1 Stochastic Gradient Descent

Stochastic gradient descent is a variant of the Gradient Descent algorithm in Section 3 of the last lecture, which works with convex functions presented using an even weaker notion: an expected gradient oracle. Given a point \( z \), this oracle returns a linear function \( g_x + c \) that is drawn from a probability distribution \( D_z \) such that the expectation \( E_{g,c\in D_z}[g_x+c] \) is exactly the gradient of \( f \) at \( z \). Such a distribution \( D_z \) is often called an unbiased estimator of the gradient.

Example 1 (Spam classification using SGD). Returning to the spam classification problem from last lecture, we see that the Loss function is a sum of many similar terms. If we randomly pick a single term and compute just its gradient (which is very quick to do!) then by linearity of expectations, the expectation of this gradient is just the true gradient. Thus the expected gradient oracle may be a much faster computation than the gradient oracle (a million times faster if the number of email examples is a million!). In fact this setting is not atypical; often the convex function of interest is a sum of many similar terms.

Stochastic gradient descent can be analysed using Online Gradient Descent (OGD) from last lecture. Let \( g_i \) be the gradient at step \( i \). Then we use the function \( g_i \cdot x \), which is a linear function and hence convex, as \( f_i \) in the \( i \)-th step of OGD from last lecture. Let \( z = \frac{1}{T} \sum_{i=1}^{T} x^{(i)} \). Let \( x^* \) be the point in \( K \) where \( f \) attains its minimum value.

**Theorem 1.** \( \mathbb{E}[f(z)] \leq f(x^*) + \frac{2DG}{\sqrt{T}} \), where \( D \) is the diameter as before and \( G \) is an upperbound of the norm of any gradient vector ever output by the oracle.

Proof.

\[
\mathbb{E}[f(z) - f(x^*)] \leq \frac{1}{T} \mathbb{E} \left[ \sum_{i} (f(x^{(i)}) - f(x^*)) \right] \quad \text{by convexity of } f
\]

\[
\leq \frac{1}{T} \sum_{i} \mathbb{E}[\nabla f(x^{(i)}) \cdot (x^{(i)} - x^*)] \quad \text{using defn of convexity}
\]

\[
= \frac{1}{T} \sum_{i} \mathbb{E}[g_i \cdot (x^{(i)} - x^*)] \quad \text{since expected gradient is the true gradient}
\]

\[
= \frac{1}{T} \sum_{i} \mathbb{E}[f_i(x^{(i)}) - f_i(x^*)] \quad \text{defn. of } f_i
\]

\[
= \frac{1}{T} \mathbb{E} \left[ \sum_{i} (f_i(x^{(i)}) - f_i(x^*)) \right]
\]

1
and the theorem now follows since the expression in the $\mathbb{E}[]$ is just the regret, which is always upperbounded by the quantity given in Zinkevich’s theorem, so the same upperbound holds also for the expectation. □

Note that in Theorem 1 we assumed that $G$ is a bound on the norm of any gradient vector outputted, and not on the gradient of the original function $f$. For SGD applications like Spam Classification where we pick a random term and compute its gradient, we need to scale this computed gradient by the number of terms to get the same expectation as the true gradient, which increases the bound on the norm of the gradient vector. This is where SGD pays over standard Gradient Descent: each iteration in SGD is cheaper but you may need more iterations.

## 2 Bandit Gradient Descent

Let’s recall the setup of Online Gradient Descent (OGD). There is a convex set $\mathcal{K}$ given via a projection oracle. For $i = 1, 2, \ldots, T$ we are presented at step $i$ a convex function $f_i$. At step $i$ we have to put forth our guess solution $x^{(i)} \in \mathcal{K}$ but the catch is that we do not know the functions that will be presented in future. So our online decisions have to be made such that if $x^*$ is the point $w$ that minimizes $\sum_i f_i(w)$ (i.e. the point that we would have chosen in hindsight after all the functions were revealed) then the following quantity (called regret) should stay small: $\sum_i \left( f_i(x^{(i)}) - f_i(x^*) \right)$.

In the last lecture we saw Online Gradient Descent which gets regret $O(DG\sqrt{T})$, where in each step $i$ we take a towards the negative gradient of $f_i$. This is possible since after we play $x^{(i)}$, the function $f_i$ is revealed so we can compute its gradient. What if instead of getting the entire function $f_i$, the algorithm gets a Bandit feedback where it is only revealed the incurred scalar payoff/cost $f_i(x^{(i)})$, so we cannot compute the gradient?

The setup for Bandit Gradient Descent (BGD) is the same as the setup for OGD, except that the algorithm only receives $f_i(x^{(i)})$ as feedback after choosing $x^{(i)}$, instead of receiving the entire function $f_i$. The goal is to still minimize the regret:

$$\sum_i \left( f_i(x^{(i)}) - f_i(x^*) \right).$$

In the rest of the lecture, we will see how to use SGD to obtain $o(T)$ regret for BGD.

To keep things simple, we will assume that $\mathcal{K}$ contains a unit-ball centered at origin and that $|f_i(x)| \leq 1$ for all $i$ and all $x \in \mathcal{K}$. These assumptions can be relaxed; see [1, 2] details. The main result is the following:

**Theorem 2** (Flaxman-Kalai-McMahan [1]). \textit{There is an online algorithm for Bandit Convex Optimization with regret $O(T^{3/4} \cdot DG\sqrt{n})$.}
2.1 Gradient Descent without a Gradient

Recall that for a function $g : \mathbb{R} \to \mathbb{R}$, its gradient at any point $x \in \mathbb{R}$ can be computed by taking

$$\lim_{\epsilon \to 0} \frac{g(x + \epsilon) - g(x - \epsilon)}{2\epsilon}.$$

The main intuition on which Theorem 2 relies is that one way of obtaining an unbiased estimate of $g'(x)$ given only access to the value of $g(\cdot)$ at a single point $w$ is to choose randomly: $w = x + \delta \epsilon$, where $\delta = +1$ with probability $1/2$ and $\delta = -1$ otherwise.

We now have

$$\frac{1}{\epsilon} \mathbb{E}\delta[\delta \cdot g(w)] = \frac{1}{2\epsilon} \left( g(x + \epsilon) - g(x - \epsilon) \right),$$

which equals $g'(x)$ as $\epsilon \to 0$. Thus, we have managed to get an unbiased estimator of the gradient using a single-sample from the function.

Next, we will generalize the above idea to higher dimensions.

**Lemma 3.** Consider any differentiable function $f : \mathbb{R}^n \to \mathbb{R}$. Let $v$ denote a uniformly-random $n$-dimensional vector on the surface of a sphere of radius 1. Then,

$$\lim_{\epsilon \to 0} d/\epsilon \mathbb{E}_v[v \cdot f(x + \epsilon v)] = \nabla f(x).$$

We will not prove this lemma since a formal proof uses Stokes’ theorem, but the intuition is the same as in the 1-d case. Roughly, the factor of $d$ comes because we are in any of the $d$ axis directions only $1/d$ fraction of the time.

Since for our discrete algorithm we cannot choose $\epsilon \to 0$, we work with a slight perturbation of the original function $f$. For any fixed $\epsilon > 0$, define a “smoothed” approximation of $f$ as $\hat{f}(x) := \mathbb{E}_v[f(x + \epsilon v)]$.

The idea is that $\hat{f}(x)$ pretty much behaves the same as the function $f(x)$, but has the advantage that it satisfies

$$\frac{d}{\epsilon} \mathbb{E}_v[v \cdot f(x + \epsilon v)] = \nabla \hat{f}(x). \quad (1)$$

2.2 Flaxman-Kalai-McMahan Algorithm

To prove Theorem 2, in each step of OGD we will play a small perturbation $w^{(i)}$ of $x^{(i)}$, and then use the feedback with Lemma 3 to perform the OGD step in expectation. The formal algorithm is described in Figure 1, where we project on to a slight rescaling $K_{1-\epsilon}$ of the convex body $K$ to ensure that we play a feasible point $w^{(i)}$ even after we perturb $x^{(i)}$—a point $x \in K_{1-\epsilon}$ iff $x(1 + \epsilon) \in K$.

We first observe that using (1), we have

$$\mathbb{E}_v[y^{(i+1)}] = x^{(i)} - \eta \frac{d}{\epsilon} \mathbb{E}_v[v_i \cdot f(w^{(i)})] = x^{(i)} - \eta \nabla \hat{f}(x^{(i)})$$
Let $\eta = \frac{D G}{G \sqrt{T} n}$ and $\epsilon = \frac{1}{T^{1/4}} \sqrt{n}$.

Repeat for $i = 0$ to $T$

1. Choose a unit vector $v_i$ in $\mathbb{R}^n$.
2. Play $w(i) = x(i) + \epsilon v_i$ and observe $f_i(w(i))$.
3. Let $y(i+1) \leftarrow x(i) - \eta n \epsilon v_i \cdot f(w(i))$.
4. $x(i+1) \leftarrow$ Projection of $y(i+1)$ on $K_{1-\epsilon}$.

Figure 1: Flaxman-Kalai-McMahan Algorithm

Thus, our algorithm in Figure 1 can be viewed as performing the Stochastic Gradient Decent algorithm on the function $\hat{f}$ and the convex body $K_{1-\epsilon}$. Notice that if the gradient of $f$ has norm at most $G$, then the gradient of any output vector has norm at most $G' = Gn/\epsilon$. So, by the online variant of Theorem 1, we get for $\eta = \frac{D G}{G' \sqrt{T}}$ that

$$\sum_i E[\hat{f}_i(x(i))] - \min_{z \in K_{1-\epsilon}} \sum_i \hat{f}_i(z) \leq O(DG' \sqrt{T}) = O(DGn \sqrt{T}/\epsilon).$$

Since we are only playing points inside $K_{1-\epsilon}$, instead of $K$, this incurs an error of at most $\epsilon DG$ per time step as the gradient is bounded by $G$ and the distance moved is at most $\epsilon D$. Moreover, the difference between $\hat{f}(x)$ and $f(x)$ is bounded by at most $\epsilon G$ since $\hat{f}$ is formed by taking a convex combination of points at distance at most $\epsilon$ from $x$. Hence, we have

$$\sum_i E[f_i(x(i))] - \sum_i f_i(x^*) \leq O(\epsilon DT + \epsilon GT) + O(DGn \sqrt{T}/\epsilon),$$

which proves Theorem 2.

Remark: It’s also known that one can achieve $O(\text{poly}(nGD) \sqrt{T})$ regret for Bandit Convex Optimization. The first efficient algorithm obtaining such $\sqrt{T}$ bound was given in [2].

References


