Introduction: Why Optimization?

Ryan Tibshirani

Convex Optimization 10-725

Course setup

Welcome to our course on Convex Optimization, with a focus on its ties to Machine Learning and Statistics!

Basic adminstrative details:

- Instructor: Ryan Tibshirani
- Teaching assistants: Kartik Gupta, Biswajit Paria, Pratik Patil, Akash Umakantha, Po-Wei Wang, Wayne Zhao
- Course website: `http://www.stat.cmu.edu/~ryantibs/convexopt/`
- We will use Piazza for announcements and discussions
- We will Canvas just as a gradebook

Prerequisites: no formal ones, but class will be fairly fast paced

Assume working knowledge of/proficiency with:

- Real analysis, calculus, linear algebra
- Core problems in Machine Learning & Statistics
- Programming (R, Python, Matlab, your choice ...)
- Data structures, computational complexity
- Formal mathematical thinking

If you fall short on any one of these things, it's certainly possible to catch up; but don't hesitate to talk to us

Evaluation:

- 5 homeworks
- 2 little tests
- 1 project
- Many easy quizzes

Project: something useful/interesting with optimization. Groups of 2 or 3, milestones throughout the semester, details to come

Quizzes: due at midnight the day of each lecture. Should be very easy if you've attended lecture ...

Scribing: sign up to scribe one lecture per semester, on the course website (multiple scribes per lecture). Can bump up your grade in boundary cases

Lecture videos: see links on course website. These are supposed to be helpful supplements, not replacements! Best to attend lectures

Auditors: welcome, please audit rather than just sitting in

Heads up: class will not be easy, but should be well worth it ... !

Optimization in Machine Learning and Statistics

Optimization problems underlie nearly everything we do in Machine Learning and Statistics. In many courses, you learn how to:

translate into $P \; : \; \min_{x \in D} \; f(x)$

Conceptual idea *Optimization problem*

Examples of this? Examples of the contrary?

This course: how to solve P, and why this is a good skill to have

Presumably, other people have already figured out how to solve

$$P \ : \ \min_{x \in D} \ f(x)$$

So why bother? Many reasons. Here's three:

1. Different algorithms can perform better or worse for different problems P (sometimes drastically so)
2. Studying P through an optimization lens can actually give you a deeper understanding of the statistical procedure
3. Knowledge of optimization can actually help you create a new P that is even more interesting/useful

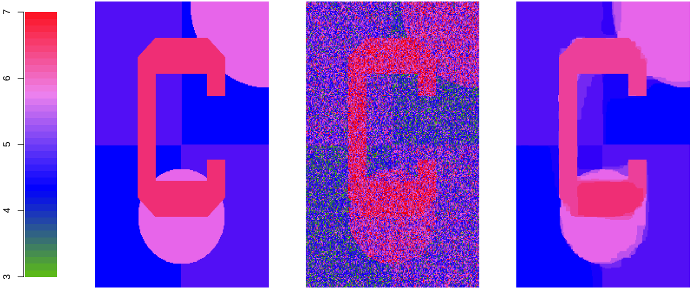
Optimization moves quickly as a field. But there is still much room for progress, especially its intersection with ML and Stats

Example: algorithms for the 2d fused lasso

The 2d fused lasso or 2d total variation denoising problem:

$$\min_{\theta} \ \frac{1}{2}\sum_{i=1}^{n}(y_i-\theta_i)^2 + \lambda \sum_{(i,j)\in E} |\theta_i - \theta_j|$$

This fits a piecewise constant function over an image, given data y_i, $i = 1, \ldots, n$ at pixels. Here $\lambda \geq 0$ is a tuning parameter

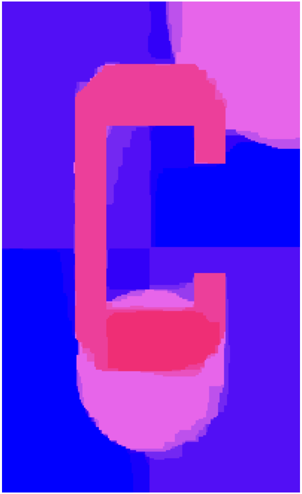


True image

Data

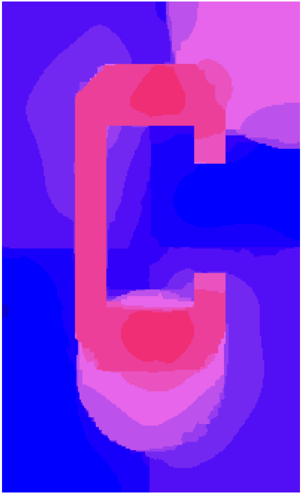
Solution

Our problem: $$\min_{\theta} \frac{1}{2}\sum_{i=1}^{n}(y_i-\theta_i)^2+\lambda\sum_{(i,j)\in E}|\theta_i-\theta_j|$$



Specialized ADMM, 20 iterations

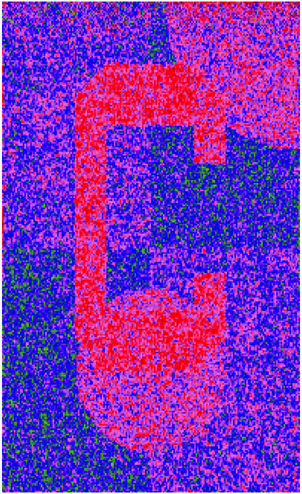
Our problem: $$\min_{\theta} \frac{1}{2}\sum_{i=1}^{n}(y_i-\theta_i)^2+\lambda\sum_{(i,j)\in E}|\theta_i-\theta_j|$$



Specialized ADMM, 20 iterations

Proximal gradient descent, 1000 iterations

Our problem: $$\min_{\theta} \frac{1}{2}\sum_{i=1}^{n}(y_i-\theta_i)^2+\lambda\sum_{(i,j)\in E}|\theta_i-\theta_j|$$

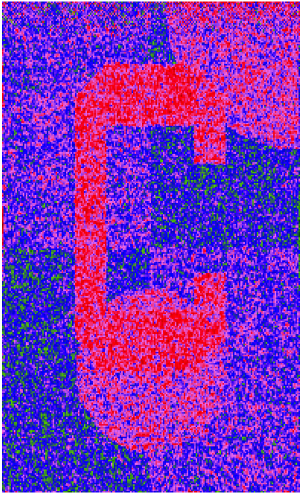


Specialized ADMM, 20 iterations

Proximal gradient descent, 1000 iterations

Coordinate descent, 10K cycles

Our problem: $$\min_{\theta} \frac{1}{2}\sum_{i=1}^{n}(y_i-\theta_i)^2+\lambda\sum_{(i,j)\in E}|\theta_i-\theta_j|$$



Specialized ADMM, 20 iterations

Proximal gradient descent, 1000 iterations

Coordinate descent, 10K cycles

(Last two from the dual)

What's the message here?

So what's the right conclusion here?

Is the alternating direction method of multipliers (ADMM) method simply a better method than proximal gradient descent, coordinate descent? ... No

In fact, different algorithms will perform better or worse in different situations. We'll learn details throughout the course

In the 2d fused lasso problem:

- Special ADMM: fast (structured subproblems)
- Proximal gradient: slow (poor conditioning)
- Coordinate descent: slow (large active set)

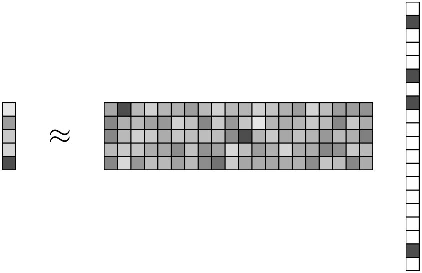
Example: sparse linear modeling

Given $y \in \mathbb{R}^n$ and a matrix $X \in \mathbb{R}^{n \times p}$, with $p \gg n$. Suppose that we know that

$$y \approx X\beta^*$$

for some unknown coefficient vector $\beta^* \in \mathbb{R}^p$. Can we generically solve for β^*? ... No!

But if β^* is known to be sparse (i.e., have many zero entries), then it's a whole different story



There are many different approaches for estimating β^*. A popular approach is to solve the lasso problem:

$$\min_{\beta \in \mathbb{R}^p} \; \frac{1}{2}\|y - X\beta\|_2^2 + \lambda\|\beta\|_1$$

Here $\lambda \geq 0$ is a tuning parameter, and $\|\beta\|_1 = \sum_{i=1}^p |\beta_i|$ denotes the ℓ_1 norm of β

There are numerous algorithms for computing a lasso solution (in fact, it can be cast as a quadratic program)

Furthermore, some key statistical insights can be derived from the Karush-Kuhn-Tucker (KKT) optimality conditions for the lasso

Lasso support recovery

The KKT conditions for the lasso problem are

$$X^T(y - X\beta) = \lambda s$$

$$s_j \in \begin{cases} \{+1\} & \beta_j > 0 \\ \{-1\} & \beta_j < 0 \\ [-1, 1] & \beta_j = 0 \end{cases}, \quad \text{for } j = 1, \ldots, p$$

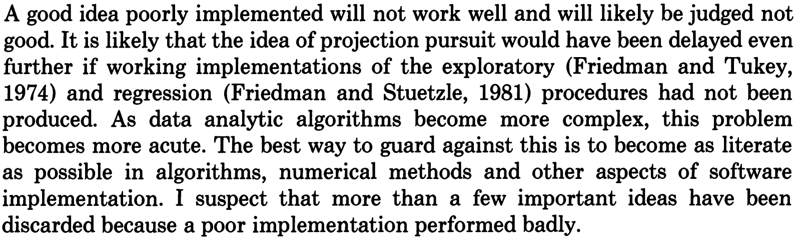
We call s a subgradient of the ℓ_1 norm at β, denoted $s \in \partial\|\beta\|_1$

Under favorable conditions (low correlations in X, large nonzeros in β^*), can show that lasso solution has same support as β^*

Proof idea: plug in (shrunken version of) β^* into KKT conditions, and show that they are satisfied with high probability (primal-dual witness method of Wainwright 2009)

Widsom from Friedman (1985)

From Jerry Friedman's discussion of Peter Huber's 1985 projection pursuit paper, in Annals of Statistics:



> A good idea poorly implemented will not work well and will likely be judged not good. It is likely that the idea of projection pursuit would have been delayed even further if working implementations of the exploratory (Friedman and Tukey, 1974) and regression (Friedman and Stuetzle, 1981) procedures had not been produced. As data analytic algorithms become more complex, this problem becomes more acute. The best way to guard against this is to become as literate as possible in algorithms, numerical methods and other aspects of software implementation. I suspect that more than a few important ideas have been discarded because a poor implementation performed badly.

Arguably, less true today due to the advent of disciplined convex programming? Maybe, but it still rings true in large part ...

Central concept: convexity

Historically, linear programs were the focus in optimization

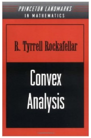
Initially, it was thought that the important distinction was between linear and nonlinear optimization problems. But some nonlinear problems turned out to be much harder than others ...

Now it is widely recognized that the right distinction is between convex and nonconvex problems

Your supplementary textbooks for the course:

Boyd and Vandenberghe (2004)

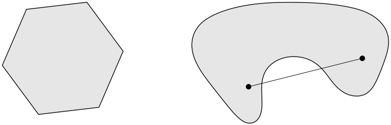
, Rockafellar (1970)



Convex sets and functions

Convex set: $C \subseteq \mathbb{R}^n$ such that

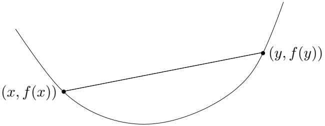
$$x, y \in C \implies tx + (1-t)y \in C \text{ for all } 0 \le t \le 1$$



Convex function: $f : \mathbb{R}^n \to \mathbb{R}$ such that $\mathrm{dom}(f) \subseteq \mathbb{R}^n$ convex, and

$$f(tx + (1-t)y) \le tf(x) + (1-t)f(y) \text{ for all } 0 \le t \le 1$$

and all $x, y \in \mathrm{dom}(f)$



Convex optimization problems

Optimization problem:

$$\begin{aligned} \min_{x\in D} \quad & f(x) \\ \text{subject to} \quad & g_i(x) \le 0,\ i = 1, \dots m \\ & h_j(x) = 0,\ j = 1, \dots r \end{aligned}$$

Here $D = \mathrm{dom}(f) \cap \bigcap_{i=1}^m \mathrm{dom}(g_i) \cap \bigcap_{j=1}^p \mathrm{dom}(h_j)$, common domain of all the functions

This is a convex optimization problem provided the functions f and $g_i, i = 1, \dots m$ are convex, and $h_j, j = 1, \dots p$ are affine:

$$h_j(x) = a_j^T x + b_j, \quad j = 1, \dots p$$

Local minima are global minima

For convex optimization problems, local minima are global minima

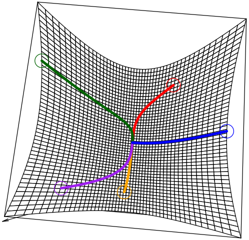
Formally, if x is feasible—$x \in D$, and satisfies all constraints—and minimizes f in a local neighborhood,

$$f(x) \le f(y) \text{ for all feasible } y, \ \|x - y\|_2 \le \rho,$$

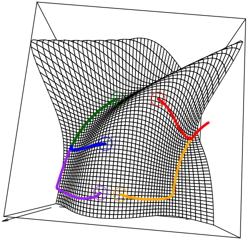
then

$$f(x) \le f(y) \text{ for all feasible } y$$

This is a very useful fact and will save us a lot of trouble!



Convex



Nonconvex