AMS 761 Nonlinear Optimization I **FALL 2023**

Lecture 1: August 29

Lecturer: Mateo Díaz *Scribe: Ian McPherson*

1.1 Syllabus

Instructor: Mateo Díaz, **Office Hours:** Monday 4:00 - 6:00pm.

TAs: All office hours are in Wyman S425

- Kaleigh Rudge, **Office Hours:** Wednesday 10:00 - 11:20 am;
- Thabo Samakhona, **Office Hours:** Thursday 10:00-11:20 am;
- Roy Siegelman, **Office Hours:** Wednesday 7:00-8:20 pm

Resources: Check Canvas, Website, https://mateodd25.github.io/nonlinear/, Piazza for general questions.

Grading Breakdown: Four components

1. Homework: Approximately a total of 5, with one every two weeks - proof-based + coding (Python by social pressure);
2. Midterm: Take home (**October 13th - 17th**);
3. Final: Take home (**December 13th - 15th**, *subject to change*);
4. Participation

Now, the grade is computed where H, M, F are variable weights for the grades given the breakdown above, C_H, C_M, C_F, C_P respectively:

$$\max C_H H + C_M M + C_F F + C_P(100 - H - M - F)$$

$$\textbf{subject to} \quad \begin{cases} (H, M, F) \in \mathbb{R}^3 \\ H + M + F \leq 100 \\ 15 \leq H, M \\ M \leq F \\ 50 \leq M + F \leq 80 \\ 90 \leq H + M + F \end{cases}$$

1.2 Motivation

Before beginning, if the following motivations do not interest you, solving the grading rubric for your final grade might.

We are interested in solving problems of the following form:

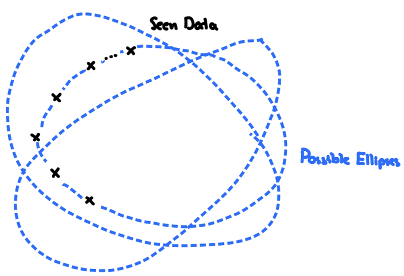
$$\min_{x \in C} f(x).$$

We focus on when $C = \mathbb{R}^d$, that is *unconstrained optimization* problems. Here are some relevant examples.

Example 1.1. Predicting Movement of a Planet

In 1801, Giuseppe Piazzi discovered a *planetoid*, a small planet orbiting in a different solar system, named Ceres. He published 22 measurements of Ceres at different snapshots in time, namely of the form $\{(x_i, y_i)\}_{i=1}^{22}$.

Euler made the assumption that the planet revolves on an ellipse. This assumption constrained what kind of trajectories would work. This looks as follows:



Mathematically, we may express such orbits in a functional form, as an equation with three coefficients:

$$\alpha x^2 + \beta y^2 + \gamma xy = 1.$$

He tried to fit the data via optimization. Then,

$$\min_{\alpha,\beta,\gamma} \sum_{i=1}^{22} (\alpha x_i^2 + \beta y_i^2 + \gamma x_i y_i - 1)^2.$$

By minimizing the L^2 error, we are finding the projection onto the space of ellipses that produces the ellipse that corresponds with the given data. This is an instance of *Least-Squares.*

Definition 1.2. Least Squares Problem

The *least squares problem* in this context is given as

$$\min_{\overline{w} \in \mathbb{R}^d} \left\| A\overline{w} - \overline{b} \right\|_2,$$

where $A \in \mathbb{R}^{n \times d}$ and $b \in \mathbb{R}^n$ are both known.

Example 1.3. Planet Movement, Continued

In the example, we would have that

$$a_i = \begin{bmatrix} x_i^2 \\ y_i^2 \\ x_i y_i \end{bmatrix}, \quad \overline{w} = \begin{bmatrix}\alpha, \beta, \gamma\end{bmatrix}, \quad b_i = 1,$$

where $a_i^\top$ would be the rows of the matrix $A \in \mathbb{R}^{n \times d}$, where $n = 22$ the sample size.

Euler's method was arguably one of the first examples of data fitting. Next, we will see a modern example that is similar in spirit.

Example 1.4. Learning

Consider having inputs $\{(\overline{x_i}, y_i)\}_{i=1}^n$, with the following goal.

Goal: Find a function f such that $f(\overline{x}_i) \approx y_i$.

An approach is to **parameterize** a family of functions,

$$\mathcal{F}_\theta := \{f_\theta | \theta \text{ is some parameter}\}.$$

Then, given this set we want to solve the following optimization problem

$$\min_{f \in \mathcal{F}} \sum_{i=1}^n \ell(f(\overline{x}_i, y_i)),$$

where ℓ is a *loss function* that measures the disagreements between the output of our learned function and the data.

Let's apply this framework to another example, learning a function that predicts whether or not you have COVID.

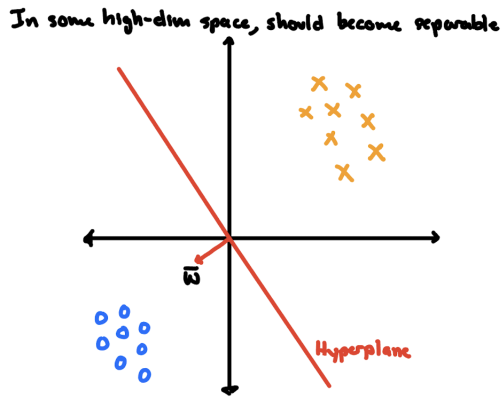
Example 1.5. Logistic Regression - Covid Example

Consider storing the quantitative descriptions of a patient x_i as a vector, where the predictors are like *age, temperature, blood pressure, heart rate,* etc. Consider $y_i \in \{0, 1\}$, corresponding with if you do or do not have covid. Since we have a binary output, this classification problem is called *logistic regression.*

The idea is simple, in this high-dimensional feature space, we should be able to separate the two classes of individuals by a **hyperplane**, where the weight vector w will give us the *normal vector* of the hyperplane. More concretely, we can do the assignments by consider the inner product with the normal vector, which geometrically corresponds with *how much you are one side of the hyperplane*

$$\begin{cases} \langle x_i, w \rangle > 0, & \text{you're sick;} \\ \langle x_i, w \rangle < 0, & \text{you're healthy.} \end{cases}$$

Of course, the closer you are to 0, the closer you are to the hyperplane. Intuitively, we have the following image:



To construct, we consider the following family of functions:

$$f_{\overline{w}}(x) = \frac{1}{1+\exp(-\overline{x}^T w)},$$

which mimics an assignment of probabilities of having COVID. Thus, we can simply state the problem as

$$\min_{\overline{w}\in\mathbb{R}^d} \sum_{i=1}^{n} l(f_{\overline{w}}(\overline{x}_i, y_i) = \min_{\overline{w}\in\mathbb{R}^d} -\left(\sum_{y_i=1} y_i \ln(f_{\overline{w}}(x_i) + \sum_{y_i=0} (1-y_i)\ln(1-f_{\overline{w}}(x_i))\right).$$

Note, that this is just the usual MLE formulation, this makes sense in this context.

For a last example, we highlight a much more complicated optimization problem: **Neural Networks**.

Example 1.6. Neural Networks

We can think of $f_{\overline{w}}(x)$, where the weights correspond with different matrices. This can be codified as a massive composition

$$W_L \circ \sigma_{L-1} \circ W_{L-1} \circ \cdots \circ \sigma_1 \circ W_1 x,$$

where σ_i are nonlinear functions and W_i are matrices. For instance, one might just use the RELU function for σ_i.

This is non-smooth, and non-convex! That's HARD.

The point of this class is that given these optimization problems, without having to think about the construction of the problem, how can we solve the problems efficiently and accurately?

1.3 Overview

The layout of the course is as follows:

1. **Geometry:** This will be focused on optimality conditions and basic convex analysis;
2. **First-order methods:** These methods only use gradient information, that is only can call $x \mapsto \nabla f(x)$. The types of functions we consider are of following regimes:
   - Smooth Functions - we have access to derivatives;
   - Convex Functions - in this regime local guarantees becomes global;
   - Non-smooth Functions - we will have to use tangent cones and such;
   - Stochastic Functions - these functions are of the from

     $$f(x) = \mathbb{E}_z F(x, z).$$

     This is especially important in data science where we only have access to the samples of the distribution, not the distribution itself.
3. **Second-order methods:** These methods also use Hessian information, that is we get $x \mapsto (\nabla f(x), \nabla^2 f(x))$. This will highlight the tension between convergence rate and computation cost. These regimes are of the flavor of:
   - Newton's Method;
   - Quasi-Newton's Method;
   - Trust Reason Methods
4. **(Time Permitting)**: Linear Programming, Conjugate Gradient Method, Composite Optimization. We will most likely only have enough time to cover one of the three problems.