(1)},\dots,d^{(k)},r^{(k+1)}), && \text{by (6)}\\
&\quad=\operatorname{span}(r^{(1)},Ar^{(1)},\dots,A^{k-1}r^{(1)},r^{(k+1)}), && \text{by the inductive hypothesis for (ii)}\\
&\quad=\operatorname{span}(r^{(1)},\dots,r^{(k)},r^{(k+1)}), && \text{by the inductive hypothesis for (i)}\\
&\quad=\operatorname{span}(r^{(1)},Ar^{(1)},\dots,A^{k}r^{(1)}), && \text{by (i) for } k+1
\end{aligned}$$

Step 3. Inductive Step of (iii). From (6),

$$\langle d^{(k+1)},d^{(i)}\rangle_A=-\langle r^{(k+1)},d^{(i)}\rangle_A+\beta_{k+1}\langle d^{(k)},d^{(i)}\rangle_A. \qquad (**)$$

When $i=k$, the right side of $(**)$ is zero by (5). For any $1\le i<k$, we first apply Lemma 3.15 to get $\langle r^{(k+1)},d^{(i)}\rangle=0$. Then, applying (ii) twice

$$Ad^{(i)}\in\operatorname{span}(Ar^{(1)},\dots,A^{i}r^{(1)})\subseteq\operatorname{span}(d^{(1)},\dots,d^{(i)}).$$

Therefore, $\langle r^{(k+1)},d^{(i)}\rangle_A=0$ for any $1\le i<k$. So if $1\le i<k$, the first term on the right of $(**)$ is zero, and the second term is zero as well by the inductive hypothesis for (iii). In conclusion, $(**)$ is zero for any $1\le i\le k$, completing the inductive step.
Step 4. Inductive Step of (iv).

From Lemma 3.15, $\langle r^{(k)},d^{(i)}\rangle=0$ for all $1\le i\le k-1$. From (6), $r^{(i)}\in\operatorname{span}(d^{(i)},d^{(i-1)})$ for all $2\le i\le k-1$. Therefore, $\langle r^{(k)},r^{(i)}\rangle=0$ for any $2\le i\le k-1$. Finally, by definition of $r^{(1)}$, we have $\langle r^{(k)},r^{(1)}\rangle=\langle r^{(k)},d^{(1)}\rangle=0$ by Lemma 3.15. □

Theorem 3.17. *Let A be a real symmetric positive definite $n\times n$ matrix, and let $b\in\mathbb{R}^n$. Then the output y of Algorithm 3.13 solves the equation $Ay=b$.*
Put another way, there exists $1\le i\le n+1$ such that $r^{(i)}=0$.

Proof. Let $x\in\mathbb{R}^n$ such that $Ax=b$. By Remark 3.14, it suffices to consider the case that $r^{(i)}\neq0$ for all $1\le i\le n$. We will show that $r^{(n+1)}=0$.

From Lemma 3.16(iv), the vectors $r^{(1)},\dots,r^{(n)}$ are nonzero vectors which are orthogonal with respect to the inner product $\langle\cdot,\cdot\rangle_A$. Therefore, the span of $r^{(1)},\dots,r^{(n)}$ is n-dimensional. So, combining Lemma 3.16(i) and (ii), the span of $d^{(1)},\dots,d^{(n)}$ is also n-dimensional. Therefore, there exist scalars $c_1,\dots,c_n$ such that

$$x-x^{(1)}=\sum_{i=1}^{n}c_id^{(i)}. \qquad (*)$$

Fix $1\le j\le n$. Take the inner product of both sides of $(*)$ with $d^{(j)}$ and apply Lemma 3.16(iii) to get

$$c_j=\frac{\langle x-x^{(1)},d^{(j)}\rangle_A}{\langle d^{(j)},d^{(j)}\rangle_A}. \qquad (**)$$

Applying (3) iteratively, we have

$$x^{(j)}=x^{(1)}+\alpha_1d^{(1)}+\cdots+\alpha_{j-1}d^{(j-1)}. \qquad (***)$$

Take the inner product of both sides with $d^{(j)}$ and apply Lemma 3.16(iii) to get

$$\langle d^{(j)}, x^{(j)}\rangle_A = \langle d^{(j)}, x^{(1)}\rangle_A.$$

So, using this equality, $Ax = b$, then (4),

$$\langle d^{(j)}, x - x^{(1)}\rangle_A = \langle d^{(j)}, x - x^{(j)}\rangle_A = \langle d^{(j)}, b - Ax^{(j)}\rangle = -\langle d^{(j)}, r^{(j)}\rangle$$

Substituting this equality into $(**)$, then using (2),

$$c_j = -\frac{\langle d^{(j)}, r^{(j)}\rangle}{\langle d^{(j)}, d^{(j)}\rangle_A} = \alpha_j.$$

Comparing $(*)$ and $(***)$ for $j = n+1$ concludes the proof. $\square$

We now mention a slight simplification of Algorithm 3.18. Using Lemma 3.16,

$$\alpha_i \overset{(2)}{=} -\frac{\langle r^{(i)}, d^{(i)}\rangle}{\langle d^{(i)}, d^{(i)}\rangle_A} \overset{(6)}{=} -\frac{\langle r^{(i)}, -r^{(i)} + \beta_{i-1}d^{(i-1)}\rangle}{\langle d^{(i)}, d^{(i)}\rangle_A} = \frac{\langle r^{(i)}, r^{(i)}\rangle}{\langle d^{(i)}, d^{(i)}\rangle_A}.$$

Now, from $(*)$ in Lemma 3.16, $r^{(i+1)} - r^{(i)} = \alpha_i A d^{(i)}$. So, by Lemma 3.16

$$\begin{aligned}\beta_{i+1} &\overset{(5)}{=} \frac{\langle d^{(i)}, r^{(i+1)}\rangle_A}{\langle d^{(i)}, d^{(i)}\rangle_A} = \frac{\langle r^{(i+1)} - r^{(i)}, r^{(i+1)}\rangle}{\langle r^{(i+1)} - r^{(i)}, d^{(i)}\rangle} = \frac{\langle r^{(i+1)}, r^{(i+1)}\rangle}{\langle -r^{(i)}, d^{(i)}\rangle}\\ &\overset{(6)}{=} \frac{\langle r^{(i+1)}, r^{(i+1)}\rangle}{\langle -r^{(i)}, -r^{(i)} + \beta_{i-1}d^{(i-1)}\rangle} = \frac{\langle r^{(i+1)}, r^{(i+1)}\rangle}{\langle r^{(i)}, r^{(i)}\rangle}.\end{aligned}$$

So, we can adjust the formulas (2) and (5) in Algorithm 3.13 as follows.

Algorithm 3.18 (Conjugate Gradient Algorithm). An algorithm for solving the matrix equation $Ax = b$ for $x \in \mathbb{R}^n$, where A is an $n \times n$ symmetric positive definite matrix, and $b \in \mathbb{R}^n$, $b \neq 0$.

- Let $x^{(1)} := 0$
- Let $r^{(1)} := Ax^{(1)} - b$. Let $d^{(1)} := b - Ax^{(1)}$.
- For every $1 \leq i \leq n-1$, recursively define

$$\begin{aligned}\alpha_i &:= -\frac{\langle r^{(i)}, r^{(i)}\rangle}{\langle d^{(i)}, d^{(i)}\rangle_A}.\\ x^{(i+1)} &:= x^{(i)} + \alpha_i d^{(i)}.\\ r^{(i+1)} &:= r^{(i)} + \alpha_i A d^{(i)}.\\ \beta_{i+1} &:= \frac{\langle r^{(i+1)}, r^{(i+1)}\rangle}{\langle r^{(i)}, r^{(i)}\rangle_A}.\\ d^{(i+1)} &:= -r^{(i+1)} + \beta_{i+1} d^{(i)}.\end{aligned}$$

For any $1 \leq i \leq n-1$, if we ever have $r^{(i+1)} = 0$, the algorithm terminates and outputs $x^{(i)}$.

If $r^{(i)} \neq 0$ for all $1 \leq i \leq n$, the algorithm outputs $x^{(n+1)}$.

Exercise 3.19. At iteration i, Algorithm 3.18 only requires the values of $x^{(i)}, x^{(i-1)}, r^{(i)}, r^{(i-1)}$ and $d^{(i)}, d^{(i-1)}$. So, the memory storage requirement of Algorithm 3.18 (and 3.13) is fairly small.

Give a bound on the maximum amount of numbers that Algorithm 3.18 needs to store in memory, at any point in time while the algorithm is running.

Compared to Algorithm 3.13, Algorithm 3.18 requires a few less matrix multiplications at each iteration, which saves some time.

Give a bound on the total number of arithmetic operations that Algorithm 3.18 performs over the duration of the entire algorithm. Your bound should depend on the size n of the matrix A and on the number m of nonzero entries of A. (Your bound should be something like mn)

Remark 3.20. Even if an $n \times n$ matrix has e.g. $10n$ nonzero entries, Gaussian elimination still might require around n^3 arithmetic operations, since even the first step of Gaussian elimination could create a matrix with around n^2 nonzero entries. So, Gaussian elimination is strictly worse than the Conjugate Gradient method, in terms of the number of arithmetic operations that may need to be done.

Remark 3.21. Some modern algorithms require significantly less arithmetic operations than the Conjugate Gradient method in Algorithm 3.18. In particular, some algorithms almost only require a number of arithmetic operations proportional to the number of nonzero entries in A. However, a discussion of these algorithms is beyond the scope of this course.

3.4. **Least Squares.** In our discussion of the conjugate gradient method, we found the solution $x \in \mathbb{R}^n$ of the equation $Ax = b$ where A is an $n \times n$ real symmetric positive definite matrix, and $b \in \mathbb{R}^n$. Since A is positive definite, A has rank n, so a solution x exists. In general the matrix A may not be positive definite, or the matrix A may not have full rank, so a solution x is not guaranteed to exist.

In this section, we let $m \geq n$, we let A be any real $m \times n$ matrix of rank n, and we let $b \in \mathbb{R}^m$. So, a solution to the equation

$$Ax = b$$

still exists, but we cannot use the Conjugate Gradient Method. Instead of trying to solve $Ax = b$ directly, we instead minimize

$$f(x) := \frac{1}{2} \|Ax - b\|^2, \qquad x \in \mathbb{R}^n.$$

Exercise 3.22. Let A be a real $m \times n$ matrix. Let $x \in \mathbb{R}^n$ and let $b \in \mathbb{R}^m$. Show that the function $f \colon \mathbb{R}^n \to \mathbb{R}$ defined by $f(x) = \frac{1}{2} \|Ax - b\|^2$ is convex. Moreover, show that

$$\nabla f(x) = A^T(Ax - b), \qquad D^2 f(x) = A^T A.$$

From Exercises 3.22 and 2.39, if $\nabla f(x) = 0$, then x is a global minimum of f. That is, if

$$A^T A x = A^T b,$$

then x is a global minimum of f.

Exercise 3.23. Let A be an $m \times n$ real matrix with $m \geq n$. Then A has rank n if and only if $A^T A$ is positive definite.

(Hint: $A^T A$ is automatically positive semidefinite by Exercise 2.21.)

From Exercise 3.23, if A has rank n, then $A^T A$ is invertible. We summarize the above discussion

Proposition 3.24 (**Linear Least Squares**). *Let $m \geq n$. Let A be a real $m \times n$ matrix with rank n. Let $b \in \mathbb{R}^m$. Then the global minimum of $\|Ax - b\|^2$ among all $x \in \mathbb{R}^n$ occurs when*

$$x = (A^T A)^{-1} A^T b.$$

From Exercise 3.23, we could try to applying the Conjugate Gradient method to the equation $A^T Ax = A^T b$ in order to find x. But the multiplication $A^T A$ itself could require as many as n^3 arithmetic operations, which is fairly costly. (The multiplication of two $n \times n$ matrices using the most naive method takes around n^3 arithmetic operations; a popular implementation of matrix multiplication known as Strassen's algorithm uses about $n^{2.8}$ matrix multiplications; the best known matrix multiplication algorithm uses $Cn^{2.373}$ arithmetic operations for a large constant C.)

Inverting $A^T A$ directly could also introduce numerical error. So, although Proposition 3.24 theoretically finds the minimum of $\|Ax - b\|^2$, in practice the formula $x = (A^T A)^{-1} A^T b$ may not be so useful.

There are a few algorithms for minimizing $\|Ax - b\|^2$, such as

- Computing the Cholesky decomposition of $A^T A$. (That is, we write $A^T A = R^T R$ where R is an upper triangular $n \times n$ matrix with positive diagonal elements.)
- Computing the QR decomposition of the matrix A. (That is, we write $A\Pi = Q \begin{pmatrix} R \\ 0 \end{pmatrix}$, where Π is an $n \times n$ permutation matrix, Q is an $m \times m$ orthogonal matrix, and R is an upper triangular $n \times n$ matrix with positive diagonal elements.)
- Compute the singular value decomposition of the matrix A. (That is, we write $A = U \begin{pmatrix} S \\ 0 \end{pmatrix} V$, where U is an $m \times m$ orthogonal matrix, V is an $n \times n$ orthogonal matrix, and S is an $n \times n$ diagonal matrix with nonnegative entries.)

Since matrix decompositions are a topic of Math 115B, we will not focus on these methods. We instead mention an iterative method for minimizing $\|Ax - b\|^2$. Below, we list some simplifying assumptions for the recursive algorithm.

Algorithm 3.25 (**Recursive Least Squares/ Online Learning**). Let $m \geq n$, let A be an $m \times n$ real matrix. Let $a^{(1)}, \ldots, a^{(m)} \in \mathbb{R}^n$ be row vectors which are the rows of A (data), and let $b \in \mathbb{R}^m$. For any $j \geq n$, let

$$A_j := \begin{pmatrix} a^{(1)} \\ \vdots \\ a^{(j)} \end{pmatrix}, \qquad b^{(j)} := \begin{pmatrix} b_1 \\ \vdots \\ b_j \end{pmatrix}.$$

Assume that A_n has rank n (so that $A_n^T A_n$ is invertible). Define

$$x^{(n)} := (A_n^T A_n)^{-1} A_n^T b^{(n)} \in \mathbb{R}^n, \qquad P_n := (A_n^T A_n)^{-1}.$$

For any $j \geq n$, define

$$P_{j+1} = P_j - \frac{P_j (a^{(j+1)})^T a^{(j+1)} P_j^T}{1 + a^{(j+1)} P_j (a^{(j+1)})^T}.$$
$$x^{(j+1)} = x^{(j)} + P_{j+1} (a^{(j+1)})^T (b_{j+1} - a^{(j+1)} x^{(j)}).$$

The vectors $x^{(n)},\ldots,x^{(m)}$ recursively minimize the quantity $\|Ax-b\|^2$ in the following sense.

Proposition 3.26. *Let $\lambda>0$. Let $x^{(n)},\ldots,x^{(m)}$ be the output of Algorithm 3.25. Let $n\leq j\leq m$. Define $f_j\colon\mathbb{R}^n\to\mathbb{R}$ by*

$$f_j(x):=\frac{1}{2}\sum_{i=1}^{j}(\langle x,a^{(i)}\rangle-b_i)^2,\qquad x\in\mathbb{R}^n.$$

Then $x^{(j)}$ minimizes f_j on $\mathbb{R}^n$. In particular, when $j=m$, $x^{(m)}$ minimizes $\|Ax-b\|^2$.

Proof. We induct on j. The case $j=n$ follows by definition of $x^{(n)}$ and by Proposition 3.24. We now complete the inductive step. Assume the Proposition holds for j, and consider the case $j+1$. Define $G_j:=(A_j^TA_j)$.

First, note that

$$G_{j+1}=\begin{pmatrix}A_j^T & (a^{(j+1)})^T\end{pmatrix}\begin{pmatrix}A_j\\ a^{(j+1)}\end{pmatrix}=A_j^TA_j+(a^{(j+1)})^Ta^{(j+1)}=G_j+(a^{(j+1)})^Ta^{(j+1)}.\qquad(*)$$

By the inductive hypothesis and Proposition 3.24, we have $x^{(j)}=G_j^{-1}A_j^Tb^{(j)}$. So,

$$A_j^Tb^{(j)}=G_jG_j^{-1}A_j^Tb^{(j)}=G_jx^{(j)}\overset{(*)}{=}(G_{j+1}-(a^{(j+1)})^Ta^{(j+1)})x^{(j)}.\qquad(**)$$

From Proposition 3.24, the minimum of f_{j+1} on $\mathbb{R}^n$ occurs when

$$\begin{aligned}x&=G_{j+1}^{-1}A_{j+1}^Tb^{(j+1)}=G_{j+1}^{-1}\begin{pmatrix}A_j\\ a^{(j+1)}\end{pmatrix}^T\begin{pmatrix}b^{(j)}\\ b_{j+1}\end{pmatrix}=G_{j+1}^{-1}(A_j^Tb^{(j)}+b_{j+1}(a^{(j+1)})^T)\\&\overset{(**)}{=}G_{j+1}^{-1}\Big(G_{j+1}x^{(j)}-(a^{(j+1)})^Ta^{(j+1)}x^{(j)}+b_{j+1}(a^{(j+1)})^T\Big)\\&=x^{(j)}+G_{j+1}^{-1}(a^{(j+1)})^T(b_{j+1}-a^{(j+1)}x^{(j)}).\end{aligned}$$

Comparing this formula to the definition of $x^{(j+1)}$ in Algorithm 3.25, it remains to manipulate the G_{j+1}^{-1} term. Applying Exercise 3.27 to $(*)$,

$$G_{j+1}^{-1}=(G_j+(a^{(j+1)})^Ta^{(j+1)})^{-1}=G_j^{-1}-\frac{G_j^{-1}(a^{(j+1)})^Ta^{(j+1)}G_j^{-1}}{1+a^{(j+1)}G_j^{-1}(a^{(j+1)})^T}.$$

Finally, note that $P_n=G_n^{-1}$, and since the matrices P_j and G_j^{-1} satisfy the same recursion, we get $P_j=G_j^{-1}$, completing the proof.

□

Exercise 3.27. Show the following identity. Let A be an $r\times r$ real matrix, let U be an $r\times s$ real matrix, and let V be an $s\times r$ real matrix. Assume that A is invertible and that $I+VA^{-1}U$ is invertible, where I is the $s\times s$ identity matrix. Then $A+UV$ is invertible and

$$(A+UV)^{-1}=A^{-1}-(A^{-1}U)(I+VA^{-1}U)^{-1}(VA^{-1}).$$

In particular, if $s=1$, we get the Sherman-Morrison formula:

$$(A+UV)^{-1}=A^{-1}-\frac{A^{-1}UVA^{-1}}{1+VA^{-1}U}.$$

Exercise 3.28. Give a bound for the number of arithmetic operations needed in Algorithm 3.25. Assume that P_n is known, so that computing P_n does not require any arithmetic operations. (Hint: your bound should be something like $(m-n)n^2$.) Compare this bound to simply minimizing $\|Ax-b\|^2$ directly. (In that case, you should need about mn^2 arithmetic operations.)

The key point here is that, once P_n is known, recursive least squares is much better. For example, if $m-n=10$, then recursive least squares requires around $10n^2$ arithmetic operations. But minimizing $\|Ax-b\|^2$ directly would require mn^2 operations, which is much larger.

Remark 3.29. Here are some warnings about the data analysis interpretation of least squares.

- Gradient descent should behave well for linear least squares minimization (since the function $x \mapsto \|Ax-b\|^2$ is convex), but minimizing more complicated objective functions with many local minima could result in poor performance of gradient descent. We have mentioned this point before.
- Data analysis involves fitting parameters to data. As von Neumann once remarked:

  With four parameters I can fit an elephant, and with five I can make him wiggle his trunk.

  The point of the quote is that allowing many free parameters in a model can allow you to make almost any conclusion. For example, the exponential function e^x looks a lot like the polynomial $1+x+x^2/2+x^3/6+x^4/24$. So, you could say that this polynomial is closest to the exponential function, so these two functions are very alike. But these two functions are actually quite different.
- Data analysis also involves finding correlations in data. However, if you look at enough categories of data, some of them will always be correlated with each other, essentially by the pigeonhole principle (see Exercise 3.30). For example, the chocolate consumption of a country is strongly correlated with the percentage of Nobel laureates in that country. (http://www.nejm.org/doi/full/10.1056/NEJMon1211064) But is this correlation meaningful or informative in any sense?

Exercise 3.30. Let $x^{(1)},\ldots,x^{(m)}$ be vectors in $\mathbb{R}^2$ such that $\|x^{(i)}\|=1$ for all $1\leq i\leq m$. Show that there exist $i,j\in\{1,\ldots,m\}$ with $i\neq j$ such that $\langle x^{(i)},x^{(j)}\rangle > 1-100/m^2$.

Exercise 3.31 (Logistic Regression). Let $x^{(1)},\ldots,x^{(m)}\in\mathbb{R}^n$ and let $y_1,\ldots,y_m\in\{0,1\}$. For the sake of intuition, we can think of each vector $x^{(i)}$ as a vector of words in an email, and y_i classifies email $i\in\{1,\ldots,m\}$ as either spam ($y_i=1$) or not spam ($y_i=0$). Given this data, we would like to find a way to classify future emails as spam or not spam. (This is what a spam filter does.) For any $t\in\mathbb{R}$, define the logistic function $g\colon\mathbb{R}\to(0,1)$ by

$$g(t)=\frac{1}{1+e^{-t}}.$$

The function g is meant to be a differentiable approximation to a function whose output is either 0 or 1.

First, verify that $g'(t) = g(t)(1-g(t))$ for any $t \in \mathbb{R}$. Then, consider the log-likelihood function $L\colon \mathbb{R}^n \to \mathbb{R}$ defined by

$$L(z) := \log\left(\prod_{i=1}^{m}[g(\langle z, x^{(i)}\rangle)]^{y_i}[1-g(\langle z, x^{(i)}\rangle)]^{1-y_i}\right), \qquad \forall\, z \in \mathbb{R}^n.$$

We would like to maximize L. The idea here is that if L is large, then z is a set of parameters (or "weights") that accurately classifies known emails as spam or not spam. So, once we find z, and if we have some new email $x \in \mathbb{R}^n$, then $g(\langle z, x^{(i)}\rangle) \approx 0$ means the new email is probably not spam, and $g(\langle z, x^{(i)}\rangle) \approx 1$ means the new email is probably spam.

Show that

$$\nabla L(z) = \sum_{i=1}^{m}(y_i - g(\langle z, x^{(i)}\rangle))x^{(i)}, \qquad \forall\, z \in \mathbb{R}^n.$$

This computation then gives the formula for a gradient ascent method for maximizing L.

Exercise 3.32. Define

$$A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 1 & 3 \\ 2 & 3 \end{pmatrix}, \qquad b = \begin{pmatrix} 1 \\ 3 \\ 4 \\ 6 \end{pmatrix}.$$

Minimize the function $\|Ax-b\|^2$ over all $x \in \mathbb{R}^2$, either by hand, or using a computer program that you write yourself. In either case, use the recursive least squares method.

Verify that the x you found does actually minimize $\|Ax-b\|^2$.

4. Linear Programming

4.1. **Introduction.** The invention of linear programming is attributed to Kantorovich, who developed the theory for the USSR starting in 1939 to improve the allocation of resources for factories. Koopmans investigated the same problems contemporaneously. Both were awarded the Nobel Prize in Economics in 1975 for their work.

Here is a sample of the kind of problem Kantorovich considered.

Example 4.1. Suppose we have m machines which can produce any of n products, and the machines are running all day. For any machine $i \in \{1,\ldots,m\}$, and for any product $j \in \{1,\ldots,n\}$, let $a_{ij} \geq 0$ be the amount of product j that machine i can produce in one day. For any $i \in \{1,\ldots,m\}$, $j \in \{1,\ldots,n\}$, let $x_{ij} \geq 0$ be fraction of the day that machine i is assigned to produce product j. For any $j \in \{1,\ldots,m\}$, let z_j be the amount of product j that is produced by all machines in total in a single day.

In summary, the constraints on the machines are:

$$z_j = \sum_{i=1}^{m} a_{ij}x_{ij}, \qquad \forall\, 1 \leq j \leq n.$$

$$\sum_{j=1}^{n} x_{ij} \leq 1, \qquad \forall\, 1 \leq i \leq m.$$

$$x_{ij} \geq 0, \qquad \forall\, 1 \leq i \leq m,\ 1 \leq j \leq n.$$

And we would like to maximize some linear function of the total production. That is there is some vector $c \in \mathbb{R}^m$ such that we want to maximize

$$\sum_{j=1}^{m} c_j z_j = c^T z,$$

subject to the above constraints.

Definition 4.2 (Partial Ordering of Vectors). Let $x = (x_1, \ldots, x_n), y = (y_1, \ldots, y_n) \in \mathbb{R}^n$. We write $x \geq y$ if and only if $x_i \geq y_i$ for all $1 \leq i \leq n$.

In general, a linear program is the maximization or minimization of a linear function, with some linear equality and/or inequality constraints on the variables. More formally, we begin with the standard definition of a linear program. The definition we give will begin with linear constraints, but inequality constraints can also be accommodated in a standard way.

Definition 4.3 (Linear Program, Standard Form). Let $c \in \mathbb{R}^n$, let $b \in \mathbb{R}^m$, and let A be a real $m \times n$ matrix. Let $x \in \mathbb{R}^n$ be a variable vector. Then a **linear program in standard form** is a minimization problem of the following form:

$$\text{minimize} \quad c^T x \quad \text{subject to the constraints}$$

$$Ax = b,\, x \geq 0.$$

The set $\{x \in \mathbb{R}^n \colon x \geq 0,\, Ax = b\}$ is called the **feasible set** of the linear program. If the feasible set is empty, the linear program is called **infeasible**. If there exist a sequence $x^{(1)}, x^{(2)}, \ldots$ of points in the feasible set such that $c^T x^{(i)} \to -\infty$ as $i \to \infty$, we say the linear program is **unbounded**.

If the linear program is infeasible or unbounded, then no minimum exists for the linear program.

Remark 4.4. Sometimes, a linear program can be written with inequality constraints, such as

$$\text{minimize} \quad c^T x \quad \text{subject to the constraints}$$

$$Ax \leq b,\, x \geq 0.$$

We can convert this to a linear program in standard form as follows

$$\text{minimize} \quad c^T x \quad \text{subject to the constraints}$$

$$Ax = b - y,\, x \geq 0,\, y \geq 0.$$

Here $y \in \mathbb{R}^m$. Note that $Ax = b - y$ for some $y \geq 0$ if and only if $Ax \leq b$. So,

$$\{x \in \mathbb{R}^n \colon \exists\, y \in \mathbb{R}^m,\, Ax = b - y,\, x \geq 0,\, y \geq 0\} = \{x \in \mathbb{R}^n \colon Ax \leq b,\, x \geq 0\}.$$

Therefore, these two linear programs are equivalent. That is, they have the same minimum value, and this minimum value is achieved at the same vectors $x \in \mathbb{R}^n$.

Note that the constraint $x \geq 0, y \geq 0$ is equivalent to requiring a single concatenated vector to satisfy $(x, y) \geq 0$. So, the second linear program is in standard form.

Here the variable y which is added to the constraint is called a **slack variable**.

Exercise 4.5. Let $c, x \in \mathbb{R}^n$, let $b \in \mathbb{R}^m$, and let A be an $m \times n$ real matrix.

It is possible to essentially put all of the variables of a linear program into the constraint. Show that the following linear program is equivalent to the standard one. (That is, the minimum/maximum values and the x achieving the minimum/maximum value is the same for both linear programs.)

$$\text{maximize} \quad t \quad \text{subject to the constraints}$$
$$\{t \in \mathbb{R} \colon t \leq c^T x,\ \forall\, x \in \mathbb{R}^n \text{ such that } Ax = b,\ x \geq 0\}.$$

Exercise 4.6. Let $c, x \in \mathbb{R}^n$, let $b \in \mathbb{R}^m$, and let A be an $m \times n$ real matrix. Show that the linear program

$$\text{minimize} \quad c^T x \quad \text{subject to the constraints}$$
$$Ax \leq b.$$

is equivalent to the following linear program in standard form

$$\text{minimize} \quad \begin{pmatrix} c \\ -c \\ 0 \end{pmatrix}^T \begin{pmatrix} x^+ \\ x^- \\ z \end{pmatrix} \quad \text{subject to the constraints}$$
$$\begin{pmatrix} A & -A & I \end{pmatrix} \begin{pmatrix} x^+ \\ x^- \\ z \end{pmatrix} = b, \qquad \begin{pmatrix} x^+ \\ x^- \\ z \end{pmatrix} \geq 0.$$

By equivalent, we mean that if $x \in \mathbb{R}^n$, and if we define $x^+ := \max(x, 0)$, $x^- := \max(-x, 0)$, then $x = x^+ - x^-$. Similarly, if $x^+, x^- \geq 0$, then we define $x = x^+ - x^-$. And the minimum values and the x achieving the minimum value for both linear programs is the same. (Here the maximum is defined component-wise, e.g. if $x = (-1, 2, 3)$, then $\max(x, 0) = (0, 2, 3)$.)

Linear programs can also be interpreted geometrically.

Definition 4.7 (Half Space). Let $z \in \mathbb{R}^n$ and let $t \in \mathbb{R}$. A (closed) **half space** is any set of the form

$$\{x \in \mathbb{R}^n \colon \langle x, z \rangle \geq t\}.$$

That is, a half space is a any set of points lying on one side of a hyperplane.

Definition 4.8 (Polytope). A **polytope** is a subset of $\mathbb{R}^n$ which is the intersection of a finite number of half spaces.

Remark 4.9. A polytope is a convex set.

Exercise 4.10. Is $\{(x_1, x_2) \in \mathbb{R}^2 \colon x_1^2 + x_2^2 \leq 1\}$ a polytope? Prove your assertion.

Definition 4.11 (Extreme Point/ Vertex). Let $K \subseteq \mathbb{R}^n$ be a convex set. Let $x \in K$. We say that x is an **extreme point** or a **vertex** of K if x is not contained in any open line segment in K. That is, for any $y, z \in K$ with $y \neq z$, we have

$$x \notin \{ty + (1 - t)z \colon t \in (0, 1)\}.$$

Exercise 4.12. Find all extreme points of the set $\{(x_1, x_2) \in \mathbb{R}^2 \colon 0 \leq x_1 \leq 1,\ 0 \leq x_2 \leq 1\}$. Then, find all extreme points of the set $\{(x_1, x_2) \in \mathbb{R}^2 \colon x_1^2 + x_2^2 \leq 1\}$.

Exercise 4.13. Prove that a polytope is convex. Then, draw the following polytopes in the plane:

$$\{(x_1,x_2)\in\mathbb{R}^2\colon x_1\geq 0,\, x_2\geq 0,\, x_1+x_2\leq 2,\, x_2\leq 1\}.$$
$$\{(x_1,x_2)\in\mathbb{R}^2\colon x_1\geq 0,\, x_2\geq 0,\, x_1+x_2\leq 1,\, x_1+2x_2=1\}.$$
$$\{(x_1,x_2)\in\mathbb{R}^2\colon x_1\geq 0,\, x_2\geq 0,\, x_1+x_2=1,\, x_1+2x_2=1\}.$$

Definition 4.14. Let $K\subseteq\mathbb{R}^n$. We say that K is **bounded** there exists some $r>0$ such that $\|x\|\leq r$ for all $x\in K$.

Note that a polytope may or may not be bounded.

Remark 4.15. A linear program in standard form is the minimization of a linear function on a polytope. To see this, note that the set $\{x\in\mathbb{R}^n\colon x\geq 0\}$ is the intersection of n half spaces. More specifically, if $e_1,\ldots,e_n\in\mathbb{R}^n$ are the standard basis vectors (so that for any $1\leq i\leq n$, e_i has a 1 in the i^{th} coordinate, and all other coordinates are zero),

$$\{x\in\mathbb{R}^n\colon x\geq 0\}=\bigcap_{i=1}^n\{x\in\mathbb{R}^n\colon \langle x,e_i\rangle\geq 0\}.$$

Also, if $a^{(i)}$ denotes the i^{th} row of an $m\times n$ real matrix A for any $1\leq i\leq m$, then

$$\{x\in\mathbb{R}^n\colon Ax=b\}=\left(\bigcap_{i=1}^m\{x\in\mathbb{R}^n\colon \langle a^{(i)},x\rangle\geq b_i\}\right)\bigcap\left(\bigcap_{i=1}^m\{x\in\mathbb{R}^n\colon \langle -a^{(i)},x\rangle\geq -b_i\}\right).$$

So, the set $\{x\in\mathbb{R}^n\colon x\geq 0,\, Ax=b\}$ is a polytope, since

$$\{x\in\mathbb{R}^n\colon x\geq 0,\, Ax=b\}=\left(\bigcap_{i=1}^n\{x\in\mathbb{R}^n\colon x_i\geq 0\}\right)$$
$$\bigcap\left(\bigcap_{i=1}^m\{x\in\mathbb{R}^n\colon \langle a^{(i)},x\rangle\geq b_i\}\right)\bigcap\left(\bigcap_{i=1}^m\{x\in\mathbb{R}^n\colon \langle -a^{(i)},x\rangle\geq -b_i\}\right).$$

Remark 4.16. At first glance, a linear program may not seem difficult to solve, since we are simply optimizing a linear function on a convex set. If a solution to the problem exists, then we should be able to find it. However, the difficulty of a solving a linear program arises from the feasible set. For example, if there are 100 constraints, it seems essentially impossible to satisfy all of these constraint by hand, or even by computer.

4.2. **The Simplex Method.**

Exercise 4.17. Let $K\subseteq\mathbb{R}^n$ be a polytope formed by the intersection of $m>n$ half spaces. Assume that K is nonempty and bounded. Show that K has at most $\binom{m}{n}=\frac{m!}{(m-n)!n!}$ vertices.

(Hint: first show that a vertex of the polytope must be in the boundary of at least n half spaces, using linear algebra. That is, for any vertex of K, there exist at least n of the half spaces that define K such that equality occurs in the definition of that half space.)

Exercise 4.18. Let $K\subseteq\mathbb{R}^n$ be a polytope. Let $f\colon K\to\mathbb{R}$ be a concave function (so that $-f$ is convex). Assume that a minimum value of f exists. That is, there exists $x\in K$ such that $f(x)\leq f(k)$ for all $k\in K$. Conclude that there exists an extreme point $y\in K$ such that f attains its minimum value at y.

Exercise 4.19. Let K be a bounded polytope and let $f\colon K \to \mathbb{R}$ be a linear function. Show that the minimum value of f is attained at a vertex of K. (Hint: a linear function is concave. Also, from Exercise 4.17, there are only finitely many vertices of K. Using the definition of a vertex, show that: for any point $k \in K$, there exists a vertex $x \in K$ where $f(x) \leq f(k)$.)

The Simplex Method is based upon Exercises 4.19 and 4.17. As shown in Remark 4.15, a linear program in standard form is the minimization of a linear function on a polytope K. If K is a nonempty and bounded polytope, then the minimum of the linear program exists by Exercise 4.19. In order to minimize the linear function on K, the simplex method moves through the vertices of the polytope, eventually finding the minimizing vertex. By Exercise 4.17, there are only finitely many vertices of the polytope. So, eventually, we will find the vertex minimizing the given linear function.

Though there are sensible ways of choosing how to move between the vertices of the feasible set, unfortunately the simplex method may require checking the value of all vertices of the polytope. We discuss this issue further below, since it demonstrates that the simplex method is highly inefficient in the worst-case.

Below we will assume that the feasible set $Ax = b$, $x \geq 0$ has an $m \times n$ matrix A with full row rank. In practice, this assumption is reasonable. If A does not have full row rank, then the constraint $Ax = b$ has redundancy, which can be removed by row operations, resulting in a matrix with less rows and with full row rank.

Definition 4.20 (**Basic Feasible Solution**). Let m, n be positive integers with $m \leq n$. Let A be a real $m \times n$ matrix with full row rank, and let $b \in \mathbb{R}^m$. For any $1 \leq i \leq n$, let A_i denote the i^{th} column of A. A vector $x \in \mathbb{R}^n$ is a **basic feasible solution** if the following conditions are satisfied:

- $Ax = b$ and $x \geq 0$.
- There is a subset $S \subseteq \{1, \ldots, n\}$ such that S has size m. We write $S = \{s_1, \ldots, s_m\}$.
- If $i \in \{1, \ldots, n\}$ satisfies $i \notin S$, then $x_i = 0$.
- The $m \times m$ matrix $B := (A_{s_1}, \ldots, A_{s_m})$ is an invertible matrix.

The following Lemma shows that basic feasible solutions and vertices are the same.

Lemma 4.21. *Let m, n be positive integers with $m \leq n$. Let A be a real $m \times n$ matrix with full row rank, and let $b \in \mathbb{R}^m$. Let $K = \{w \in \mathbb{R}^n \colon w \geq 0,\ Aw = b\}$. Then x is a vertex of K if and only if x is a basic feasible solution.*

Proof. Let $x \in \mathbb{R}^n$ be a basic feasible solution. Let $S \subseteq \{1, \ldots, n\}$ have size m so that $x_i = 0$ for any $i \notin S$. Without loss of generality, $S = \{1, \ldots, m\}$. Let $0 < t < 1$ and let $y, z \in K$. Assume that $x = ty + (1-t)z$. Since $t > 0$, $(1-t) > 0$, $y \geq 0$ and $z \geq 0$, we must have $y_i = z_i = 0$ for any $i \notin S$. Let $x' := (x_1, \ldots, x_m)^T$, and similarly define y', z'. We can therefore rewrite $Ax = Ay = Az = b$ as $Bx' = By' = Bz'$. Since B is invertible, we conclude that $x' = y' = z'$, so that $x = y = z$. That is, x is a vertex of K.

We now prove the converse. Let $x \in \mathbb{R}^n$ be a vertex of K. Without loss of generality, assume $x_1, \ldots, x_p$ are the only nonzero components of x, where $p \geq 1$. If the columns $A_1, \ldots, A_p$ of A are linearly dependent, then without loss of generality, there exist scalars $t_1, \ldots, t_{p-1}$ such that $-A_p + \sum_{i=1}^{p-1} t_i A_i = 0$. Let $\varepsilon \in (-1, 1)$ and note that the vector

$$y(\varepsilon) := x + \varepsilon(t_1, \ldots, t_{p-1}, -1, 0, \ldots, 0)^T$$

satisfies $Ay(\varepsilon) = b$. Since $x_1, \ldots, x_p > 0$, there exists $\delta > 0$ such that, if $|\varepsilon| < \delta$, then $y \geq 0$. In particular, $y(\varepsilon)$ and $y(-\varepsilon)$ are both in the feasible set K. And $x = \frac{1}{2}(y(\varepsilon) + y(-\varepsilon))$. Since this equality contradicts that x is a vertex of K, we conclude that the columns $A_1, \ldots, A_p$ are linearly independent. Since A is an $m \times n$ matrix with $m \leq n$, we conclude that $p \leq m$. If $p = m$, then x is a basic feasible solution with $S = \{1, \ldots, m\}$. If $p < m$, then since A has full row rank m, A also has column rank m, so we can add more indices to the set $\{1, \ldots, p\}$ to form a set S of size m such that $\{A_i\}_{i \in S}$ form a basis of $\mathbb{R}^m$. In any case, x is a basic feasible solution. $\square$

Definition 4.22 (**Degeneracy**). Let m, n be positive integers with $m \leq n$. Let A be a real $m \times n$ matrix with full row rank, and let $b \in \mathbb{R}^m$. A linear program with feasible region $\{w \in \mathbb{R}^n \colon w \geq 0,\ Aw = b\}$ is said to be **degenerate** if there exists a basic feasible solution $x \in \mathbb{R}^n$ with corresponding subset $S \subseteq \{1 \ldots, n\}$ of size m, and there exists $i \in S$ with $x_i = 0$. If the linear program is not degenerate, we say it is **non-degenerate**.

As mentioned above, the simplex method amounts to moving between vertices of the feasible set. Put another way, the simplex method involves moving from one basic feasible solution to another. We now describe how to make such a move, assuming that the linear program is non-degenerate.

Suppose we have one basic feasible solution x with corresponding set $S = \{s_1, \ldots, s_m\} \subseteq \{1, \ldots, n\}$. As above, let $B := (A_{s_1}, \ldots, A_{s_m})$. In linear algebra terminology, B is a change of basis matrix. Let $i \in \{1, \ldots, n\}$ with $i \notin S$. Let $S' := (S \cup \{i\}) \smallsetminus \{s_1\}$. Then S' also has size m, and S' is obtained by replacing one index in S with another index not in S.

Write $A_i = \sum_{j=1}^{m} r_{ij} A_{s_j}$ for some real numbers $(r_{i1}, \ldots, r_{im})^T =: r^{(i)}$. These real numbers exist since $A_{s_1}, \ldots, A_{s_m}$ is a basis of $\mathbb{R}^m$. (Note that $Br^{(i)} = A_i$ so $r^{(i)} = B^{-1} A_i$.) Since $\sum_{j=1}^{m} x_{s_j} A_{s_j} = Ax = b$, if $\varepsilon \in \mathbb{R}$, we add two equations to get

$$\varepsilon A_i + \sum_{j=1}^{m} (x_{s_j} - \varepsilon r_{ij}) A_{s_j} = b.$$

For any $1 \leq i \leq n$, let $e_i \in \mathbb{R}^n$ denote the vector with a 1 in the i^{th} entry and a 0 in all other entries. Then, the vector

$$z(\varepsilon) := \varepsilon e_i + \sum_{j=1}^{m} (x_{s_j} - \varepsilon r_{ij}) e_{s_j}.$$

satisfies $Az(\varepsilon) = b$.

Since the linear program is not degenerate and x is a basic feasible solution, there exists $\delta > 0$ such that, for all $|\varepsilon| < \delta$, $x_{s_j} - \varepsilon r_{ij} > 0$. That is, if $|\varepsilon| < \delta$, then $z(\varepsilon)$ is in the feasible set. We then increase ε until there exists some $1 \leq k \leq m$ such that $x_{s_k} - \varepsilon r_{ik} = 0$. (Such an ε may not exist, but for now suppose it does.) For such an ε, we have $Az(\varepsilon) = b$, $z(\varepsilon) \geq 0$. So, we might hope that $(S \smallsetminus \{k\}) \cup \{i\}$ and $z(\varepsilon)$ are another basic feasible solution, where we can compare $c^T z(\varepsilon)$ to $c^T x$. We summarize the above discussion.

Algorithm 4.23 (**Simplex Method/ Simplex Algorithm**). Let m, n be positive integers with $m \leq n$. Let A be a real $m \times n$ matrix with full row rank, and let $b \in \mathbb{R}^m$. The Simplex Method solves the linear program in standard form

$$\text{minimize} \quad c^T x \quad \text{subject to the constraints}$$

$$Ax = b,\ x \geq 0.$$

We assume that this linear program is non-degenerate.

We are given a basic feasible solution y in the feasible region with associated set $S = \{s_1, \ldots, s_m\} \subseteq \{1, \ldots, n\}$ of size m and invertible matrix $B = (A_{s_1}, \ldots, A_{s_m})$.

- Let y be a basic feasible solution with corresponding set $S = \{s_1, \ldots, s_m\} \subseteq \{1, \ldots, n\}$. Let $T := \{1, \ldots, n\} \smallsetminus S$.
- Let $i \in \{1, \ldots, n\}$ with $i \in T$. (If $T = \emptyset$, terminate the algorithm.) Let $r^{(i)} \in \mathbb{R}^m$ so that $r^{(i)} := B^{-1}A_i$. (Alternatively, let $r^{(i)} = (r_{i1}, \ldots, r_{im})^T$ solve $Br^{(i)} = A_i$, using e.g. the Conjugate Gradient Method of Algorithm 3.13.)
- Define $\varepsilon := \min\{y_{s_j}/r_{ij} \colon j = 1, \ldots, m,\ r_{ij} > 0\}$, and let k so that $y_{s_k}/r_{ik} = \varepsilon$. Define
$$z := \varepsilon e_i + \sum_{j=1}^{m} (y_{s_j} - \varepsilon r_{ij}) e_{s_j}.$$
(If no such ε exists, terminate the algorithm. The feasible set is unbounded.)
- If $c^T y > c^T z$, return to the first step, redefining y to be z, and redefining S to be $(S \cup \{i\}) \smallsetminus \{s_k\}$. If on the other hand $c^T y \leq c^T z$, return to the first step, keep y as it is, and redefine T to be $T \smallsetminus \{i\}$.

Theorem 4.24. *Suppose we have a linear program whose feasible set is nonempty and bounded. Then one step of the simplex algorithm moves from one basic feasible solution to another. After a finite number of steps, the algorithm terminates at the minimum value of the linear program.*

Proof. First, if $r_{ij} \leq 0$ for all $1 \leq j \leq m$, then ε is undefined. That is, for every $\varepsilon' > 0$, the vector
$$z := \varepsilon' e_i + \sum_{j=1}^{m} (y_{s_j} - \varepsilon' r_{ij}) e_{s_j}$$
satisfies $Az = b$. Letting $\varepsilon' \to \infty$ shows that the feasible region is unbounded.

So, if the feasible region is nonempty and bounded, we must have $r_{ij} > 0$ for some $1 \leq j \leq m$, so that ε is well-defined.

By definition of z, $Az = b$ and z has at most $m + 1$ nonzero entries. Since $y \geq 0$, the definition of ε implies that $z \geq 0$ and z has at most m nonzero entries. Let $k \in \{1, \ldots, m\}$ be the unique integer such that $y_{s_k}/r_{ik} = \min\{y_{s_j}/r_{ij} \colon j = 1, \ldots, m,\ r_{ij} > 0\}$. (The uniqueness of k follows since the linear program is assumed to be non-degenerate.) We claim that $\{A_j\}_{j \in (S \cup \{i\}) \smallsetminus \{s_k\}}$ is a linearly independent set of vectors. Recall that $\{A_j\}_{j \in S}$ is a linearly independent set. Also,
$$A_i = \sum_{j=1}^{m} r_{ij} A_{s_j} = r_{ik} A_{s_k} + \sum_{j \neq k} r_{ij} A_{s_j}.$$
Since $r_{ik} > 0$, we have
$$A_{s_k} = \frac{A_i - \sum_{j \neq k} r_{ij} A_{s_j}}{r_{ik}}.$$
Since $\{A_{s_1}, \ldots, A_{s_m}\}$ is a basis, and any vector in its span can be written as a linear combination of $\{A_j\}_{j \in (S \cup \{i\}) \smallsetminus \{s_k\}}$, we conclude that $\{A_j\}_{j \in (S \cup \{i\}) \smallsetminus \{s_k\}}$ is also a basis.

In summary, $\{A_j\}_{j\in(S\cup\{i\})\setminus\{s_k\}}$ and $z\geq 0$ has at most m nonzero entries. Since the linear program is not degenerate, z has exactly m nonzero entries, and z is a basic feasible solution.

If the fourth step of the algorithm occurs, then we have moved from one vertex of the feasible region to another, while decreasing the value of the linear program. From Exercises 4.19 and 4.17, the minimum value of the program is attained at a vertex of the feasible region, and there are only finitely many vertices to visit. By the definition of the fourth step of the algorithm, the algorithm never returns to a previously visited vertex. So, eventually, we will reach the second step of the algorithm with $T=\emptyset$. That is, we have found a basic feasible solution x and compared it to other possible basic feasible solutions, all of which have a larger or equal value than x in the linear program.

Without loss of generality, x is a basic feasible solution with $S=\{1,\ldots,m\}$. By assumption, for any $\varepsilon>0$, and for any $m+1\leq i\leq n$

$$c^Tz(\varepsilon)-c^Tx=c^Tz(\varepsilon)-c^Tz(0)=\varepsilon c^T(e_i-\sum_{j=1}^m r_{ij}e_j)=\varepsilon\left(c_i-\sum_{j=1}^m r_{ij}c_j\right)\geq 0. \qquad (*)$$

We now show that x is the global minimum of the linear program. Let $y\in\mathbb{R}^n$ satisfy the equation $Ay=b$. Applying B^{-1} to both sides, we get $(I,B^{-1}A')y=B^{-1}b$, where I is the $m\times m$ identity matrix, and $A':=(A_{m+1},\ldots,A_n)$. So, if we write $y=(w,y')^T$ where $w\in\mathbb{R}^m$ and $y'\in\mathbb{R}^{n-m}$, then we have $w+B^{-1}A'y'=B^{-1}b$, so $w=B^{-1}(b-A'y')$. Since $Ax=b$, we similarly have $(I,0)x=(I,B^{-1}A')x=B^{-1}b$. In summary, any solution to the equation $Ay=b$ is of the form

$$y=(I,0)x+\begin{pmatrix}-B^{-1}A'y'\\ y'\end{pmatrix},\qquad y'\in\mathbb{R}^{n-m}.$$

Also, using the notation above, $r_j=(B^{-1}A')_{j(i-m)}$ for any $m+1\leq i\leq n$, $1\leq j\leq m$. So

$$\begin{aligned}c^Ty-c^Tx&=-\sum_{j=1}^m c_j(B^{-1}A'y')_j+\sum_{i=m+1}^n y'_{i-m}c_i\\ &=-\sum_{i=m+1}^n\sum_{j=1}^m c_j(B^{-1}A')_{j(i-m)}y'_{i-m}+\sum_{i=m+1}^n y'_{i-m}c_i=\sum_{i=m+1}^n y'_{i-m}\left(c_i-\sum_{j=1}^m c_jr_{ij}\right).\end{aligned}$$

If y is in the feasible set, then $Ay=b$ and $y\geq 0$, so that $y'\geq 0$. By $(*)$, we therefore conclude that $c^Ty-c^Tx\geq 0$. That is, x is the global minimum of the linear program, as desired. □

Exercise 4.25. Using the Simplex Algorithm, solve the following linear program:

$$\text{minimize}\quad -4x_1-2x_2\quad\text{subject to the constraints}$$

$$x_1+x_2+x_3=5$$

$$2x_1+x_2/2+x_4=8,\qquad x\geq 0.$$

(Hint: start at the point $(x_1,x_2,x_3,x_4)=(0,0,5,8)$)

4.3. **The Ellipsoid Method.**

The strategy of the simplex method can be summarized as: move through the vertices of the feasible region. Unfortunately, the simplex method has several problems.

First, in general, how can we start with a basic feasible solution? For some problems it is easy, but for others it may not be so easy. In fact, finding a basic feasible solution for a general problem is equivalent to finding the minimum value of a general linear program!

Proposition 4.26 (**Equivalence of Separation and Optimization**). *Let $P \subseteq \mathbb{R}^n$ be a polytope formed by the intersection of m half spaces. Suppose there is an algorithm that, in a time polynomial in n and m, can either find some $x \in P$ or state that P is empty. Suppose we have a linear program in standard form whose minimum value v is in $[0,1]$. Then for any $\varepsilon > 0$, there is an algorithm that finds x in the feasible region of the linear program in time polynomial in n, m and $\log(1/\varepsilon)$. Moreover, $c^T x \leq v + \varepsilon$. That is, x is ε close to minimizing the linear program.*

Proof. From Exercise 4.5, any linear program in standard form can be equivalently written as: maximize $t \in \mathbb{R}$ subject to the constraint: $\{t \in \mathbb{R} \colon t \leq c^T x, \forall x \in \mathbb{R}^n$ such that $Ax = b, x \geq 0\}$. For any $t \in [0,1]$, define

$$R(t) := \{x \in \mathbb{R}^n \colon Ax = b,\, x \geq 0,\, t \geq c^T x\}. \qquad (*)$$

We now perform a binary search in t. Let $t_1 := 1/2$. If $R(t_1) = \emptyset$, we let $t_2 := 3/4$ and then check if $R(t_2)$ is empty or not. If $R(t_1) \neq \emptyset$, we then let $t_2 := 1/4$ and check if $R(1/4)$ is empty or not. In general, at the n^{th} step of this procedure, $2^{-n} \leq t_n \leq 1 - 2^{-n}$ is given. If $R(t_n) = \emptyset$, we then let $t_{n+1} := t_n + 2^{-n-1}$. If $R(t_n) \neq \emptyset$, we then let $t_{n+1} := t_n - 2^{-n-1}$. We continue this procedure a total of $N := 10\log(1/\varepsilon)$ steps. Each step requires time polynomial in n and m. So, the total run time is a polynomial in n, m and $\log(1/\varepsilon)$. Also, $R(t_N - \varepsilon/2) = \emptyset$ and $R(t_N + \varepsilon/2) \neq \emptyset$. So, if we let $x \in R(t_N + \varepsilon/2)$, then by definition $(*)$ we have $c^T x \leq t_N + \varepsilon/2$, and $v \geq t_N - \varepsilon/2$. So, $c^T x \leq v + \varepsilon$. □

The second and more significant issue with the Simplex Method is that it may take an exponentially long time to terminate!

Exercise 4.27. Let $H_n := \{(x_1, \ldots, x_n) \in \mathbb{R}^n \colon 0 \leq x_i \leq 1, \forall\, 1 \leq i \leq n\}$. The set H_n is the n-dimensional cube. First, show that H_n is a polytope which is formed by the intersection of $2n$ half spaces. Then, show that H_n has 2^n vertices.

In summary, H_n is described by $2n$ inequalities on n variables, but H_n has 2^n vertices. If we perturb the half spaces defining H_n slightly, and if we e.g. try to maximize x_n, then it is possible that the simplex method will visit all 2^n vertices of the perturbed cube. That is, the simplex method may need an exponential number of steps to terminate, even though the feasible region is described by a linear number of inequalities and variables.

Thankfully, there are algorithms that are guaranteed to solve linear programming problems in a polynomial amount of time in the number of inequalities and variables of the problem. One such algorithm is the Ellipsoid method. This algorithm has nice theoretical properties but it is not often used in practice. As far as I know, interior point methods (which we describe in the next section) are typically implemented to solve linear programming problems. Interior point methods behave well in practice and in theory.

We now describe the Ellipsoid method. From Proposition 4.26, in order to find the minimum of a linear program, it suffices to find an algorithm that can check whether or not a given feasible region is empty.

Definition 4.28. The **Euclidean ball of radius** $r > 0$ **and center** $y \in \mathbb{R}^n$ is the following set

$$B_r(y) := \{x \in \mathbb{R}^n \colon \ \|x - y\| \leq r\}.$$

In particular, $B_1(0) \subseteq \mathbb{R}^n$ is the unit ball centered at the origin. For any $C \subseteq \mathbb{R}^n$, let $\mathrm{vol}_n(C)$ denote the volume of C:

$$\mathrm{vol}_n(C) := \int_C dx.$$

Let A be a symmetric positive definite $n \times n$ matrix and let $y \in \mathbb{R}^n$. Define an **ellipsoid** to be any set of the form

$$\begin{aligned} E = E(A, y) &:= \{x \in \mathbb{R}^n \colon (x - y)^T A^{-1}(x - y) \leq 1\} \\ &= \{x \in \mathbb{R}^n \colon \|A^{-1/2}(x - y)\|^2 \leq 1\} = \{A^{1/2}x + y \in \mathbb{R}^n \colon \|x\|^2 \leq 1\} = A^{1/2}B_1(0) + y. \end{aligned}$$

Equivalently, an ellipsoid is the invertible linear image of the unit ball $B_1(0)$.

From the last equality and the change of variables formula

$$\mathrm{vol}_n(E(A, y)) = \sqrt{\det(A)} \cdot \mathrm{vol}_n(B_1(0)).$$

Exercise 4.29. Let $C \subseteq \mathbb{R}^n$. Let A be a positive semidefinite matrix. Show that

$$\mathrm{vol}_n(AC) = \mathrm{vol}_n\{Ax \in \mathbb{R}^n \colon x \in C\} = \det(A)\mathrm{vol}_n(C).$$

Remark 4.30. Recall that if A is a positive semidefinite matrix, then the Spectral Theorem, Theorem 2.22, says there exists an orthogonal matrix Q and a diagonal matrix D with nonnegative entries such that $A = QDQ^{-1}$. We therefore define $A^{1/2} = QD^{1/2}Q^{-1}$. Note that $(A^{1/2})^2 = A$.

In the ellipsoid method, we will be given an arbitrary ellipsoid, cut in half by a half space which passes through the ellipsoid's center. We then wish to find another ellipsoid of small volume that encloses the resulting half-ellipsoid. It turns out one can find explicit formulas to do this.

Exercise 4.31. Let A be a positive definite $n \times n$ matrix. Let $y \in \mathbb{R}^n$ and define the ellipsoid $E(A, y) := \{x \in \mathbb{R}^n \colon (x - y)^T A^{-1}(x - y) \leq 1\}$. Let $z \in \mathbb{R}^n$, $z \neq 0$, and define the half-ellipsoid

$$E'(A, y, z) := E(A, y) \cap \{x \in \mathbb{R}^n \colon \langle z, (x - y)\rangle \leq 0\}.$$

(Note that y is the center of $E(A, y)$, and y lies on the boundary of $E'(A, y, z)$.) Define

$$d := \frac{1}{\sqrt{z^T A z}} Az.$$

$$y' := y - \frac{1}{n+1} d.$$

$$A' := \frac{n^2}{n^2 - 1}\left(A - \frac{2}{n+1} dd^T\right).$$

Show that $E'(A, y, z) \subseteq E(A', y')$. The set $E(A', y')$ is called the **Löwner-John ellipsoid** of $E'(A, y, z)$. Justify why is A' positive definite.

We are now in a position to describe the Ellipsoid method.

Theorem 4.32. ***(The Basic Central-Cut Ellipsoid Method)*** *Suppose $P \subseteq \mathbb{R}^n$ is a nonempty polytope that contains a Euclidean ball of any center and radius. Let A be an $m \times n$ real matrix and let $b \in \mathbb{R}^m$. We then write P as*

$$P := \{x \in \mathbb{R}^n \colon \langle a^{(i)}, x\rangle \leq b_i,\, \forall\, 1 \leq i \leq m\} = \{x \in \mathbb{R}^n \colon Ax \leq b\}.$$

Suppose also that we begin with some Euclidean ball B such that $P \subseteq B$ and the volume ratio $\mathrm{vol}_n(B)/\mathrm{vol}_n(P)$ is bounded above by some absolute constant $c' \geq 1$. Then there exists an algorithm that explicitly finds a point $x \in P$, and such that the time to run this algorithm is a polynomial expression in n, m, and $\log(c')$.

Proof. The algorithm uses a divide and conquer strategy.

Begin with the ellipsoid $E_0 := B$, and let us then define an iterative procedure. Suppose B has center $y_0 \in \mathbb{R}^n$, and let $A_0 := I$ be the $n \times n$ identity matrix. Let $k \geq 0$ be an integer. We define the k^{th} step of the algorithm. We are given $E_k = E(A_k, y_k)$, and we inductively assume that $P \subseteq E_k$. We then check whether or not $y_k \in P$ (i.e. check if $Ay_k \leq b$). If $y_k \in P$, then the algorithm terminates, since we have found a point in P. If $y_k \notin P$, then suppose the j^{th} inequality of $Ax \leq b$ is violated, i.e. $\langle a^{(j)}, y_k\rangle > b_j$ for some $1 \leq j \leq m$. By definition of P, $P \cap \{x \in \mathbb{R}^n \colon \langle a^{(j)}, x\rangle > b_j\} = \emptyset$. Since $P \subseteq E_k$, we conclude that

$$\begin{aligned} P &\subseteq E_k \cap \{x \in \mathbb{R}^n \colon \langle a^{(j)}, x\rangle \leq b_j\} \subseteq E_k \cap \{x \in \mathbb{R}^n \colon \langle a^{(j)}, x\rangle \leq \langle a^{(j)}, y_k\rangle\} \\ &= E_k \cap \{x \in \mathbb{R}^n \colon \langle a^{(j)}, x - y_k\rangle \leq 0\} \ = E'(A_k, y_k, a^{(j)}). \end{aligned}$$

Finally, define E_{k+1} to be the Löwner-John ellipsoid of $E'(A_k, y_k, a^{(j)})$ from Exercise 4.31, so that $P \subseteq E_{k+1}$. The k^{th} step of the algorithm is now complete. A calculation shows

$$\frac{\mathrm{vol}_n(E_{k+1})}{\mathrm{vol}_n(E_k)} < e^{-1/2n} < 1, \quad \forall\, k \geq 0. \qquad (*)$$

So, suppose we perform the above procedure with $N > 2n\log(c)$ iterations where N is a positive integer. Then

$$\frac{\mathrm{vol}_n(E_N)}{\mathrm{vol}_n(B)} = \frac{\mathrm{vol}_n(E_N)}{\mathrm{vol}_n(E_0)} = \prod_{k=0}^{N-1} \frac{\mathrm{vol}_n(E_{k+1})}{\mathrm{vol}_n(E_k)} \overset{(*)}{<} (e^{-1/2n})^{2n\log(c')} = (c')^{-1}.$$

That is, $\mathrm{vol}_n(E_N) < \mathrm{vol}_n(B)(c')^{-1}$. On the other hand, $\mathrm{vol}_n(B)/\mathrm{vol}_n(P) \leq c'$, so $\mathrm{vol}_n(P) \geq (c')^{-1}\mathrm{vol}_n(B)$. That is, $\mathrm{vol}_n(E_N) < \mathrm{vol}_n(P)$. So, it is not possible that $P \subseteq E_N$. That is, the algorithm terminates after at most N steps. At termination, we must have found a point in P, as desired.

To justify the calculation $(*)$, note that the change of variables formula implies that the volume ratio $\frac{\mathrm{vol}_n(E_{k+1})}{\mathrm{vol}_n(E_k)}$ is unchanged if we apply the same invertible linear transformation to E_k and to E_{k+1}. So, for the purposes of the computation $(*)$, we may assume that $E_k = B_1(0)$, $A = I$ and $z = (-1, 0, \ldots, 0)$. Then we have from Exercise 4.31,

$$y' = \left(\frac{1}{n+1}, 0, \ldots, 0\right), \qquad A' = \begin{pmatrix} \frac{n^2}{(n+1)^2} & 0 & \cdots & 0 \\ 0 & \frac{n^2}{n^2-1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{n^2}{n^2-1} \end{pmatrix}.$$

which leads to

$$\frac{\mathrm{vol}_n(E_{k+1})}{\mathrm{vol}_n(E_k)} = \sqrt{\det(A')} = \left(\frac{n^{2n}}{(n+1)^n(n-1)^n}\frac{n-1}{n+1}\right)^{\frac{1}{2}} = \left(\left(\frac{n}{n+1}\right)^{n+1}\left(\frac{n}{n-1}\right)^{n-1}\right)^{\frac{1}{2}}.$$

Then Exercise 4.33 completes the proof. □

Exercise 4.33. By taking logarithms, show that for any positive integer n,

$$\left(\frac{n}{n+1}\right)^{n+1}\left(\frac{n}{n-1}\right)^{n-1} \leq e^{-1/n}.$$

(Hint:)

$$\begin{aligned}
&(n+1)\log(1+1/n) + (n-1)\log(1-1/n) \\
&= \sum_{k=1}^{\infty}(-1)^{k+1}(n+1)n^{-k}k^{-1} - \sum_{k=1}^{\infty}(n-1)n^{-k}k^{-1} \\
&= \sum_{k=1}^{\infty}(-1)^{k+1}2n^{-k}k^{-1} + \sum_{k=1}^{\infty}(-1)^{k+1}(n-1)n^{-k}k^{-1} - \sum_{k=1}^{\infty}(n-1)n^{-k}k^{-1} = \cdots
\end{aligned}$$

Remark 4.34. Strictly speaking, Theorem 4.32 and Proposition 4.26 do not show that every linear program can be solved in a polynomial amount of time. For example, if P is very small, then c' in Theorem 4.32 may need to be very large, in which case the Ellipsoid Method would run for a long time. However, once $\varepsilon > 0$ is fixed, if we want to minimize the linear program within an additive error of ε, then we can adjust the procedure in Proposition 4.26 as follows. We can select the values of t_n to instead be $t_n \pm \varepsilon$, thereby increasing the volume of $R(t_n \pm \varepsilon)$ in order to ensure that $c' > c''2^{-n^2}$ in Theorem 4.32. We omit the details of this argument, and we will not discuss this issue further.

The following combinatorial problem is a well-known application of linear programming.

Example 4.35 (Minimum Vertex Cover). Suppose we have a set of vertices $V := \{1,\ldots,n\}$ and a set of undirected edges $E \subseteq \{\{i,j\}: i,j \in V\}$. The goal of the minimum vertex cover problem is to find the smallest subset $S \subseteq V$ such that every $\{i,j\} \in E$ satisfies $i \in S$ or $j \in S$. More generally, for any $i \in V$, let $c_i \in \mathbb{R}$, $c_i \geq 0$. We are asked to minimize

$$\sum_{i\in S} c_i$$

over all $S \subseteq V$ such that every $\{i,j\} \in E$ satisfies $i \in S$ or $j \in S$. For a somewhat contrived example, we could think of the vertices as cities, and the set S as a subset of cities where cell phone towers are placed. And each cell phone tower is designed to cover the city in which it resides, and any adjacent cities.

This problem is known to be NP-complete. That is, if we could solve this problem in time polynomial in n, then $P = NP$ and we would solve one of the Millennium Prize Problems. Since it is widely believed that $P \neq NP$, it is doubtful that the Minimum Vertex Cover problem can be solved in time polynomial in n.

However, we can *approximately* solve this problem in polynomial time, using linear programming. In order to do this, we first rephrase the problem. Let $c := (c_1,\ldots,c_n)^T$. The

Minimum Vertex Cover problem is:

$$\text{minimize} \quad c^T x \quad \text{subject to the constraints}$$

$$x_i \in \{0,1\}, \quad \forall\, i \in V$$

$$x_i + x_j \geq 1, \quad \forall\, \{i,j\} \in E$$

This optimization problem is not a linear program, since the first constraint is not a linear function of the variables. However, there is a natural way to create a linear program from this optimization problem, by **relaxing** the first constraint. That is, we replace the constraint $x_i \in \{0,1\}\ \forall\, i \in V$ with another less stringent constraint, namely $0 \leq x_i \leq 1\ \forall\, i \in V$. Since any x satisfying the previous constraint also satisfies the new constraint, we say that the new constraint is a **relaxation** of the previous constraint. Crucially, the new constraint is linear, so that the following problem is a linear programming problem:

$$\text{minimize} \quad c^T x \quad \text{subject to the constraints}$$

$$0 \leq x_i \leq 1, \quad \forall\, i \in V$$

$$x_i + x_j \geq 1, \quad \forall\, \{i,j\} \in E$$

Theorem 4.36. *Let $\widetilde{x}$ minimize the above linear program. Let x minimize the minimum vertex cover problem. Then*

$$c^T \widetilde{x} \leq c^T x \leq 2 c^T \widetilde{x}.$$

That is, $c^T \widetilde{x}$ approximates the minimum value of the minimum vertex cover problem within a multiplicative factor of 2. And $c^T \widetilde{x}$ can be computed in time polynomial in n.

*We therefore say that the above linear program is a **factor 2 approximation algorithm** for the minimum vertex cover problem.*

Proof. First, note that x exists since there are only a finite number of vectors that are feasible for the minimum vertex cover problem. Also, note that the linear program above is bounded and feasible.

Since x is feasible for the minimum vertex cover problem, x is also feasible for the associated linear program. So, by definition of $\widetilde{x}$, we have $c^T \widetilde{x} \leq c^T x$. We now prove the other inequality.

We know that each entry of $\widetilde{x}$ is in the interval $[0,1]$, but we would like to somehow associate the vector $\widetilde{x}$ to a vector whose entries are either 0 or 1. To do this, we "round" the entries of $\widetilde{x}$ to either 0 or 1, whichever is closest. That is, define $y \in \mathbb{R}^n$ so that, for any $1 \leq i \leq n$,

$$y_i := \begin{cases} 0 & \text{, if } \widetilde{x}_i < 1/2 \\ 1 & \text{, if } \widetilde{x}_i \geq 1/2. \end{cases}.$$

We claim that y is feasible for the minimum vertex cover problem. To see this, let $\{i,j\} \in E$. Since $\widetilde{x}$ is feasible for the linear program, $\widetilde{x}_i + \widetilde{x}_j \geq 1$. That is, $\max(\widetilde{x}_i, \widetilde{x}_j) \geq 1/2$. Therefore, by definition of y, $y_i + y_j \geq 1$. In conclusion, y is feasible for the minimum vertex cover problem. Therefore, $c^T x \leq c^T y$.

Now, by the definition of y, we have $y_i \leq 2\widetilde{x}_i$ for all $1 \leq i \leq n$. Since $c \geq 0$, we conclude

$$c^T y \leq 2 c^T \widetilde{x}.$$

□

Remark 4.37. The above approximation algorithm for the minimum vertex cover problem is relatively simple, but it is conjecturally the best that can be done! That is, it is conjectured that, for every $\varepsilon > 0$, it is impossible to create a $2 - \varepsilon$ approximation algorithm for the minimum vertex cover problem that runs in polynomial time.

Exercise 4.38. Using any method you want to use, minimize

$$x_2 - x_1$$

subject to the constraints

$$x_1 \geq 0,\ x_2 \geq 0,\ x_1 + x_2 \leq 2,\ x_2 \leq 1.$$

(Note that only drawing a picture is insufficient justification.)

4.4. **Interior Point Methods.** The simplex method moves between the vertices of the feasible region in the direction of decrease of the given linear function. The ellipsoid method simply looks for a single point in the feasible region itself, by constructing a smaller and smaller ellipsoids which enclose the feasible region. Interior point methods are somewhat similar to the simplex method, but interior point methods actually avoid the boundary of the feasible region altogether. These methods always stay in the interior of the polytope.

If we try to move in the directions of decrease on the boundary of the feasible region, we must deal directly with the constraints of the feasible region. Interior point methods somehow avoid this issue, by always staying away from the boundary of the feasible region. This way, the constraints on the feasible region are less burdensome, and we do not have to worry about potentially checking all of the vertices of the feasible region (which led to the simplex method being inefficient).

Let A be a real $m \times n$ matrix with $m \geq n$. Let $x, c \in \mathbb{R}^n$ and let $b \in \mathbb{R}^m$. Consider the linear program

$$\text{maximize} \quad c^T x \quad \text{subject to the constraints}$$
$$Ax \geq b.$$

Instead of maximizing $c^T x$ directly, we maximize a modified function. This function is modified to "penalize" points that are close to the boundary of the feasible region. There are many versions of this method. Below we present one such version.

For any $\beta \in \mathbb{R}$, let $D_\beta := \{x \in \mathbb{R}^n \colon Ax > b,\ c^T x > \beta\}$ and define the function $f_\beta \colon D_\beta \to \mathbb{R}$

$$f_\beta(x) := m \log(c^T x - \beta) + \sum_{i=1}^{m} \log((Ax)_i - b_i), \qquad \forall x \in D_\beta.$$

Note that D_β is a convex set, and f_β is a strictly concave function which goes to $-\infty$ on the boundary of D_β. So, if D_β is nonempty and bounded, then f_β has a unique maximum in D_β.

Algorithm 4.39 (An interior point method). Start with $\beta_0 \in \mathbb{R}$, $x^{(0)} \in \mathbb{R}^n$.

- Let $\beta_j := \beta_{j-1} + \frac{.05}{\sqrt{m}}(c^T x^{(j-1)} - \beta_{j-1})$, where $j \geq 1$.
- Perform one step of Newton's Method for maximizing f_{β_j}, starting at $x^{(j-1)}$, and ending at the point which we define to be $x^{(j)}$.
- If $j < 100\sqrt{m}$, return to step 1. Otherwise, stop.

4.5. **Duality.** A linear program has another linear program naturally associated to it, called the dual linear program. The relation of these two linear programs gives us a deeper understanding of the original linear program.

Definition 4.40 (**Dual Linear Program**). Let $c \in \mathbb{R}^n$, let $b \in \mathbb{R}^m$, and let A be a real $m \times n$ matrix. Let $x \in \mathbb{R}^n$ be a variable vector. Recall the linear program in standard form, which we refer to as the **primal problem**:

$$\text{minimize} \quad c^T x \quad \text{subject to the constraints}$$
$$Ax = b,\ x \geq 0.$$

Given this linear programming problem, we define the **dual problem** as follows: let $y \in \mathbb{R}^m$ be a variable vector, and define the linear program:

$$\text{maximize} \quad b^T y \quad \text{subject to the constraints}$$
$$A^T y \leq c.$$

Exercise 4.41. Show that the dual problem of the dual problem is the primal problem. (Hint: first write the constraint $A^T y \leq c$ as $A^T y + z = c$, $z \geq 0$.)

Definition 4.42. Let $y \in \mathbb{R}^n$. Let $R > 0$. We define the **ball of radius R centered at y** to be the set

$$B_R(y) := \{x \in \mathbb{R}^n \colon \|x - y\| \leq R\}.$$

Definition 4.43. Recall that a sequence of points $x^{(1)}, x^{(2)}, \ldots$ in $\mathbb{R}^n$ **converges** to a point $x \in \mathbb{R}^n$ if $\lim_{j\to\infty} \|x^{(j)} - x\| = 0$.

Let $K \subseteq \mathbb{R}^n$. We say that K is **closed** if the following condition is satisfied. If $x^{(1)}, x^{(2)}, \ldots$ is any sequence of points in K, and if this sequence converges to some point $x \in \mathbb{R}^n$, then we must have $x \in K$.

Recall also that K is **bounded** there exists some $R > 0$ such that $K \subseteq B_R(0)$.

Definition 4.44. Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be a function. We say that f is **continuous** if, for any $x \in \mathbb{R}^n$ and for any sequence of points $x^{(1)}, x^{(2)}, \ldots$ in $\mathbb{R}^n$ that converges to x, we have $\lim_{j\to\infty} |f(x^{(j)}) - f(x)| = 0$.

The following theorem is proven in Math 131B, so we will not give its proof.

Theorem 4.45 (**Extreme Value Theorem**). *Let $K \subseteq \mathbb{R}^n$ be a closed and bounded set. Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be a continuous function. Then f achieves its maximum and minimum on K. That is, there exist points $a, b \in K$ such that $f(a) \leq f(x) \leq f(b)$ for all $x \in K$. We denote $f(a) = \min_{x\in K} f(x)$ and $f(b) = \max_{x\in K} f(x)$.*

Exercise 4.46. Show that the intersection of two closed sets is a closed set.

Exercise 4.47. Let $b \in \mathbb{R}^n$. Define $f\colon \mathbb{R}^n \to \mathbb{R}$ by $f(x) := \|x - b\|$. Show that f is continuous. (Hint: show that it suffices to consider the case $b = 0$. In the case $b = 0$, use a triangle inequality.)

Lemma 4.48 (**Farkas' Lemma, Version 1**). *Let A be an $m \times n$ real matrix and let $b \in \mathbb{R}^m$. Then exactly one of the following statements holds.*

- *$\exists\ x \in \mathbb{R}^n$, $x \geq 0$ such that $Ax = b$.*
- *$\exists\ y \in \mathbb{R}^m$ such that $y^T A \geq 0$ and $y^T b < 0$.*

Remark 4.49. Geometrically, Farkas' Lemma says that either b belongs to the convex set $K := \{Ax \colon x \geq 0\}$, or there exists a vector y which separates b from K, in the sense that $y^T k \geq 0$ for every $k \in K$ and $y^T b < 0$.

Proof. First, note that both conditions cannot be satisfied. If both conditions were satisfied, then $Ax = b$, but then $x \geq 0$ and $y^T A \geq 0$ mean $0 \leq y^T Ax = y^T b < 0$, a contradiction.

It remains to show that if the first condition is not satisfied, then the second condition is satisfied. So, assume the first condition is not satisfied. Define $K := \{Ax \colon x \geq 0\}$. Note that K is a convex set. Since the first condition is not satisfied, $b \notin K$.

Let k be any point in K. Choose $R := 1 + \max(\|k\|, 2\|b\|)$. Then $k \in B_R(0)$, so that $B_R(0)$ has nonempty intersection with K. Note that $B_R(0)$ is a closed set, so $K \cap B_R(0)$ is also closed by Exercise 4.46. Also, $K \cap B_R(0)$ is bounded, since $(K \cap B_R(0)) \subseteq B_R(0)$. Define $f \colon \mathbb{R}^d \to \mathbb{R}$ by $f(x) := \|x - b\|$. Then f is continuous by Exercise 4.47. Since $K \cap B_R(0)$ is closed and bounded, f achieves its minimum value on $K \cap B_R(0)$, by Theorem 4.45. In particular, there exists $z \in K$ such that $\|z - b\| \leq \|x - b\| \leq R$, for all $x \in K \cap B_R(0)$. Any point $x \notin B_R(0)$ satisfies $\|x\| \geq 2\|b\| + 1$, so using the reverse triangle inequality and $0 \in K$,

$$\|x - b\| \geq \|x\| - \|b\| \geq \|b\| + 1 \geq f(0) \geq f(z) = \|z - b\|.$$

So, for any $x \in K$, we have

$$\|z - b\| \leq \|x - b\|, \qquad \forall\, x \in K \qquad (*)$$

Also, from Exercise 2.61 applied to the function $f(x) := \frac{1}{2}\|x - b\|^2$, $f \colon K \to \mathbb{R}$, $\nabla f(z) = z - b$ is orthogonal to any vector in K. Since $z \in K$, we have

$$z^T(z - b) = 0 \qquad (**)$$

Let $x \in K$ and let $0 < t < 1$. Since K is convex, we have $tx + (1-t)z \in K$. So, applying $(*)$, we have

$$\begin{aligned}(z-b)^T(z-b) = \|z - b\|^2 \leq \|tx + (1-t)z - b\|^2 &= (t(x-z) + z - b)^T(t(x-z) + z - b)\\ &= t^2\|x - z\|^2 + 2t(x-z)^T(z-b) + (z-b)^T(z-b).\end{aligned}$$

That is,

$$0 \leq t^2\|x - z\|^2 + 2t(x-z)^T(z-b).$$

Dividing by t, letting t go to zero,

$$0 \leq (x-z)^T(z-b) \overset{(**)}{=} x^T(z-b), \qquad \forall\, x \in K. \qquad (***)$$

So, define $y := z - b$. Since $z \in K$ and $b \notin K$, we have $y \neq 0$. And

$$y^T b = y^T(z - y) = (z-b)^T z - y^T y \overset{(**)}{=} -\|y\|^2 < 0.$$

We finally claim that $y^T A \geq 0$. Since $y^T x \geq 0$ for all $x \in K$ by $(***)$, we have $y^T Aw \geq 0$ for all $w \geq 0$. If there exists $i \in \{1, \ldots, n\}$ with $(y^T A)_i < 0$, then we could choose $w \in \mathbb{R}^n$ such that $w_i = 1$ and $w_j = 0$ for all $j \neq i$, $j \in \{1, \ldots, n\}$. Then $w \geq 0$ but $y^T Aw < 0$, a contradiction. We conclude that $y^T A \geq 0$.

□

Exercise 4.50 (Farkas' Lemma, Version 2). Let A be an $m \times n$ real matrix and let $b \in \mathbb{R}^m$. Then there exists $x \in \mathbb{R}^n$, such that $Ax \leq b$ if and only if: for all $y \in \mathbb{R}^m$ with $y \geq 0$ and $y^T A = 0$, we have $y^T b \geq 0$.

(Hint: $Ax \leq b$ has a solution if and only if $Ax^+ - Ax^- + z = b$, where $x^+, x^-, z \geq 0$. Apply Farkas' Lemma, Version 1, to the equality.)

Lemma 4.51 (Weak Duality for Linear Programming). *Consider the primal and dual linear programs defined in Definition 4.40. If x, y are in the feasible sets of the primal and dual, respectively, then $c^T x \geq b^T y$.*

Proof. Since x, y are feasible we have $Ax = b$, $x \geq 0$ and $A^T y \leq c$, so that $x^T A^T y \leq x^T c$ and

$$y^T b = y^T A x \leq x^T c.$$

□

The following theorem gives a strong relation between the primal and dual problems in linear programming

Theorem 4.52 (Strong Duality for Linear Programming). *Consider the primal and dual linear programs defined in Definition 4.40. Exactly one of the following four things occurs.*

- *The primal and dual are infeasible.*
- *The primal is unbounded and the dual is infeasible.*
- *The dual is unbounded and the primal is infeasible.*

($*$) *The primal and dual are feasible, and there exist x, y which are feasible for the primal and dual respectively such that $c^T x = b^T y$.*

Proof. First, note that all four of these situations are exclusive.

Suppose the primal problem is unbounded. Then there are a sequence of points $x^{(1)}, x^{(2)}, \ldots$ which are feasible for the primal problem such that $c^T x^{(i)} \to -\infty$ as $i \to \infty$. So, if there exists $y \in \mathbb{R}^m$ which is feasible for the dual problem, Lemma 4.51 says that $b^T y \leq c^T x^{(i)}$ for all $i \geq 1$, a contradiction. In conclusion, if the primal problem is unbounded then the dual problem is infeasible. A similar argument implies: if the dual problem is unbounded then the primal problem is infeasible.

So, the only remaining case to consider is that the primal and dual are not unbounded and one of them is feasible. We assume that the dual problem is feasible. By Exercise 4.41, assuming instead that the primal problem is feasible results in an identical argument.

Suppose the dual problem is feasible and its maximum value is $t \in \mathbb{R}$. If y is feasible for the dual problem, then $-b^T y \geq -t$. Then for any $\varepsilon > 0$, the set

$$\{y \in \mathbb{R}^m \colon A^T y \leq c,\, -b^T y \leq -(t + \varepsilon)\}$$

is empty. Equivalently, there does not exist $y \in \mathbb{R}^m$ with

$$\begin{pmatrix} A^T \\ -b^T \end{pmatrix} y \leq \begin{pmatrix} c \\ -(t+\varepsilon) \end{pmatrix}.$$

From Farkas' Lemma Version 2, Exercise 4.50, there exists $x \in \mathbb{R}^n$, $\lambda \in \mathbb{R}$ with $x \geq 0$, $\lambda \geq 0$ such that

$$\begin{pmatrix} x^T & \lambda \end{pmatrix} \begin{pmatrix} A^T \\ -b^T \end{pmatrix} = 0, \qquad \begin{pmatrix} x^T & \lambda \end{pmatrix} \begin{pmatrix} c \\ -(t+\varepsilon) \end{pmatrix} < 0.$$

That is, $x^T A^T - \lambda b^T = 0$ and $x^T c - \lambda(t+\varepsilon) < 0$.

If $\lambda = 0$, then $x^T A^T = 0$, $x \geq 0$ and $x^T c < 0$. Since the dual is feasible, let $y \in \mathbb{R}^m$ with $A^T y \leq c$. From Farkas' Lemma Version 2, Exercise 4.50, every $x \geq 0$ with $x^T A^T = 0$ satisfies $x^T c \geq 0$, a contradiction. We therefore conclude that $\lambda \neq 0$, so that $\lambda > 0$. Define $\widetilde{x} := x/\lambda$. Then $\widetilde{x} \geq 0$, $A\widetilde{x} = b$ and $\widetilde{x}^T c < (t+\varepsilon)$. That is, $\widetilde{x}$ is feasible for the primal problem, and the minimum value of the primal problem is at most $t+\varepsilon$. Since $\varepsilon > 0$ is arbitrary, the minimum value of the primal problem is at most t. Since t is the maximum of the dual problem, Lemma 4.51 implies that the maximum of the dual problem and the minimum of the primal problem are equal. $\square$

Exercise 4.53. Give an example of a linear program where both the primal and dual problems are infeasible.

Exercise 4.54. Consider the linear program

$$\text{maximize} \quad y_1 \quad \text{subject to the constraints}$$

$$y_1 \geq 0,\ y_2 \geq 0,\ y_1 + y_2 \leq 1.$$

Draw the feasible region, and draw the point where the maximum occurs.

Find the dual problem, draw the feasible region for the dual problem, and draw the point where the minimum occurs.

Exercise 4.55. Using the Strong Duality for Linear Programming, prove von Neumann's Minimax Theorem from Game Theory:

Let m, n be positive integers. Let A be an $m \times n$ real matrix. Define

$$\Delta_m := \{x = (x_1, \ldots, x_m) \in \mathbb{R}^m \colon \sum_{i=1}^m x_i = 1,\ x_i \geq 0,\ \forall\, 1 \leq i \leq m\}.$$

Then

$$\max_{x \in \Delta_m} \min_{y \in \Delta_n} x^T A y = \min_{y \in \Delta_n} \max_{x \in \Delta_m} x^T A y.$$

(Hint: consider the linear program of maximizing t subject to the constraints: $t - \sum_{j=1}^n a_{ij} y_j \leq 0$ for all $1 \leq i \leq m$, $\sum_{j=1}^n y_j \leq 1$, $y \geq 0$.)

(Hint: First show that $\max_{x \in \Delta_m} x^T A y = \max_{i=1,\ldots,m} (Ay)_i = \max_{i=1,\ldots,m} \sum_{j=1}^n a_{ij} y_j$. So, the linear program mentioned above should compute the term on the right side of the equality.)

Strong Duality, Theorem 4.52 has many nice consequences, including the max flow/min cut theorem.

Let $V = \{1, \ldots, n\}$ be vertices, which we can think of as n cities. We think of $1 \in V$ as the "source" of some product, and $n \in V$ as the "sink," or destination for the products. For every $i, j \in V$ with $i \neq j$ there is some maximum amount $c_{ij} \geq 0$ of product that can be routed from i to j. The goal of the maximum flow problem is to maximize the amount of product that can be routed from the source to the sink. Let P be the set of non-self-intersecting paths that start at vertex 1 and end at vertex n. That is, $p \in P$ is an ordered sequence of distinct points $(v_1, v_2, \ldots, v_{k-1}, v_k)$ where $v_1, \ldots, v_k \in V$ are distinct, $v_1 = 1$ and $v_k = n$. We write $(i, j) \in p$ to denote that the elements $i, j \in V$ appear adjacent to each other in the ordered list of points in p.

For every $p \in P$, we let $f_p \in \mathbb{R}$, $f_p \geq 0$ be the amount of product that travels along the path p. That is, we can think of a box containing an amount of product traveling along the path p. Then the maximum flow problem is:

$$\text{maximize} \quad \sum_{p \in P} f_p \quad \text{subject to the constraints}$$

$$\sum_{p \in P\colon (i,j) \in p} f_p \leq c_{ij}, \qquad \forall\, i, j \in \{1, \ldots, n\} \text{ with } i \neq j.$$

$$f_p \geq 0, \qquad \forall\, p \in P.$$

Exercise 4.56. Let $E := \{(i,j) \in \{1, \ldots, n\} \colon i \neq j\}$. Show that the dual of the maximum flow problem is:

$$\text{minimize} \quad \sum_{(i,j) \in E} c_{ij} x_{ij} \quad \text{subject to the constraints}$$

$$\sum_{(i,j) \in p} x_{ij} \geq 1, \qquad \forall\, p \in P.$$

$$x_{ij} \geq 0, \qquad \forall\, (i,j) \in E.$$

The value of the primal problem is the maximum amount of product that can flow from the source to the sink. It turns out that we can interpret the dual problem in a different but enlightening way. Let $S \subseteq V$ with $1 \in S$ and $n \notin S$. Define

$$c(S) := \sum_{(i,j) \in E\colon i \in S,\, j \in S^c} c_{ij}.$$

That is, $c(S)$ is the amount of product that is flowing outwards from S into S^c. For this reason, we refer to the ordered pair of sets (S, S^c) as a **cut**.

Lemma 4.57. *Let C be the minimum value of the dual problem for the max flow problem. Then*

$$C \leq \min_{S \subseteq V\colon 1 \in S,\, n \notin S} c(S).$$

Proof. For any $S \subseteq V$ with $1 \in S$, $n \notin S$, define x so that $x_{ij} = 1$ if $i \in S$ and $j \notin S$, and $x_{ij} = 0$ otherwise. Note that x is feasible for the dual problem, since any path that begins at 1 and ends at n must eventually exit the set S (since $1 \in S$ and $n \notin S$). And $c(S) = \sum_{(i,j) \in E} c_{ij} x_{ij}$. The Lemma follows by the definition of C. $\square$

Corollary 4.58 (**Max Flow/ Min Cut Theorem**). *Let f be the maximum value of the maximum flow problem. Then*

$$f = \min_{S \subseteq V\colon 1 \in S,\, n \notin S} c(S).$$

That is, the maximum flow is equal to the minimum cut.

Proof. (Optional reading, since it uses some probability). First, note that the primal problem is bounded and feasible, so Strong Duality, Theorem 4.52, implies that the primal and dual problem are both bounded and feasible.

Let x achieve the minimum value of the dual problem. For any $v \in V$, let P_v be the set of non self-intersecting paths from 1 to v. Define $d\colon V \to \mathbb{R}$ so that $d(v) := \min\{\sum_{(i,j) \in p} x_{ij} \colon p \in P_v\}$. Note that $d(1) = 0$ and $d(n) \geq 1$, by the first constraint of the dual problem. For any $t \in [0,1)$ define $S_t := \{v \in V : d(v) \leq t\}$. Then $1 \in S_t$ and $n \notin S_t$ for any $t \in [0,1)$.

It turns out that a randomly chosen set S_t will minimize $c(\cdot)$. Let T be a random variable which is uniformly distributed in $[0,1)$. For any $i \in V$, define $t' := \min\{t \geq 0 : i \in S_t\}$. For any $t < t'$ we have $i \notin S_t$. And, by definition of t', for any $t \geq t' + x_{ij}$, we have $i, j \in S_t$. Therefore, for any $(i,j) \in E$ we have $\mathbf{P}(i \in S_T,\, j \notin S_T) \leq x_{ij}$. So,

$$\mathbf{E}c(S_T) = \sum_{(i,j)\in E} c_{ij}\mathbf{P}(i \in S_T,\, j \notin S_T) \leq \sum_{(i,j)\in E} c_{ij}x_{ij}.$$

Therefore, there exists some $t \in [0,1)$ such that $c(S_t) \leq \sum_{(i,j)\in E} c_{ij}x_{ij}$. Lemma 4.57 then implies that $c(S_t) = \sum_{(i,j)\in E} c_{ij}x_{ij}$. Finally, Strong Duality, Theorem 4.52(∗), concludes the proof. □

Exercise 4.59 (Set Cover Problem). Let U be a finite set, and let $S_1, \ldots, S_n$ be subsets of U such that $\cup_{i=1}^n S_i = U$. The goal of the set cover problem is to find the minimum number of sets $S_1, \ldots, S_n$ whose union is still all of U. For example, we could think of $S_1, \ldots, S_n$ as time intervals for shifts of workers, and we want to minimize the number of shifts that occur, while always having at least one person on the job.

For each $1 \leq i \leq n$, let $x_i = 1$ if we want to keep the set S_i, and $x_i = 0$ if we do not keep the set S_i. Then we can state the set cover problem as follows:

$$\text{minimize} \quad \sum_{i=1}^n x_i \quad \text{subject to the constraints}$$

$$\sum_{i\in\{1,\ldots,n\}\colon u\in S_i} x_i \geq 1, \quad \forall\, u \in U.$$

$$x_i \in \{0,1\}, \quad \forall\, i \in \{1,\ldots,n\}.$$

Show that the first constraint implies that $\cup_{i\in\{1,\ldots,n\}\colon x_i=1} S_i = U$.

We relax this integer program to a linear program:

$$\text{minimize} \quad \sum_{i=1}^n x_i \quad \text{subject to the constraints}$$

$$\sum_{i\in\{1,\ldots,n\}\colon u\in S_i} x_i \geq 1, \quad \forall\, u \in U.$$

$$0 \leq x_i \leq 1, \quad \forall\, i \in \{1,\ldots,n\}.$$

Find the dual of the linear program.

(Optional challenge: can you modify what we did for the minimum vertex cover problem, and create an approximation algorithm for the set cover problem? If so that would be nice, since the minimum vertex cover problem is also NP-complete.)

5. Semidefinite Programming

Recall that linear programming involves minimizing a linear function on a polytope. Exercises 4.19 and 4.17 show that we only need to look through a finite number of extreme points of the polytope in order to minimize the linear function.

A semidefinite program involves minimizing a linear function on linear subsets of the cone of positive semidefinite matrices. Such a subset of positive semidefinite matrices could have infinitely many extreme points, so the simplex method cannot be used to solve a semidefinite program.

Exercise 5.1. Let K be the following set of positive semidefinite 2×2 matrices

$$K = \left\{ \begin{pmatrix} a & b \\ b & c \end{pmatrix} \geq 0 \colon a, b, c \in \mathbb{R},\ a + c = 1 \right\}.$$

First, show that K is convex and nonempty. Then show that K has infinitely many extreme points.

(Hint: which matrices in K have determinant zero?)

5.1. Introduction.

Now that we have described the ellipsoid method in the setting of linear programming, it is a short step to apply the method to semidefinite programming. With this analysis, we see the (often-observed) fact that semidefinite programming is a slight generalization of linear programming. In semidefinite programming, we are still optimizing a linear function, but the constraints are much different. We consider the following problem:

Definition 5.2. A **Semidefinite Program** is an optimization problem of the following form

$$\text{minimize} \quad c^T x \quad \text{subject to the constraint}$$
$$F(x) \geq 0.$$

where $c \in \mathbb{R}^m$ is fixed, $x \in \mathbb{R}^m$, and $F(x)$ is an $n \times n$ matrix with

$$F(x) := F_0 + \sum_{i=1}^{m} x_i F_i, \quad F_i \in \mathbb{R}^{n \times n}, F_i^T = F_i, \qquad \forall\, 0 \leq i \leq m.$$

Here the notation $F(x) \geq 0$ means that $F(x)$ is positive semidefinite.

One can also define a semidefinite program as follows.

Remark 5.3. The following optimization problem is a special case of a semidefinite program, which is often seen in the literature:

$$\text{minimize} \quad \mathrm{Tr}(C^T Y) \quad \text{subject to the constraints}$$
$$\mathrm{Tr}(D_i^T Y) \leq \gamma_i,\ \forall\, 1 \leq i \leq m, \qquad Y \geq 0.$$

where C, Y, D_i are real symmetric $n \times n$ matrices for all $1 \leq i \leq m$, and $\gamma_1 \in \mathbb{R}$ for all $1 \leq i \leq m$. As above, the number of constraints m may exceed n.

To see that this program is a special case of Definition 5.2, for any $1 \leq i \leq j \leq n$, let F_{ij} be an $n \times n$ symmetric matrix all of whose entries are zero except for the entries (i, j) and (j, i), which are 1. Then $Y = \sum_{1 \leq i \leq j \leq n} y_{ij} F_{ij}$, and the program in Remark 5.3 can be written as

$$\text{minimize} \quad c^T y \quad \text{subject to the constraint}$$

$$\begin{pmatrix} A & 0 \\ 0 & \sum_{1\leq i\leq j\leq n} F_{ij}y_{ij} \end{pmatrix} \geq 0.$$
where A is a diagonal matrix such that $A_{ii} = -\mathrm{Tr}(D_i^T Y) + \gamma_i$, $\forall\, 1 \leq i \leq n$, and c is a vector whose entries are the entries of C, so that $\mathrm{Tr}(C^T Y) = \sum_{i,j=1}^n C_{ij}Y_{ij} = c^T y$, and y is a vector whose entries are the entries of Y.

Remark 5.4. A **linear program** $\max\{c^T x \colon Ax + b \geq 0\}$ is a special case of a semidefinite program, since we can write the condition $Ax + b \geq 0$ equivalently as:

$$\begin{pmatrix} (Ax+b)_1 & 0 & \cdots & 0 \\ 0 & (Ax+b)_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & (Ax+b)_n \end{pmatrix} \geq 0.$$

Or, equivalently,

$$= \begin{pmatrix} b_1 + \sum_{j=1}^n A_{1j}x_j & 0 & \cdots & 0 \\ 0 & b_2 + \sum_{j=1}^n A_{2j}x_j & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & b_n + \sum_{j=1}^n A_{nj}x_j \end{pmatrix} \geq 0.$$

Put another way, semidefinite programming is a generalization of linear programming. Remarkably, the Ellipsoid method and Interior Point Methods can still solve semidefinite programming problems in a polynomial amount of time.

We briefly mention how the Ellipsoid method can be applied to semidefinite programs.

5.2. **Algorithms.** As noted in Proposition 4.26, "solving" a semidefinite program reduces to finding a point in any given nonempty feasible set.

Theorem 5.5. *(**Solvability of Semidefinite Programs**) Suppose we have a semidefinite program with a feasible region $P \subseteq \mathbb{R}^m$. (In particular P is a convex subset of $\mathbb{R}^m$). Let $\varepsilon > 0$. Assume that $B \subseteq \mathbb{R}^m$ is a Euclidean Ball such that $P \subseteq B$. Assume that P contains a Euclidean ball $B' \subseteq \mathbb{R}^m$ of any center and radius. Let $c' \geq 1$ so that $\mathrm{vol}_m(B)/\mathrm{vol}_m(P) \leq c'$. Then there exists an algorithm that runs in time polynomial in m, n and $\log(c')$ which either finds some $p \in P$ or concludes that $P = \emptyset$.*

Proof. We need only slightly modify our use of the Ellipsoid method in Theorem 4.32. The proof proceeds in essentially the same way. The only change that we require is to describe how to cut the ellipsoid.

Consider the condition $F(x) = F_0 + \sum_{i=1}^m x_i F_i \geq 0$. Suppose at the k^{th} step of the Ellipsoid Method, the center of the current ellipsoid E_k is $x \in \mathbb{R}^m$ violates this inequality. Then $\exists$ $u \in \mathbb{R}^n$ with $u^T F(x) u < 0$. Define $g \in \mathbb{R}^m$ by $g_i := -u^T F_i u$, $i = 1, \ldots, m$. Then for any $z \in \mathbb{R}^m$ with $\langle g, (z-x)\rangle \geq 0$ we have

$$u^T F(z) u = u^T \left(F_0 + \sum_{i=1}^m (x_i + (z_i - x_i)) F_i \right) u = u^T F(x) u - \langle g, (z-x)\rangle < 0.$$

That is, $\{z \in \mathbb{R}^m \colon F(z) \geq 0\} \subseteq \{z \in \mathbb{R}^m \colon \langle g, (z-x)\rangle < 0\}$. But $z = x$ is on the boundary of the half space $\{z \in \mathbb{R}^m \colon \langle g, (z-x)\rangle \leq 0\}$. Therefore, we have found a hyperplane through

x that cuts $\mathbb{R}^m$ into two regions, one of which contains $\{x\colon F(x) \geq 0\}$. That is, from the ellipsoid E_k, we then define the half-ellipsoid $E_k \cap \{z \in \mathbb{R}^m \colon \langle g, (z-x)\rangle \leq 0\}$. The rest of the proof proceeds as written. $\square$

5.3. **MAX-CUT.** The MAX-CUT problem asks for the partition of the vertices of an undirected graph into two sets S and S^c that maximizes the number of edges that go from S to S^c. For a graph on n vertices, it is expected that in general we cannot solve the MAX-CUT problem in time which is polynomial in n. More specifically, MAX-CUT is an NP-complete problem. Fortunately, one can approximately solve the MAX-CUT problem in time polynomial in n, within a multiplicative factor of .87856.

Definition 5.6 (**MAX-CUT**)**.** We define the weighted MAX-CUT problem for any positive integer n. In this problem, we are given a symmetric matrix $\{a_{ij}\}_{i,j=1}^n$ with $a_{ij} \geq 0$ for all $i, j \in \{1, \ldots, n\}$. We can think of a_{ij} as the weight of an edge connecting vertices i and j. The goal of the MAX-CUT problem is to find the quantity:

$$\max_{S \subseteq \{1,\ldots,n\}} \sum_{\substack{i,j\in\{1,\ldots,n\}\colon\\ i\in S,\, j\notin S}} a_{ij}.$$

Written another way, we want to compute

$$\max_{\varepsilon_1,\ldots,\varepsilon_n \in\{-1,1\}} \frac{1}{2}\sum_{i,j=1}^n a_{ij}(1-\varepsilon_i\varepsilon_j). \tag{7}$$

(Given $S \subseteq \{1,\ldots,n\}$, define $\varepsilon_1,\ldots,\varepsilon_n$ such that $\varepsilon_i = 1$ if $i \in S$, and $\varepsilon_i = -1$ if $i \notin S$. Then $(1-\varepsilon_i\varepsilon_j)/2 = 1$ if $i \in S$ and $j \notin S$ and $(1-\varepsilon_i\varepsilon_j)/2 = 0$ if $i \in S$ and $j \in S$.)

We now want to adjust this problem in order to apply semidefinite programming. Note that the function in (7) is quadratic in the $\varepsilon_1,\ldots,\varepsilon_n$ terms, so we cannot hope to apply linear programming. We approximate (7) by replacing the numbers ε_i by unit vectors v_i in the unit sphere $S^{n-1} := \{x \in \mathbb{R}^n \colon \|x\| = 1\}$, and by maximizing the same quantity.

$$\max_{v_1,\ldots,v_n\in S^{n-1}} \frac{1}{2}\sum_{i,j=1}^n a_{ij}(1-\langle v_i, v_j\rangle). \tag{8}$$

Then (8) defines a semidefinite program! Given $v_1,\ldots,v_n \in S^{n-1}$, we can define a matrix $Y := (v_1,\ldots,v_n)(v_1,\ldots,v_n)^T$, which is positive semidefinite by Exercise 2.21, with $Y_{ii} = 1$ for all $1 \leq i \leq n$. Conversely, if Y is a real positive semidefinite matrix with $Y_{ii} = 1$ for all $1 \leq i \leq n$, then by Exercise 2.21, $Y = BB^T$ for some $n \times n$ matrix B. Since $Y_{ii} = 1$ for all $1 \leq i \leq n$, we conclude that the columns of B are in S^{n-1}. That is, the constraint $v_1,\ldots,v_n \in S^{n-1}$ can be equivalently written as $Y \geq 0$, $Y_{ii} = 1$ for all $1 \leq i \leq n$. That is, the expression in (8) is a semidefinite program in the form of Remark 5.3.

Since unit vectors in S^{n-1} include unit vectors in $\{-1,1\}$, the quantity (8) exceeds the quantity (7).

$$\max_{v_1,\ldots,v_n\in S^{n-1}} \frac{1}{2}\sum_{i,j=1}^n a_{ij}(1-\langle v_i, v_j\rangle) \geq \max_{\varepsilon_1,\ldots,\varepsilon_n \in\{-1,1\}} \frac{1}{2}\sum_{i,j=1}^n a_{ij}(1-\varepsilon_i\varepsilon_j). \tag{9}$$

Finally, the vectors achieved by solving (8) can be related back to (7) using a randomized rounding method.

Theorem 5.7 (**Approximation Algorithm for MAX-CUT**). *There exists a randomized polynomial time algorithm whose input is a real symmetric matrix $\{a_{ij}\}_{i,j=1}^{n}$ of nonnegative numbers, and whose output is $\delta_1,\ldots,\delta_n\in\{-1,1\}$, such that the expected value of $\frac{1}{2}\sum_{i,j=1}^{n}a_{ij}(1-\delta_i\delta_j)$ is at least*

$$(.87856)\cdot\max_{\varepsilon_1,\ldots,\varepsilon_n\in\{-1,1\}}\frac{1}{2}\sum_{i,j=1}^{n}a_{ij}(1-\varepsilon_i\varepsilon_j).$$

(Optional reading, since some probability is used). Let $r_1,\ldots,r_n\in S^{n-1}$ achieve the maximum in (8). These vectors can be found in polynomial time since (8) is a semidefinite program. Let z be a uniformly random element of S^{n-1}. For any $1\le i\le n$, we "round" the vectors r_i to an element $\delta_i\in\{-1,1\}$ as follows:

$$\delta_i:=\begin{cases}1 & \text{, if } z^T r_i>0\\ -1 & \text{, if } z^T r_i\le 0.\end{cases}$$

In expectation, we then compute

$$\begin{aligned}\mathbf{E}\frac{1}{2}\sum_{i,j=1}^{n}a_{ij}(1-\delta_i\delta_j)&=\sum_{i,j=1}^{n}a_{ij}\frac{\cos^{-1}\langle r_i,r_j\rangle}{\pi}\\&\geq\left[\min_{0\le\theta\le\pi}\frac{2}{\pi}\frac{\theta}{1-\cos\theta}\right]\frac{1}{2}\sum_{i,j=1}^{n}a_{ij}(1-\langle r_i,r_j\rangle)\\&\approx[.87856]\frac{1}{2}\sum_{i,j=1}^{n}a_{ij}(1-\langle r_i,r_j\rangle)\\&=[.87856]\max_{v_1,\ldots,v_n\in S^{n-1}}\frac{1}{2}\sum_{i,j=1}^{n}a_{ij}(1-\langle v_i,v_j\rangle).\end{aligned}\tag{10}$$

Finally, combining (10) and (9),

$$\mathbf{E}\frac{1}{2}\sum_{i,j=1}^{n}a_{ij}(1-\delta_i\delta_j)\geq[.87856]\max_{\varepsilon_1,\ldots,\varepsilon_n\in\{-1,1\}}\frac{1}{2}\sum_{i,j=1}^{n}(1-\varepsilon_i\varepsilon_j).\tag{11}$$

□

In summary, we used semidefinite programming to approximately solve MAX-CUT in polynomial time within a multiplicative factor .87856, as shown by (11). No one has been able to improve upon this approximation factor, and there is strong evidence that it may never be improved at all.

Exercise 5.8. Let $u,v\in\mathbb{R}^3$ be vectors such that $\|u\|=\|v\|=1$. Let $S^2=\{x\in\mathbb{R}^3\colon\ \|x\|=1\}$. For any $t\in\mathbb{R}$, define

$$\text{sign}(t):=\begin{cases}1 & \text{, if } t>0\\ -1 & \text{, if } t\le 0.\end{cases}$$

Let dS denote the surface area element on S^2. Show the following calculation, which was used in the analysis of the MAX-CUT semidefinite program.

$$\frac{\int_{S^2} \mathrm{sign}(\langle u, x\rangle) \cdot \mathrm{sign}(\langle v, x\rangle) dS(x)}{\int_{S^2} dS(x)} = 1 - \frac{2}{\pi} \cos^{-1}(\langle u, v\rangle).$$

Exercise 5.9 (Optional)**.** Using any numerical method you wish to use, compute

$$\min_{0 \leq \theta \leq \pi} \frac{2}{\pi} \frac{\theta}{1 - \cos\theta}$$

to ten decimal places of accuracy.

6. Calculus of Variations

Up until this point, we have focused on optimization problems, mostly in the finite-dimensional vector space $\mathbb{R}^n$. Certain optimization problems are naturally phrased in infinite-dimensional vector spaces. For example, the set of functions $\{f : [0, 1] \to \mathbb{R} : f(0) = f(1) = 0\}$ is an infinite-dimensional vector space, and perhaps we want to maximize or minimize some integral quantity over this set of functions. Due to the infinite number of dimensions, some issues become more subtle. Most importantly, the existence of a minimum or maximum becomes more difficult to justify.

Optimization in infinite-dimensional vector spaces is called the calculus of variations, or variational calculus. Rather than providing a broad theory of the calculus of variations, we will focus on a few specific examples.

6.1. **Shortest Paths in the Plane.**

Exercise 6.1. Let $g : [0, 1] \to \mathbb{R}$ be a continuous function. Assume that, for any C^1function $h : [0, 1] \to \mathbb{R}$ with $h(0) = h(1) = 0$, we have

$$\int_0^1 g(x)h(x)dx = 0.$$

Conclude that $g(x) = 0$ for all $x \in [0, 1]$. (Hint: Argue by contradiction. Assume g is nonzero somewhere. Use the definition of continuity of g to show that g is nonzero in some interval. Then choose h such that h is only nonzero on this interval.)

Exercise 6.2. Let $D = \{(x, y) \in \mathbb{R}^2 : x^2 + y^2 \leq 1\}$ be the unit disc in the plane and let $\partial D = \{(x, y) \in \mathbb{R}^2 : x^2 + y^2 = 1\}$ be the boundary of D. Let A be the set of all functions $f : D \to \mathbb{R}$ such that f is a C^2 function, such that $f(x) = 0$ for all $x \in \partial D$, and $f(0) = 1$. Show that there does not exist a function $g \in A$ such that

$$\iint_D \|\nabla g(x)\| \, dx = \min_{f \in A} \iint_D \|\nabla f(x)\| \, dx$$

(Hint: it suffices to find a sequence of functions $f_1, f_2, \ldots \in A$ where $\iint_D \|\nabla f_k(x)\| \, dx \to 0$ as $k \to \infty$. Why is this sufficient? Second hint: it may be easier to use polar coordinates.)

Proposition 6.3 (**Shortest Paths in the Plane**)**.** *Let $f : [0, 1] \to \mathbb{R}$ be a C^1 function. Assume $f(0) = 0$ and $f(1) = 1$. Then*

$$\sqrt{2} \leq \int_0^1 \sqrt{1 + \left(\frac{d}{dx} f(x)\right)^2} dx.$$

So, the shortest path between $(0,0) \in \mathbb{R}^2$ and $(1,1) \in \mathbb{R}^2$ is the straight line between them.

Proof. We use the following fact, which we will not prove. There exists a function $g\colon [0,1] \to \mathbb{R}$ such that g is C^1, $g(0) = 0$, $g(1) = 1$, and such that

$$\int_0^1 \sqrt{1 + \left(\frac{d}{dx} g(x)\right)^2} dx = \min_{\substack{f\colon [0,1]\to\mathbb{R}, \\ f(0)=0,\, f(1)=1,\, f\in C^1}} \int_0^1 \sqrt{1 + \left(\frac{d}{dx} f(x)\right)^2} dx. \qquad (*)$$

In general, assuming the existence of an extremum is **not allowed**, since the extremum may not exist, as we saw in Exercise 6.2.

Let $t \in \mathbb{R}$, and let $h\colon [0,1] \to \mathbb{R}$ such that $h(0) = h(1) = 0$ and such that h is a C^1 function. Let $g_t := g + th$. Then $g_t(0) = 0$, $g_t(1) = 1$. So, since g minimizes the length, the following derivative is zero:

$$0 = \frac{d}{dt}|_{t=0} \int_0^1 \sqrt{1 + \left(\frac{d}{dx} g_t(x)\right)^2} dx = \int_0^1 \left(1 + \left(\frac{d}{dx} g(x)\right)^2\right)^{-1/2} \frac{d}{dx} g(x) \frac{d}{dx} h(x) dx$$

$$= -\int_0^1 \frac{d}{dx}\left[\left(1 + \left(\frac{d}{dx} g(x)\right)^2\right)^{-1/2} \frac{d}{dx} g(x)\right] h(x) dx.$$

In the last line, we integrated by parts. Then, by Exercise 6.1,

$$\frac{d}{dx}\left[\left(1 + \left(\frac{d}{dx} g(x)\right)^2\right)^{-1/2} \frac{d}{dx} g(x)\right] = 0.$$

That is, there exists a constant $c \in \mathbb{R}$ such that

$$\left(1 + \left(\frac{d}{dx} g(x)\right)^2\right)^{-1/2} \frac{d}{dx} g(x) = c.$$

Rearranging a bit,

$$\left(\frac{d}{dx} g(x)\right)^2 (1 - c^2) = c^2.$$

If $c^2 = 1$ we get a contradiction $0 = 1$. So, we can divide both sides by $1 - c^2$ to conclude that $|(d/dx)g(x)|$ is constant. Since $(d/dx)g(x)$ is continuous, we conclude that $\frac{d}{dx}g(x)$ is constant. Since $g(0) = 0$ and $g(1) = 1$, we must therefore have $g(x) = x$ for all $x \in [0,1]$. Equation $(*)$ therefore concludes the proposition. □

6.2. **Curves of Quickest Descent.** Suppose we have an object in the plane that falls from the point $(0,1)$ to the point $(1,0)$ under the influence of gravity (which acts in the negative vertical direction). Suppose also the object moves along a path f. That is, $f\colon [0,1] \to \mathbb{R}$, $f(0) = 1$, $f(1) = 0$. If the object has mass m and velocity $v(x)$ at the point $x \in \mathbb{R}^2$, and if G is the gravitational constant, then the law of conservation of energy says

$$\frac{1}{2} m (v(x))^2 + mGf(x) = mGf(0) = mG.$$

Using that velocity is distance over time, then solving for $v(x)$, the time that the object takes to traverse f is

$$\int_0^1 \frac{\sqrt{1+\left(\frac{df}{dx}\right)^2}}{v(x)}dx = \frac{1}{\sqrt{2G}}\int_0^1 \sqrt{\frac{1+\left(\frac{df}{dx}\right)^2}{1-f(x)}}dx.$$

(Since the object falls under the influence of gravity, it is sensible to assume that $f(x) < 1$ for all $x > 0$, so that we do not divide by zero.)

The Brachistochrone problem asks for the path f that takes the shortest amount of time to fall from one point to another. For example, what is the path f that takes the shortest amount of time to fall from $(0, 1)$ to $(1, 0)$? For the purpose of intuition, we could think of a small bead traveling along the rigid path f. (And we neglect any frictional forces.) Your first guess might be to take f to be a straight line between the two points: $f(x) = -x + 1$ for all $x \in [0, 1]$. However, this is not the quickest path! The fastest path is a cycloid.

Theorem 6.4 (Brachistochrone Problem). *Let A be the set of all functions $f\colon [0, 1] \to \mathbb{R}$ such that f is a C^1 function on $(0, 1)$, $f(0) = 1$, $f(1) = 0$ and $f(x) < 1$ for all $0 < x \leq 1$. Assume there exists $g \in A$ satisfying*

$$\int_0^1 \sqrt{\frac{1+\left(\frac{d}{dx}g(x)\right)^2}{1-g(x)}}dx = \min_{f\in A}\int_0^1 \sqrt{\frac{1+\left(\frac{d}{dx}f(x)\right)^2}{1-f(x)}}dx.$$

Then g is a cycloid.

Proof. Let $t \in \mathbb{R}$, and let $h\colon [0, 1] \to \mathbb{R}$ such that $h(0) = h(1) = 0$ and such that h is a C^1 function. Let $g_t := g + th$. Then $g_t(0) = 1$, $g_t(1) = 0$. So, since g minimizes the length, the following derivative is zero:

$$0 = \frac{d}{dt}|_{t=0}\int_0^1 \sqrt{\frac{1+\left(\frac{d}{dx}g_t(x)\right)^2}{1-g_t(x)}}dx.$$

Instead of differentiating under the integral explicitly as before, we differentiate symbolically. Let $a, b \in \mathbb{R}$ and define $L(a, b) := \sqrt{\frac{1+a^2}{1-b}}$. We use prime notation to denote derivatives with respect to x, and we omit the arguments of L for brevity. Then

$$\begin{aligned}0 &= \frac{d}{dt}|_{t=0}\int_0^1 L(g_t'(x), g_t(x))dx\\ &= \int_0^1 \left(h'(x)\frac{\partial L}{\partial a} + \frac{\partial L}{\partial b}h(x)\right)dx = \int_0^1 \left(-\frac{d}{dx}\frac{\partial L}{\partial a} + \frac{\partial L}{\partial b}\right)h(x)dx.\end{aligned}$$

In the last line, we integrated by parts. Then, by Exercise 6.1,

$$-\frac{d}{dx}\frac{\partial L}{\partial a} + \frac{\partial L}{\partial b} = 0. \qquad (*)$$

Instead of working with $(*)$ directly, we investigate a related conserved quantity H. For any $a, b \in \mathbb{R}$, let $H(a, b) := a\frac{\partial L}{\partial a}(a, b) - L(a, b)$. We now show that

$$\frac{d}{dx}H\left(\frac{d}{dx}g(x), g(x)\right) = 0. \qquad (**)$$

From the Chain rule and $(*)$,

$$\begin{aligned}\frac{d}{dx}H\left(g'(x),g(x)\right)&=\frac{d}{dx}\left(g'(x)\frac{\partial L}{\partial a}-L\right)\\&=g''(x)\frac{\partial L}{\partial a}+g'(x)\frac{d}{dx}\frac{\partial L}{\partial a}-g''(x)\frac{\partial L}{\partial a}-g'(x)\frac{\partial L}{\partial b}\\&=g'(x)\left(\frac{d}{dx}\frac{\partial L}{\partial a}-\frac{\partial L}{\partial b}\right)=0.\end{aligned}$$

So, $(**)$ holds. That is, there exists $c\in\mathbb{R}$ such that $H(g'(x),g(x))=c$. Using the definition of H and L,

$$c=H(g'(x),g(x))=\frac{(g'(x))^2}{\sqrt{1+(g'(x))^2}\sqrt{1-g(x)}}-\sqrt{\frac{1+(g'(x))^2}{1-g(x)}}=\frac{-1}{\sqrt{1+(g'(x))^2}\sqrt{1-g(x)}}.$$

Squaring both sides and rearranging (and noting that $c=0$ cannot occur),

$$(1-g(x))(1+(g'(x))^2)=c^{-2}.$$

We can solve this differential equation by separation of variables. We have

$$\frac{dg}{dx}=\sqrt{\frac{1}{c^2(1-g(x))}-1}=\sqrt{\frac{1-c^2(1-g(x))}{c^2(1-g(x))}}=\sqrt{\frac{c^{-2}-(1-g(x))}{1-g(x)}}.$$

That is,

$$x=\int\sqrt{\frac{1-g}{c^{-2}-(1-g)}}dg.$$

The proof then concludes via the following exercise. $\square$

Exercise 6.5. Assume that $g\colon[0,1]\to\mathbb{R}$ with $g(x)<1$ for all $0<x\leq 1$, $g(0)=1$, and $g(1)=0$. Assume that for some $c>0$ we have

$$x=\int\sqrt{\frac{1-g}{c^{-2}-(1-g)}}dg.$$

By substituting $1-g=c^{-2}\sin^2(\theta/2)=c^{-2}(1-\cos\theta)/2$, conclude that

$$x=x(\theta)=c^{-2}(\theta-\sin\theta).$$

In summary, $(x(\theta),g(\theta))=(c^{-2}(\theta-\sin\theta),1-c^{-2}(1-\cos\theta)/2)$. That is, the curve is a cycloid.

Exercise 6.6 (Isoperimetric Inequality in the Plane). Let $t\in[0,1]$. Let $s(t)=(x(t),y(t))$ be a parametrization of a curve in the plane. Assume that $s(t)\neq s(t')$ for all $t,t'\in[0,1)$ with $t\neq t'$. Assume also that $s(0)=s(1)$. That is, the curve does not intersect itself except at the points $t=0$ and $t=1$.

Assume that x,y are C^1 functions. The length of the curve s is defined to be

$$L(s):=\int_0^1\|s'(t)\|\,dt=\int_0^1\sqrt{(x'(t))^2+(y'(t))^2}dt.$$

The area enclosed by the curve is defined to be

$$A(s) := \frac{1}{2}\int_0^1 x(t)y'(t) - y(t)x'(t)\,dt.$$

Let $c > 0$. Subject to the constraint $L(s) = c$, we wish to maximize $A(s)$.

In order to solve this constrained maximization problem, we assume there exists a curve s of maximal area and of length c. And we also assume that the Lagrange Multiplier Theorem applies to this problem. That is, there exists $\lambda \in \mathbb{R}$ such that, if $r(t) = (w(t), z(t))$ is any function with $t \in [0,1]$, and if $s_p(t) = s(t) + pr(t)$ for any $p \in (-1,1)$, then

$$\frac{d}{dp}L(s_p) = \lambda\frac{d}{dp}A(s_p).$$

By investigating this equation, conclude that s is a circle. That is, there exists $c, d \in \mathbb{R}$ such that $(x-c)^2 + (y-d)^2 = \lambda^2$. Deduce the **isoperimetric inequality**: for any curve s such that $x, y \in C^1$, $s(0) = s(1)$ and such that s does not intersect itself, we have

$$A(s) \leq \frac{1}{4\pi}(L(s))^2.$$

(Hint: first consider r where $w(t) = 0$ for all $t \in [0,1]$, and then consider r where $z(t) = 0$ for all $t \in [0,1]$. Also, you may assume that s is a constant-speed parametrization, so that $\|s'(t)\|$ is constant for all $t \in [0,1]$.)

Exercise 6.7. In Exercise 2.60, we investigated the largest entropies on finite-dimensional vector spaces. (The smallest possible entropy is zero, so minimizing entropy is not so interesting.) The infinite-dimensional case is a bit different than the finite-dimensional case.

Let A be the set of all $f \colon \mathbb{R} \to [0,\infty)$ such that $\int_{-\infty}^{\infty} f(x)dx = 1$. (In probability terminology, f is a probability density function.) Maximize the entropy

$$-\int_{-\infty}^{\infty} f(x)\log f(x)\,dx$$

over the set A, subject to the constraint

$$\int_{-\infty}^{\infty} x^2 f(x)\,dx = 1.$$

(You may assume that the maximum exists.) (Hint: if $g \in A$, what functions h can we add to g so that $g + h \in A$?)

7. Appendix: Notation

Let n, m be a positive integers. Let A, B be sets contained in a universal set Ω.

$\mathbb{R}$ denotes the set of real numbers

$\mathbb{Z}$ denotes the set of integers

$\in$ means "is an element of." For example, $2 \in \mathbb{R}$ is read as "2 is an element of $\mathbb{R}$."

$\forall$ means "for all"

$\exists$ means "there exists"

$\mathbb{R}^n = \{(x_1, x_2, \ldots, x_n) \colon x_i \in \mathbb{R}\, \forall\, 1 \leq i \leq n\}$

$f \colon A \to B$ means f is a function with domain A and range B. For example,

$f \colon \mathbb{R}^2 \to \mathbb{R}$ means that f is a function with domain $\mathbb{R}^2$ and range $\mathbb{R}$

$\emptyset$ denotes the empty set

$A \subseteq B$ means $\forall\, a \in A$, we have $a \in B$, so A is contained in B

$A \smallsetminus B := \{a \in A \colon a \notin B\}$

$A^c := \Omega \smallsetminus A$, the complement of A in Ω

$A \cap B$ denotes the intersection of A and B

$A \cup B$ denotes the union of A and B

Let $a_1, \ldots, a_n$ be real numbers. Let n be a positive integer.

$$\sum_{i=1}^{n} a_i = a_1 + a_2 + \cdots + a_{n-1} + a_n.$$

$$\prod_{i=1}^{n} a_i = a_1 \cdot a_2 \cdots a_{n-1} \cdot a_n.$$

Let A be a real matrix. Let n be a positive integer. Let K be a closed and bounded subset of $\mathbb{R}^n$. Let $f \colon \mathbb{R}^n \to \mathbb{R}$ be a continuous function.

A^T denotes the transpose of A.

I denotes the $n \times n$ identity matrix.

A^{-1} denotes the inverse of A (if A^{-1} exists), so that $AA^{-1} = A^{-1}A = I$.

$\det(A)$ denotes the determinant of A.

$\min(a, b)$ denotes the minimum of a and b.

$\max(a, b)$ denotes the maximum of a and b.

$\min_{x \in K} f(x)$ denotes the minimum value of f on K.

$\max_{x \in K} f(x)$ denotes the maximum value of f on K.

Let $x, y \in \mathbb{R}^n$. We write $x = (x_1, \ldots, x_n)$, so that $x_i \in \mathbb{R}$ for all $1 \leq i \leq n$. Let $r > 0$.

$$\langle x, y\rangle = \sum_{i=1}^{n} x_i y_i \text{ , the dot product, or standard inner product, of } x \text{ and } y$$

$$\|x\| = \sqrt{\langle x, x\rangle} = (\sum_{i=1}^{n} x_i^2) \text{ , the length of the vector } x$$

$$B_r(y) = \{x \in \mathbb{R}^n \colon \ \|x - y\| \leq r\}. \text{ , the ball of radius } r \text{ centered at } y$$

$$x \geq 0 \text{ means } x_i \geq 0 \,\forall\, 1 \leq i \leq n.$$

Let A be an $n \times n$ real symmetric matrix. We write $A \geq 0$ to denote that A is positive semidefinite.

Let $f \colon \mathbb{R}^n \to \mathbb{R}$. Let $x \in \mathbb{R}^n$. Let $i \in \{1, \ldots, n\}$. Let $v \in \mathbb{R}^n$ be a vector. Let k be a positive integer.

$$\frac{\partial f}{\partial x_i}(x) = f_{x_i}(x), \text{ denotes the partial derivative of } f \text{ at } x \text{ in the } x_i\text{-direction}$$

$$Df(x) = \left(\frac{\partial f}{\partial x_1}(x), \ldots, \frac{\partial f}{\partial x_n}(x)\right)$$

$$\nabla f(x) = (Df(x))^T = \left(\frac{\partial f}{\partial x_1}(x), \ldots, \frac{\partial f}{\partial x_n}(x)\right)^T \text{ , denotes the gradient vector of } f \text{ at } x$$

$$D_v f(x) = \nabla f(x) \cdot v, \text{ denotes the derivative of } f \text{ at } x \text{ in the direction } v$$

$$D^2 f(x) = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1^2}(x) & \cdots & \frac{\partial^2 f}{\partial x_n \partial x_1}(x) \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_1 \partial x_n}(x) & \cdots & \frac{\partial^2 f}{\partial x_n^2}(x) \end{pmatrix}, \qquad \text{denotes the Hessian, of } f \text{ at } x.$$

$f \in C^k$ means that all iterated partial derivatives of f up to order k exist and are continuous.

UCLA Department of Mathematics, Los Angeles, CA 90095-1555
E-mail address: heilman@math.ucla.edu