EECS 127: Optimization Models in Engineering

Tyler Zhu

January 19, 2021

> *"A good stock of examples, as large as possible, is indispensable for a thorough understanding of any concept, and when I want to learn something new, I make it my first job to build one."*
>
> – *Paul Halmos.*

These are course notes for the Spring 2021 rendition of EECS 127, Optimization Models in Engineering, taught by Professor Laurent El Ghaoui.

Contents

1 Tuesday, January 19th

1.1 Introduction

In this course, we'll be talking primarily about *optimization*. A standard form of optimization is the following:

$$p^* = \min_{\boldsymbol{x}} f_0(\boldsymbol{x}) \quad \text{subject to: } f_i(\boldsymbol{x}) \leq 0,\ i = 1, \ldots, m,$$

where

- vector $\boldsymbol{x} \in \mathbb{R}^n$ is the *decision variable*;
- $f_0 : \mathbb{R}^n \to \mathbb{R}$ is the *objective* function, or *cost*;
- $f_i : \mathbb{R}^n \to \mathbb{R}$, $i = 1, \ldots, m$, represent the *constraints*;
- p^* is the *optimal value*.

Realistically, $\boldsymbol{x} = [x_1 \ \ldots \ x_n]$ represents different decisions, i.e. x_2 would be our decision at time $t = 2$. Also note that we can easily solve instead to maximize some $r(x)$ by setting $f_0(x) = -r(x)$. This above setup is known as the *standard form*.

Oftentimes we will have multiple optimal solutions to the constraint, in which case any $x^* \in \arg\min f_0(x)$ for which $f_i(x^*) \leq 0, i = 1, \ldots, m$ is satisfied acts as an optimizer.

In this class, we're not as concerned with algorithms for optimization, but more so translating problems from the real world into this language.

Example 1.1 (Least-squares regression)**.** A classic example in machine learning is when we have a given vector y and we're trying to express it as a linear function of an input vector z, i.e. data points. The goal is the solve the objective

$$\min_x \sum_{i=1}^{m} (y_i - x^\top z^{(i)})^2$$

where

- $z^{(i)} \in \mathbb{R}^n$, $i = 1, \ldots, n$ are data points;
- $y \in \mathbb{R}^m$ is a "response" vector;
- $x^\top z$ is the scalar product $z_1 x_1 + \cdots + z_n x_n$ b/w the two vectors $x, z \in \mathbb{R}^n$.

One example of constraints we could be working with are $x \geq 0$ and $x^\top \mathbb{1} = 1$, which corresponds to modeling a discrete distribution.

Example 1.2 (Support Vector Machines (SVMs))**.** In SVMs, we instead are trying to optimize a "hinge" loss, i.e.

$$\min_{x,b} \sum_{i=1}^{m} \max(0, 1 - y_i(x^\top z^{(i)} + b))$$

where

- $z^{(i)} \in \mathbb{R}^n$, $i = 1, \ldots, n$ are data points;
- $y \in \{-1, 1\}^m$ is a *binary* response vector;
- $x^\top z + b = 0$ defines a "separating hyperplane" in data space.

We could imagine that our points are colored green and red. Then at a conceptual level, we're trying to create a hyperplane that separates our data points into two different classes as clearly as possible. Once we find the best x, b, we can predict the binary output $\hat{y}$ corresponding to a new point's predicted class.

While we just gave a few machine learning examples which were problems without constraints, we can often use optimization to act on certain situations. One example is energy production. There's a **lot** of other ones.

1.2 Optimization Problems

There is more nomenclature we need to know:

- *Feasible set*, i.e. the set of possible values satisfying the constraints.
- *Unconstrained minimizer*:x_0, i.e. minimizing the cost function without constraints.
- *Optimal Point*: x^*.
- *Level sets* of objective functions, i.e. sets $\{x|R(x) = c\}$ for some c.
- *Sub-level sets*, i.e. sets $\{x|R(x) \leq c\}$ for some c.

Usually our optimal points will be some intersection with the smallest level sets and the feasible set.

Similar to neural networks, we can have an issue in optimization of local vs. global optimal points. A point z is *locally optimal* if there exists a value $R > 0$ such that z is optimal for problem

$$\min_x \ f_0(x) \text{ s.t. } f_i(x) \leq 0, \ i = 1, \ldots, m \text{ and } |x_i - z_i| \leq R, \ i = 1, \ldots, n.$$

A local minimizer x minimizes f_0, but only compared to nearby points on the feasible set. The value of the objective function at that point is *not* necessarily the (global) optimal value of the problem. Locally optimal points might be of no practical interest to the user. Visually, you could imagine that we could find ourselves in certain pits that locally seem like optima but globally are far from it.

Due to these problems (and others), often times even coming up with any minimizer can be difficult. However, there's a special class of problems called *convex problems* which have the nice property that *all* local optimums are also global optimums. Usually your objective function when plotted looks something like a bowl, i.e. there's a single point at the bottom which is the previously mentioned global optimum.

Figure 1: Convex and Non-convex functions (missing!)

Funny enough, even though most problems are non-convex, we could find a convex function that lower bounds our objective function and hope its global optimum is a decent solution to our original objective. Pushing this further, if we could find the tightest convex function which is a lower bound (think convex hulls), then its optimal point *is* the same as our original's! It looks like we just turned a hard problem into a much easier one, but we unfortunately have no idea how to find this convex-hull. Back to square one.

1.3 Course Outline

In this course, we shall deal specifically with convex optimization problems with special structure, such as:

- Least-Squares (LS)
- Linear Programs (LP)
- Convex Quadratic Program (QP)
- Second-order cone programs (SOCP)

For such specific models, very efficient solution algorithms exist with high quality implementations/software (CVX, for example). Most of the problems we come across can be categorized into one of the above structures, as we'll come to see.

A large part of our time will be spent talking about affine subspaces, and normal vectors to hyperplanes to geometrically construct feasible sets.

We'll also discuss a few non-convex problems that come up often in the real world:

- *Boolean/integer optimization*: $x_i \in \{0,1\}^n$; some variables are constrained to be Boolean or integers. Convex optimization can be used for getting (sometimes) good approximations.
- *Cardinality-constrained problems*: we seek to bound the number of non-zero elements in a vector variable. Convex optimization can be used for getting good approximations.
- *Non-linear programming*: usually non-convex problems with differentiable objective and functions. Algorithms provide only local minima.

It goes without saying that most (but not all) non-convex problems are *hard*!

For example, let's look at boolean optimization. We would like to solve $\min_{\boldsymbol{x}} c^\top \boldsymbol{x}$ subject to $A\boldsymbol{x} \leq b$ and $\boldsymbol{x} \in \{0,1\}^n$. One way of relaxing this into a problem that's easier to solve is to consider instead $\boldsymbol{x} \in [0,1]^n$, i.e. going from discrete to continuous feasible set. This is an important question because this allows us to do sparsity and feature-selection.

What this course if for:

- Learnign to model and efficiently solve problems arising in Engineering, Management, Control, Finance, ML, etc.
- Learning to prototype small- to medium- sized problems on numerical computing platforms.
- Learning basics of applied linear algebra, convex optimization.

What this course is NOT:

- A course on mathematical convex analysis.
- A course on details of optimization algorithms.

Here's a high level overview of what we're talking about:

- Linear algebra models
  - Vectors, projection theorem, matrices, symmetric matrices.
  - Linear equation, least-squares and minimum-norm problems.
  - Singular value decomposition (SVD), PCA, and related optimization problems.

- Convex optimization models
  - Convex sets, convex functions, convex problems.
  - Optimality conditions, duality.
  - Special convex models: LP, QP, SOCP.
- Applications
  - Machine Learning.
  - Control.
  - Finance.
  - Engineering desisgn.

There will only be six homeworks in this course.