Lecture 17: Robust optimization: (aka robust mean estimation is all you need)

November 24, 2025

1 Introduction

In this lecture we'll see how ideas from what we've been discussing so far for robust mean estimation can be used in an almost black-box fashion to obtain algorithms for outlier robust optimization, regression, classification, and beyond. This is also partially why robust mean estimation is such an important primitive in this setting. We first begin by recalling the standard stochastic optimization setting.

In its most general form, stochastic optimization can be phrased as follows. There is a distribution $\mathcal{D}$ over random functions $\mathbf{f} : \mathbb{R}^d \to \mathbb{R}$ (we will use boldface to denote that the function is random). We are given a dataset of draws from this distribution $\mathbf{f}_1, \dots, \mathbf{f}_n$, and our goal is to find a point $x \in \mathbb{R}^d$ which approximately minimizes the expected function

$$F(x) = \mathop{\mathbb{E}}_{\mathbf{f} \sim \mathcal{D}}[\mathbf{f}(x)] \, .$$

Example 1.1. A standard setting in which this comes up is for regression and/or classification. Here, we are given a labeled dataset $(X_1, y_1), \dots, (X_n, y_n)$, where $X_i \in \mathbb{R}^d$ and $y \in \mathbb{R}$ (or $y \in \{0, 1\}$ for classification) are drawn from some joint distribution D, and some loss function $\ell_\theta(x, y)$ parameterized by θ, and our goal is to find a θ which minimizes the expected loss over the dataset:

$$L(\theta) = \mathop{\mathbb{E}}_{(X,y) \sim D}[\ell_\theta(X, y)] \, .$$

Any such problem can easily be seen as a special form of stochastic optimization in the underlying parameters θ, where $\mathbf{f}(\theta) = \ell_\theta(X, y)$, for $X, y \sim D$. This formulation captures many natural supervised statistical learning problems:

- For least-squares regression, we have that

$$\ell_\theta(X, y) = (\langle X, \theta \rangle - y)^2 \, .$$

- For logistic regression, we have that

$$\ell_\theta(X, y) = -\Big(y \log \sigma(\theta^\top X) + (1 - y) \log \big(1 - \sigma(\theta^\top X)\big)\Big),$$

  where $\sigma(z) = \frac{1}{1+e^{-z}}$. As a simple generalization of this, naturally this model also captures more complex (non-convex) versions of this problem such as those which naturally occur in the training of modern-day neural networks.

We note that there are many notions of approximate minimizer that one might wish for, depending on the exact setting of the problem. In this class, we'll consider a fairly generic notion of finding an approximate first-order minimizer:

Definition 1.1. We say a point $x \in \mathbb{R}^d$ is an ε-approximate first-order minimizer (or just ε-approximate minimizer for short) of f if $\|\nabla f(x)\|_2 \le \varepsilon$.

We note that under standard assumptions (e.g. strong convexity, smoothness), this notion also implies closeness in function value and/or parameter distance to the true global minimizer. A full discussion of the appropriate notions would be too much to cover in a single class; an interested reader can see e.g. [1] for a more detailed breakdown.

We can how state the robust version of this problem:

Definition 1.2 (Robust stochastic optimization)**.** Let $\mathcal{D}$ be a distribution over functions $\mathbf{f} : \mathbb{R}^d$ to $\mathbb{R}$, and let $\varepsilon > 0$ be sufficiently small. Let $S = \{\mathbf{f}_1, \dots, \mathbf{f}_n\}$ be an ε-corrupted set of samples from $\mathcal{D}$. Given S, find a point $x \in \mathbb{R}^d$ so that x is an approximate minimizer of $F(x) = \mathbb{E}_{\mathbf{f} \sim D}[\mathbf{f}(x)]$.

The remainder of this lecture will be dedicated to two different algorithmic approaches for solving this problem.

2 Approach 1: Robust gradient descent

The first approach we'll consider is arguably the most natural approach to try, but as we'll see, may not be as practical in some applications as the second approach. This is based on a simplification of the exposition in [2] and to a lesser extent, the appendix in [3].

A standard way to solve the problem of stochastic optimization is to find the empirical risk minimizer (ERM):

Definition 2.1 (Empirical risk minimizer)**.** Let $\mathbf{f}_1, \dots, \mathbf{f}_n$ be a dataset of functions, and let $\mathbf{F}(x) = \frac{1}{n}\sum_{i=1}^n \mathbf{f}_i(x)$. We say that x is an *empirical risk minimizer* if it is a minimizer of $\mathbf{F}$.

It is well-known from generalization theory that for many natural problems, such as logistic and linear regression, the ERM will be close to the true population minimizer, under suitable assumptions on the data distribution.

So, it remains to find the ERM. Given this function, a natural first pass to find a solution is to use gradient descent: initialize x_1, and for all $t = 1, \dots, T$, we let

$$x_{t+1} = x_t - \eta_t \nabla \mathbf{F}(x_t) \,, \tag{1}$$

for some step-size parameter η_t.

We recall a simple setting where gradient descent can be easily analyzed: namely, the setting where $\mathbf{F}$ is *smooth and strongly convex*. Recall that $\mathbf{F}$ is *ℓ-strongly convex* if for all $x, y \in \mathbb{R}^d$, we have

$$\mathbf{F}(y) \ge \mathbf{F}(x) + \langle \nabla \mathbf{F}(x), y - x \rangle + \frac{\ell}{2}\|y - x\|_2^2 \,.$$

This is intuitively saying that the value of the function as we move away from x is growing faster than the standard convex bound plus a *quadratic* term. We also say that $\mathbf{F}$ is L-smooth if $x, y \in \mathbb{R}^d$, we have that

$$\|\nabla \mathbf{F}(x) - \nabla \mathbf{F}(y)\|_2^2 \le \frac{L}{2}\|x - y\|_2^2 \,.$$

Note that in particular, if we let $x = x_t$ and $y = x^*$ where x^* is a global minimizer of the function, these equations simplify to the following:

$$-\langle \nabla \mathbf{F}(x_t), x_t - x^* \rangle \le -\frac{\ell}{2}\|x_t - x^*\|_2^2$$
$$\|\nabla \mathbf{F}(x_t)\|_2^2 \le \frac{L}{2}\|x_t - x^*\|_2^2 \,.$$

In particular, this implies that

$$\begin{aligned}\|x_{t+1} - x^*\|_2^2 &= \|x_t - \eta\nabla\mathbf{F}(x_t) - x^*\|_2^2 \\ &= \|x_t - x^*\|_2^2 - 2\eta\langle\nabla\mathbf{F}(x_t), x_t - x^*\rangle + \eta^2\|\mathbf{F}(x_t)\|_2^2 \\ &\le \|x_t - x^*\|_2^2 - \eta\cdot\ell\|x_t - x^*\|_2^2 + \eta^2\cdot\frac{L}{2}\|x_t - x^*\|_2^2 ,\end{aligned}$$

so in particular, if we set $\eta_t = \frac{\ell}{L}$, we have that

$$\|x_{t+1} - x^*\|_2^2 \le \left(1 - \frac{\ell^2}{2L}\right)\|x_t - x^*\|_2^2 ,$$

and so we get exponentially fast convergence to x^*. This is known as a *linear rate* in the literature (see e.g. [1]).

Gradient descent with approximate gradients. We now ask: suppose at iteration t, instead of applying the update (1), we apply the update

$$x_{t+1} = x_t - \eta g_t , \quad \text{where} \quad \|g_t - \nabla\mathbf{F}(x_t)\|_2 \le \alpha , \tag{2}$$

for some parameter α. We call these g_t a set of α-*approximate* gradients.

Now, the only change is that

$$\|x_{t+1} - x^*\|_2 \le \sqrt{1 - \frac{\ell^2}{2L}}\cdot\|x_t - x^*\|_2 + \alpha ,$$

so in particular, as long as $\|x_t - x\|_2 \gg \alpha$, then we still experience geometric decay. Thus, we have shown the following:

Theorem 2.1. *Let $\mathbf{F}$ be ℓ-strongly convex and L-smooth. Suppose we are given α-approximate gradients. Then in $O\left(\frac{L}{\ell^2}\cdot\log(1/\alpha)\right)$ iterations, we can find a point x so that $\|x - x^*\|_2 \le O(\alpha)$.*

Robust gradient descent We now come back to the setting of robust stochastic optimization. By the above, to run gradient descent, it suffices to find an approximate gradient at every iterate. At some point x_t, let $z_i = \nabla f_i(x_t)$ for $i = 1, \ldots, n$. We need to approximately find $\nabla\mathbf{F}(x_t)$ given an ε-corruption of $\nabla f_i(x_t)$, where each $f_i \sim D$. The observation is that this is *exactly a robust mean estimation problem*! Let $Z_i = \nabla f_i(x_t)$. Then, if Z_i is not corrupted, then $\mathbb{E}[Z_i] = \mathbf{F}(x_t)$. So as long as the uncorrupted Z_i are well-behaved, we can black-box apply an appropriate robust mean estimation algorithm!

There are several details that should be worked out carefully that we will neglect here. In particular, the regularity conditions that we need are a little bit intricate, since we will either need to draw new corrupted samples at every iteration, or we will need to re-use samples, in which case we will want that the set of gradients is good at all $x \in \mathbb{R}^d$ (technically, we only need this at the x_t we encounter).

3 Approach 2: Filtering at ERMs

The second approach is based on the algorithm presented in [3]. We first recall the basic promise of *applying a single iteration of the* filtering algorithm: given an ε-corrupted set of points $S = S_{\text{good}} \cup S_{\text{bad}} \setminus S_r$ where the uncorrupted set of points has mean μ_g and are sufficiently nice, then in a single iteration, either the filter will:

- Certify that the empirical mean of this set of points $\widehat{\mu}$ is close to the true mean μ, or
- Remove more bad points than good points.

The algorithm we will present will crucially use this guarantee of the filtering algorithm. This algorithm will proceed as follows: until termination, repeat the following procedure. Let S be initially set to $\{f_1, \ldots, f_n\}$. Then:

- Using any algorithm find an first-order ERM of the current set of S, i.e. a point x so that
$$\left\| \frac{1}{|S|} \sum_{i \in S} \nabla \mathbf{f}_i(x) \right\|_2 = 0 \, .$$
- Let $Z_i = \nabla \mathbf{f}_i(x)$, and run a single iteration of filtering on $\{Z_i\}_{i \in S}$.
- If the filter certifies that the empirical mean of the $\{Z_i\}_{i \in S}$ is close to the true mean, output x and terminate.
- Otherwise, let S' be the filtered dataset, and repeat.

The proof of correctness of this algorithm, under the appropriate regularity conditions on the gradients, is again almost trivial: assuming that filtering succeeds in every iteration, then if the filter removes points, then it removes more bad points than good points, and so the remaining dataset is still an ε-corrupted set of points from D.

On the other hand, if the filter certifies that the empirical mean is close to the true mean, this means that if we let $Z_g = \nabla \mathbf{F}(x) = \mathbb{E}_{\mathbf{f} \sim D}[\nabla \mathbf{f}(x)]$ be the true mean of the good points, then the filter certifies that
$$\left\| \frac{1}{|S|} \sum_{i \in S} Z_i - \nabla \mathbf{F}(x) \right\|_2 \leq f(\varepsilon) \, .$$
But we are at an ERM! Thus $\frac{1}{|S|} \sum_{i \in S} Z_i = 0$, and so $\|\mathbf{F}(x)\|_2 \leq f(\varepsilon)$, and thus we are at an approximate first-order minimizer.

Comparison to robust gradient descent There are a couple of comparisons between this method and the method presented previously. The main downside of the previous method is that it is only tuned to full-batch gradient descent, and hence is typically quite slow in practice. This method can also be somewhat slow, as in theory, one may require many iterations of filtering before we terminate (one can show a bound of $O(d)$ iterations suffice after some preprocessing). But in practice, it seems that this typically terminates in few iterations. To the best of my knowledge, it is still open whether or not one can obtain an approximate minimizer using only polylogarithmically calls to the ERM.

There is also another more subtle distinction, related to the regularity conditions we kind of brushed off. The robust gradient algorithm only requires that for all $x \in \mathbb{R}^d$, there exists some "good" subset of S so that the gradient of this subset is representative. Notably, this subset need not be the same for different x. However, one can verify that the second method fundamentally requires that the good subset is shared across all S. While this distinction does not seem to be very noticeable in practice, this leads to the statistical guarantees of the former algorithm to be slightly better (by logarithmic factors), at least in some settings.

References

[1] Sébastien Bubeck et al. Convex optimization: Algorithms and complexity. *Foundations and Trends® in Machine Learning*, 8(3-4):231–357, 2015.

[2] Adarsh Prasad, Arun Sai Suggala, Sivaraman Balakrishnan, and Pradeep Ravikumar. Robust estimation via robust gradient estimation. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 82(3):601–627, 2020.

[3] Ilias Diakonikolas, Gautam Kamath, Daniel Kane, Jerry Li, Jacob Steinhardt, and Alistair Stewart. Sever: A robust meta-algorithm for stochastic optimization. In *International Conference on Machine Learning*, pages 1596–1606, 2019.