Course Notes for EE227C (Spring 2018): Convex Optimization and Approximation

Instructor: Moritz Hardt
Email: hardt+ee227c@berkeley.edu

Graduate Instructor: Max Simchowitz
Email: msimchow+ee227c@berkeley.edu

February 27, 2018

10 Lecture 10: Stochastic Optimization

In this lecture, we examine the stochastic gradient descent method and different contexts of its use, including Empirical Risk Minimization and Mirror Descent.

10.1 The Stochastic Gradient Method

Following Robbins-Monro [RM51], we define the stochastic approximation method as follows.

Definition 10.1. (Stochastic Gradient Method) We want to minimize functions $f\colon$ of the following form. For all $x \in \Omega$:

$$f(x) = \mathop{\mathbb{E}}_{Z \sim \mathcal{D}} g(x, Z)$$
$$f(x) = \frac{1}{n}\sum_{i=1}^{n} f_i(x)$$

To do so, we define the following update rule, where $x_0 \in \Omega$:

$$\forall t \geqslant 0,\ x_{t+1} = x_t - \eta_t \nabla f_{i_t}(x_t)$$

where $i_t \in \{1, \ldots, n\}$ is either selected at random at each step, or cycled through a random permutation of $\{1, \ldots, n\}$.

Fact 10.2. *For all $t \geqslant 0$ and $x \in \Omega$,*

$$\mathbb{E}\, \nabla f_{i_t}(x) = \nabla f(x) \tag{1}$$

10.1.1 Sanity check

Let us check that on a simple problem, stochastic gradient descent yields the optimum. Let $p_1, \ldots, p_m \in \mathbb{R}^n$, and define $f \colon \mathbb{R}^n \to \mathbb{R}_+$:

$$\forall x \in \mathbb{R}^n,\ f(x) = \frac{1}{2m} \sum_{i=1}^{m} \|x - p_i\|_2^2$$

Note that for all $i \in \{1, \ldots, m\}$:

$$\forall x \in \mathbb{R}^n,\ f_i(x) = \frac{1}{2}\|x - p_i\|_2^2$$
$$\forall x \in \mathbb{R}^n,\ \nabla f_i(x) = x - p_i$$

and therefore

$$x^* = \arg\min_{x \in \mathbb{R}^d} f(x) = \frac{1}{m} \sum_{i=1}^{m} p_i$$

Now, run SGM with $\eta_t = \frac{1}{t}$ in cyclic order i.e. $i_t = t$ and $x_0 = 0$:

$$\begin{aligned}
x_0 &= 0 \\
x_1 &= 0 - \frac{1}{1}(0 - p_1) = p_1 \\
x_2 &= p_1 - \frac{1}{2}(p_1 - p_2) = \frac{p_1 + p_2}{2} \\
&\vdots \\
x_n &= \frac{1}{m} \sum_{i=1}^{m} p_i = x^*
\end{aligned}$$

10.2 Application: the Perceptron Algorithm

The New York Times wrote in 1958 that the Perceptron [Ros58] was "The embryo of an electronic computer that [the Navy] expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence."

Definition 10.3. (Perceptron) Given a set of datapoints and labels $\{(x_1, y_1), \ldots, (x_m, y_m) \in \mathbb{R}^n \times \{-1, 1\}\}$ and $w_0 \in \mathbb{R}^n$, the Perceptron is the following algorithm. For $(i_t)_t$ selected uniformly at random:

$$\forall t \geqslant 0,\ w_{t+1} = w_t(1 - \gamma) + \eta \begin{cases} y_{i_t} x_{i_t} & \text{if } y_{i_t}\langle w_t, x_{i_t}\rangle < 1 \\ 0 & \text{otherwise} \end{cases}$$

Reversing the problem, the Perceptron is equivalent to running the SGM on the Support Vector Machine (SVM) objective function.

Definition 10.4. (SVM) Given a set of datapoints and labels $\{(x_1, y_1), \ldots, (x_m, y_m) \in \mathbb{R}^n \times \{-1, 1\}\}$, the SVM objective function is:

$$f(w) = \frac{1}{n} \sum_{i=1}^{m} \max(1 - \langle w, x_i \rangle,\ 0) + \lambda \|w\|_2^2$$

NB: $u \mapsto \max(1 - u,\ 0)$ is known as the Hinge Loss. $\lambda \|w\|_2^2$ is known as the regularization term.

10.3 Empirical Risk Optimization

We have two spaces of objects $\mathcal{X}$ and $\mathcal{Y}$ and want to learn a function $h : x \to y$ which outputs an object $y \in \mathcal{Y}$, given $x \in \mathcal{X}$. Assume there is a joint distribution $\mathcal{D} : \mathcal{X} \times \mathcal{Y}$ and the training set consists of m instances $(x_1, y_1), \ldots, (x_m, y_m)$ drawn i.i.d. from $\mathcal{D}$.

We also define a non-negative real-valued loss function $L(y', y)$ to measure the difference between the prediction y' and the true outcome y.

Definition 10.5. The risk associated with h(x) is defined as the expectation of the loss function:

$$R[h] = \mathbb{E}_{\mathcal{X} \times \mathcal{Y} \in \mathcal{D}} L(h(x), y)$$

The ultimate goal of a learning algorithm is to find h^* among a class of functions $\mathcal{H}$ that minimizes $R[h]$:

$$h^* = \arg\min_{h \in \mathcal{H}} R[h]$$

In general, the risk $R[h]$ can not be computed because the joint distribution is unknown. Therefore,

Definition 10.6. we can compute empirical risk, by averaging the loss function of the training set:

$$R_n[h] = \frac{1}{n} \sum_{i=1}^{n} L(h(x_i), y_i)$$

And the goal is to find $h^* = \arg\min_{h \in \mathcal{H}} R_n[h]$.

10.4 Online Learning

Taking advice from experts

Assume we have access to predictions of n experts. Let these predictions at time t be $f_{1,t}, \ldots, f_{n,t}$.

At each step $t : t = 1, \ldots, T$:

- we observe $f_{1,t}, \ldots, f_{n,t}$ from n experts.

- we randomly choose one expert $I_t \in \{1, \dots, n\}$
- we receive feedback $f_t \in [-1, 1]^n$ and incur loss $f_{t,I_t} \in [-1, 1]$

Let $w_t \in \Delta_n = \{w \in \mathbb{R}^n : w \geqslant 0, \|w\| = 1\}$. At each step t, we draw I_t randomly and independently as $I_t \sim w_t$. More explicitly, $\forall i \in \{1, \dots, n\}, \ \mathbb{P}[I_t = i] = w_t[i]$.

Then we define the expected loss at step t:

$$\mathbb{E}_{I_t \sim w_t} f_{t,I_t} = \sum_{i=1}^{n} \mathbb{P}[I_t = i] f_{t,I_t} = \langle w_t, f_t \rangle$$

The update rule is:

$$\begin{aligned} \forall i, \ v_t^{(i)} &= w_{t-1}^{(i)} e^{-\eta f_t(i)} \\ w_t &= \Pi_\Delta(v_t) \end{aligned}$$

Then we measure our regret:

$$R = \sum_{t=1}^{T} \langle w_t, f_t \rangle - min_{w \in \Delta_n} \sum_{t=1}^{T} \langle w, f_t \rangle$$

where w is the best distribution from hindsight. The question is *how do we bound the regret?*

10.4.1 Mirror Descent

Recall that mirror descent requires a mirror map $\phi : \Omega \to R$ over a domain $\Omega \in \mathbb{R}^n$ where ϕ is strongly convex and continuously differentiable.

The associated projection:

$$\Pi_\Omega^\phi(y) = \arg\min_{x \in \Omega} \mathcal{D}_\phi(x, y)$$

where $\mathcal{D}_\phi(x, y)$ is Bregman distance.

Definition 10.7. Bregman Distance measures how good the first order approximation of function ϕ is:

$$\mathcal{D}_\phi(x, y) = \phi(x) - \phi(y) - \nabla\phi(y)^\intercal(x - y)$$

The mirror descent update rule is:

$$\begin{aligned} \nabla\phi(y_{t+1}) &= \nabla\phi(x_t) - \eta g_t \\ x_{t+1} &= \Pi_\Omega^\phi(y_{t+1}) \end{aligned}$$

where $g_t \in \partial f(x_t)$

Theorem 10.8. *let $\|\cdot\|$ be arbitrary norm and suppose that ϕ is α-strongly convex **w.r.t.** $\|\cdot\|$ on Ω. Suppose that f_t is L-lipschitz **w.r.t.** $\|\cdot\|$, we have:*

$$\frac{1}{T}\sum_{t=1}^{T} f_t(x_t) \leqslant \frac{\mathcal{D}_\phi(x^*, x_0)}{T\eta} + \eta\frac{L^2}{2\alpha}$$

NB: This was proved in homework 1.

10.5 Apply Multiplicative weights to Mirror Descent

Multiplicative weights are an instance of the Mirror Descent where $\Phi : w \mapsto \sum_{i=1}^{m} w_i \log(w_i)$.

Note that $\nabla\Phi : w \mapsto 1 + \log(w)$ where the log is taken elementwise. The update rule in Mirror Descent becomes:

$$\nabla\Phi(v_{t+1}) = \nabla\Phi(w_t) - \eta_t f_t$$
$$\implies v_{t+1} = w_t e^{-\eta_t f_t}$$

Now comes the projection step. The Bregman divergence is, for all $(x, y) \in \Omega^2$:

$$D_\Phi(x, y) = \Phi(x) - \Phi(y) - \nabla\Phi(y)^T(x - y)$$

yielding for all y:

$$\Pi_\Omega^\Phi(y) = \arg\min_{x\in\Omega} D_\Phi(x, y)$$

References

[RM51] H. Robbins and S. Monro. A stochastic approximation method. *Annals of Mathematical Statistics*, 22:400–407, 1951.

[Ros58] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, pages 65–386, 1958.