Khoury College of Computer Sciences
Northeastern University
CS 7870: Seminar in Theoretical Computer Science

Fall 2023
28 November 2023
Scribe: Mohammad Saneian

Lecture 20 Outline:

- Online Gradient Descent
- Stochastic Gradient Descent
- Online Algorithms and Competitive Analysis

These scribe notes cover the topics of online gradient descent and the ski rental problem. The material is partly based on [BN+09] and Nagarajan lecture notes for online optimization [Nag21].

1 Online gradient descent

In the online gradient descent problem, we have a convex body K and a sequence of convex functions f_t which is unknown to us. Our goal is to play $x_t \in K$ such that we minimize $\sum f_t(x_t)$. We define D to be the diameter of the convex body K. A possible algorithm for online gradient descent is as follows:

Algorithm 1: Online Gradient Descent

input: f, K, and set of step sizes $\{\eta_t\}$
Set $x_1 \in K$
for $t = 1$ *to* T **do**
  Play x_t and get loss $l_t = f_t(x_t)$
  $y_{t+1} = x_t - \nabla f_t(x_t)\eta_t$
  $x_{t+1} = \prod_K y_{t+1}$
return $\sum_{t=1}^{T} l_t$

Here $\prod_K$ means the point that is the projection of y_t in K in case that y_t lies outside K. The main difference with gradient descent is the step sizes η_t and how you calculate the gradients. We define G to be the upper bound for the gradients. In the previous lecture, we proved the following theorem.

Theorem 1. If we run Algorithm 1 for online gradient descent with $\eta_t = \frac{D}{G\sqrt{T}}$, it attains the regret:

$$\text{regret} = \sum_t f_t(x_t) - \min_{x^*} \sum_t f_t(x^*) \leq \frac{3GD\sqrt{T}}{2}$$

Note that proving a bound of $O(GDT)$ on the regret is easy since the functions f_t are convex and the diameter of K is D and the gradient is upper bounded by G we have

$$f_t(x_t) - f_t(x^*) \leq GD.$$

Summing over all t for f_t we get that regret $\leq GDT$. Now we prove that $\Omega(GD\sqrt{T})$ regret is unavoidable for this problem, establishing that in the worst-case, online gradient descent has asymptotically optimal regret in terms of the parameters G, D, and T.

Theorem 2. For online Gradient Descent $\Omega(GD\sqrt{T})$ loss is unavoidable.

Proof. To prove this, we give an example of a convex body and convex functions such that this loss is unavoidable. Let $K = [-1, 1]^n$ that is the set of points inside a hypercube i.e.

$$K = \{(x_1, x_2, ..., x_n)\} : -1 \leq x_i \leq 1, \forall i$$

And also let f be the set of functions on vertices of K such that:

$$f_v = v^T \cdot x, v \in \{-1, 1\}^n$$

Let us first calculate D and G. We can see that $D = \sqrt{4 + 4 + ... + 4} = 2\sqrt{n}$ which is the distance between the point $(1, 1, \ldots, 1)$ and the point $(-1, -1, \ldots, -1)$. We know that $\nabla f_v(x) = v$ so $||\nabla f_v(x)|| = \sqrt{n}$ so $G = \sqrt{n}$.

Let the sequence of f_t be the sequence where each time we pick one vertex v_t from the vertices of K uniformly at random and we set $f_t = f_{v_t}$. For a fixed t we have

$$\mathbb{E}_{v_t}[v_t^T \cdot x_t] = \mathbb{E}\left[\sum_i v_{t,i} \cdot x_{t,i}\right] = 0$$

This is because in each term $v_{t,i}$ is uniformly chosen from -1 and 1. Now lets look at x^* which minimizes the regret.

$$\begin{aligned}
\mathbb{E}_{v_1, v_2, ..., v_T}[\min_{x^*} \sum_t f_t(x^*)] &= \mathbb{E}\left[\min_{x^*} \sum_{t=1}^{T} \sum_{i=1}^{n} v_{t,i} x_i^*\right] \\
&= \mathbb{E}\left[\min_{x^*} \sum_{t=1}^{T} x_i^* \sum_{i=1}^{n} v_{t,i}\right] \\
&= \mathbb{E}\left[\min_{x^*} \sum_{t=1}^{T} x_i^* \sum_{i=1}^{n} v_{t,i}\right] \\
&= \mathbb{E}\left[\sum_{t=1}^{T} -|\sum_{i=1}^{n} v_{t,i}|\right] \\
&= -n\mathbb{E}\left[|\sum_{t=1}^{T} v_{t,i}|\right]
\end{aligned}$$

In claim 1 below, we show that $\mathbb{E}\left[|\sum_{t=1}^{T} v_{t,i}|\right] = \Omega(\sqrt{T})$; so, our regret will be

$$\mathbb{E}[\text{regret}] = \sum_t f_t(x_t) - \min_{x^*} \sum_t f_t(x^*) \leq 0 - (-n\sqrt{T}) = n\sqrt{T} = \Omega(GD\sqrt{T}).$$

The last step is due to $G = D = \Theta(\sqrt{n})$. □

Claim 1. If we have T random variables $v_1, ..., v_T$ each uniformly chosen between -1 and 1 we have:

$$\mathbb{E}[|\sum_{t=1}^{T} v_t|] = \Omega(\sqrt{T})$$

Proof. We can see this problem as a random walk problem. Let us have a point starting at 0 and each time it goes from x to $x-1$ with probability $\frac{1}{2}$ and goes to $x+1$ with probability $\frac{1}{2}$. Now we want to prove that

$$\mathbb{E}[\text{position after T rounds}] = \Omega(\sqrt{T}).$$

For the walk to end up at position r, we should choose $\frac{T}{2} - \frac{r}{2}$ left moves and $\frac{T}{2} + \frac{r}{2}$ right moves. So we have

$$\Pr[\text{walk ends at } r] = \frac{1}{2^T} \cdot \binom{T}{\frac{T}{2} + \frac{r}{2}} \leq \frac{1}{2^T} \cdot \binom{T}{T/2} \leq \frac{c}{\sqrt{T}} \tag{1}$$

where the last step is due to Stirling's approximation, which yields $\binom{T}{T/2} = \Theta\left(\frac{2^T}{\sqrt{T}}\right)$. Let us set $R = \frac{\sqrt{T}}{10c}$. The probability that the walk ends up at radius at most R from 0 due to (1) is:

$$\Pr[\text{walk ends at radius at most R}] \leq 2R \cdot \frac{c}{\sqrt{T}} = \frac{1}{5}.$$

Hence with probability $\frac{4}{5}$ distance of the ending point of random walk is at least $R = \Omega(\sqrt{T})$ from 0; this implies that the expected value of the ending point is $\Omega(\sqrt{T})$. □

2 Stochastic Gradient Descent

For the stochastic gradient descent problem, we have a convex body K and we want to find the point x such that it minimizes the value of function f, i.e. $\min_{x \in K} f(x)$. In the pseudocode above we have,

Algorithm 2: Stochastic Gradient Descent

input: f, K, and set of step sizes $\{\eta_t\}$

Set $x_1 \in K$

for $t = 1$ *to* T **do**

  $y_{t+1} = x_t - \tilde{\nabla}_t \eta_t$

  $x_{t+1} = \prod_K y_{t+1}$

return $\bar{x} = \frac{1}{T} \sum_{t=1}^{T} x_t$

$\mathbb{E}[\tilde{\nabla}_t] = \nabla(f(x_t))$. For example, if we have a classifier for a set of points, what gradient descent does it that calculates loss on all of the points to calculate the gradient. However, stochastic gradient descent computes the loss on a random point and the expected gradient is the same as what we have in gradient descent.

Theorem 3. Stochastic gradient descent has the following convergence property

$$\mathbb{E}[f(\bar{x})] - f(x^*) \leq \frac{3GD}{2\sqrt{T}}$$

for the same values of η_t as in Theorem 1.

Proof.

$$\begin{aligned}
\mathbb{E}[f(\bar{x})] - f(x^*) &= \mathbb{E}[f(\frac{1}{T}\sum_{t=1}^{T} x_t) - f(x^*)] \\
&\leq \mathbb{E}[\frac{1}{T}\sum_{t=1}^{T}(f(x_t) - f(x^*))] && \text{due to convexity} \\
&\leq \mathbb{E}[\frac{1}{T}\sum_{t=1}^{T}\nabla_t^T(x_t - x^*)] \\
&\leq \frac{1}{T}\mathbb{E}[\text{Regret}] && \text{due to online gradient descent by setting } f_t = \nabla_t^T(x_t - x^*) \\
&\leq \frac{3GD}{2\sqrt{T}}
\end{aligned}$$

□

3 Online Algorithms and Competitive Analysis

In online algorithms, we have a sequence of inputs. At step t we can take an action we knowledge of the input only up to the step t. Our algorithm will have a certain cost doing so. One measure is the regret that our algorithm ALG incurs which is

$$\text{regret} = \text{cost of ALG} - \text{cost of optimal static ALG}$$

By "static ALG", we mean that the algorithm we are comparing with knows the full sequence but makes a static choice for the entire sequence (e.g., a fixed expert in the online learning problem). Another useful notion to quantify how good an online algorithm does is in terms of *competitive ratio* which is defined as

$$\max_{\sigma} = \frac{ALG(\sigma)}{OPT(\sigma)}$$

where σ is any possible sequence of inputs.

3.1 Ski rental

A person is going skiing for an unknown number of days. Renting skis costs 1 per day and buying skis costs B. Every day, the person must decide whether to continue renting skis for one more day or buy a pair of skis. If the person knows in advance how many days she will go skiing, she can decide her minimum cost. If she will be skiing for more than B days it will be cheaper to buy skis but if she will be skiing for fewer than B days it will be cheaper to rent. What should she do when she does not know in advance how many days T she will ski?

In each step, the algorithm will decide whether to rent or buy skis. And an algorithm is essentially deciding when to buy because once we buy the ski, we do not need to rent implying that the cost incurred in any step

after the buying of the ski is zero. So let us define ALG_i to be the algorithm that rents for the first i steps and buys on step $i+1$. Also let us classify the inputs and call I_j to be the input where the sequence ends at the j-th day. The optimal cost for I_j would be j if we have $j < B$ and it is B if we have $j \geq B$.

A natural algorithm is to balance the renting and buying; that is, buy when the amount that you have paid for renting is equal to buying once. Formally we prove the following theorem.

Theorem 4. The competitive ratio of algorithm ALG_{B-1} is $2 - \frac{1}{B}$ and this is the best possible ratio for any deterministic algorithm.

Proof. There are two cases to consider. In the first case, where $j < B+1$, the competitive ratio of ALG_{B-1} is 1. In the second case where we have $j \geq B+1$, $\mathsf{ALG}_{B-1}(I_j) = 2B-1$ and $\mathsf{OPT}(I_J) = B$ so the competitive ratio is $2 - \frac{1}{B}$. Since the competitive ratio is the maximum over all inputs I_j the overall competitive ratio will be $2 - \frac{1}{B}$.

Now, to show that this is the best possible competitive ratio for any deterministic algorithm. Consider any algorithm ALG_i. The worst instance for this algorithm is I_j with $j = i+1$. Note that $\mathsf{ALG}_i(I_j) = (i+B)$. Consider the following two cases.

- if $i \geq B-1$, then $\mathsf{OPT}(I_j) = B$. Therefore,

$$\frac{\mathsf{ALG}(I_j)}{\mathsf{OPT}(I_j)} = \frac{i+B}{B} = \frac{i+1}{B} + 1 - \frac{1}{B} \geq 2 - \frac{1}{B}$$

- if $0 \geq i \geq B-2$ then $\mathsf{OPT}(I_j) = i+1$, and

$$\frac{\mathsf{ALG}(I_j)}{\mathsf{OPT}(I_j)} = \frac{i+B}{i+1} = 1 + \frac{B-1}{i+1} \geq 2$$

□

We will see in the next lecture that we can get a better competitive ratio using randomization.

References

[BN+09] Niv Buchbinder, Joseph Seffi Naor, et al. The design of competitive online algorithms via a primal–dual approach. *Foundations and Trends® in Theoretical Computer Science*, 3(2–3):93–263, 2009.

[Nag21] Viswanath Nagarajan. Introduction to online optimization. Lecture notes, available from `https://viswa.engin.umich.edu/wp-content/uploads/sites/169/2021/02/online-basics.pdf`, 2021.