Optimization: Algorithms, Complexity & Approximations

Anastasios Kyrillidis *

*Instructor, Computer Science at Rice University

Contributors: Nick Sapoval, Carlos Quintero Pena, Delaram Pirhayatifard, McKell Stauffer, Mohammad Taha Toghani

Chapter 7

In the last chapter, we considered natural ways of accelerating the performance of gradient descent, by cleverly using previous estimates via the momentum term. However, while acceleration in that setting leads to faster convergence in analytical complexity terms, it does not reduce the per iteration complexity.

In this chapter, we will discuss how we can accelerate the performance of gradient methods by computing less per iteration: based on the empirical risk minimization formulation, we will discuss stochasticity and how the stochastic version of gradient descent can be beneficial in practice. Definitely, such selection creates trade-offs (number of iterations to converge vs. computational complexity per iteration). As an alternative way of accelerating gradient descent, we also mention the notion of coordinate descent, where we update individual entries of the variable vector, instead of the full vector, per iteration (more details in class).

Stochastic gradient descent and its variants | Coordinate descent and its variants

To discuss about stochasticity in optimization, we will first need a new problem definition, that is amenable to stochasticity. While stochastic optimization and stochastic approximations are large research areas by themselves, a particular optimization criterion that appears often in the literature is that of the empirical risk minimization. Let us define that through an example.

Consider that we have a dataset $\mathcal{D}$ that is comprised by n data points $\{w_i, y_i\}_{i=1}^n$: here, we can simplify the discussion by assuming that w_i is a vectorized version of an image, $w_i \in \mathbb{R}^m$, and $y_i \in \mathbb{R}$ is the label for that image. We are interested in designing a predictor function $h : \mathbb{R}^m \to \mathbb{R}$ that takes as an input an image w_i and predicts its correct label y_i. This predictor is to be learned, and the learning parameters are described via a real vector $x \in \mathbb{R}^p$, such that $h(x, w_i) = y_i$. To make this even more explicit, if we were talking specifically for the case of linear regression, and under the setting $p = m$, the predictor should be:

$$h(x, w_i) = x^\top w_i = y_i, \forall w_i$$

In the above description, what is vague is the statement "$\forall w_i$". What if we have $n = 1000$ samples and then we have 1000 more? Should we create a predictor that operates flawlessly on the given dataset, but gives no guarantees how it operates on unseen data? This leads to the notion of *expected risk*: can we find parameters x that minimize the prediction loss on average, over all possible input-output data points? I.e., if we measure the loss as $(h(w, x) - y)^2$, can we optimize:

$$\min_x \mathbb{E}_{\{x,y\}\sim\mathcal{D}} \left[(h(w, x) - y)^2\right].$$

In reality, we do not have access to the true distribution of how data is generated. Thus, we cannot practically optimize this objective, but we can approximate it through the *empirical risk*: assuming that each data point is generated i.i.d. in a uniform way, we can approximate the expected risk with the empirical risk, and solve:

$$\min_x \tfrac{1}{n} \sum_{i=1}^n \left[(h(w_i, x) - y_i)^2\right].$$

Let us define $f_i(x) := (h(w_i, x) - y_i)^2$. Then, the above objective is written as:

$$\min_x \left\{ f(x) := \tfrac{1}{n} \sum_{i=1}^n f_i(x) \right\}.$$

How would we solve this problem? By applying gradient descent methods! In particular, we can compute the gradient at the iteration t as:

$$\nabla f(x_t) = \nabla \left[\sum_{i=1}^n f_i(x_t)\right] = \sum_{i=1}^n \nabla f_i(x_t);$$

i.e., the full gradient is actually the summation of the gradients of each individual $f_i(\cdot)$. Thus, if the computation of $\nabla f_i(\cdot)$ takes $\mathtt{T}_{\nabla f_i(\cdot)}$ time, the gradient descent requires $n \cdot \mathtt{T}_{\nabla f_i(\cdot)}$ iteration complexity to compute the full gradient. Assuming that the problem is convex, L-smooth and μ-strongly convex, then this implies that the total complexity of gradient descent:

$$x_{t+1} = x_t - \eta \nabla f(x_t),$$

to get to an ε-solution is:

$$O\left(n \cdot \mathtt{T}_{\nabla f_i(\cdot)} \cdot \log \tfrac{1}{\varepsilon}\right).$$

(We neglect the presence of κ in the above expression for clarity). *But do we really need to compute the full gradient per iteration?*

Prototypical stochastic gradient descent (SGD). The natural question to ask is "can we remove the n in the above complexity?". This stems from the fact that we have massive amount of data: imagine that you are working with the ImageNet dataset, that has more than 14 million images. Having $n = 14 \cdot 10^6$ in the complexity above creates a huge burden in terms of computational resources.

What if we just computed the gradient on only one sample per iteration? I.e., we perform:

$$x_{t+1} = x_t - \eta \nabla f_{i_t}(x_t),$$

where $i_t \in [n]$ and is selected randomly. We can easily observe that, in this case, each iteration is very cheap, involving only the computation of the gradient $\nabla f_{i_t}(x_t)$, corresponding to one sample. Thus, someone would expect to obtain something like $O\left(\mathtt{T}_{\nabla f_i(\cdot)} \cdot \log \tfrac{1}{\varepsilon}\right)$, *but unfortunately this is not the case.* Discussing such trade-offs is the goal of this chapter: how we configure SGD algorithms is an active research area.

Some motivation for using SGD. There are both practical and theoretical motivations in using SGD in practice.

For practical motivation, we can easily construct cases that intuitively does not make sense using full gradient over SGD. E.g., consider the case where our dataset $\mathcal{D}$ has n copies of a single data sample w_1, y_1. In that case, $\nabla f(x) = \frac{1}{n} \cdot \sum_{i=1}^{n} \nabla f_i(x) = \frac{1}{n} \cdot n \cdot \nabla f_1(x) = \nabla f_1(x)$. Thus, $x_{t+1} = x_t - \eta \nabla f(x_t) = x_t - \eta \nabla f_{i_t}(x_t)$, but we have wasted $(n-1) \cdot \mathtt{T}_{\nabla f_i(\cdot)}$ complexity to compute the full gradient. This reasoning stems from the fact that, in many applications, it is not the case that all data points provide *new information*. There might be cases where clusters of data can be well-represented by a single data point, and thus computing gradients for each data sample is a waste of computational resources. This issue is handled naturally with SGD.

The above can be empirically observed with real data. In the figure that follows from the writeup by Bottou, Curtis and Nocedal, it is obvious that SGD motions can be even faster than quasi-Newton methods, that exploit the curvature of the objective, beyond just the gradient information. (*Caveat: to achieve this plot though, heavy hyper-parameter tuning was required for SGD.*)

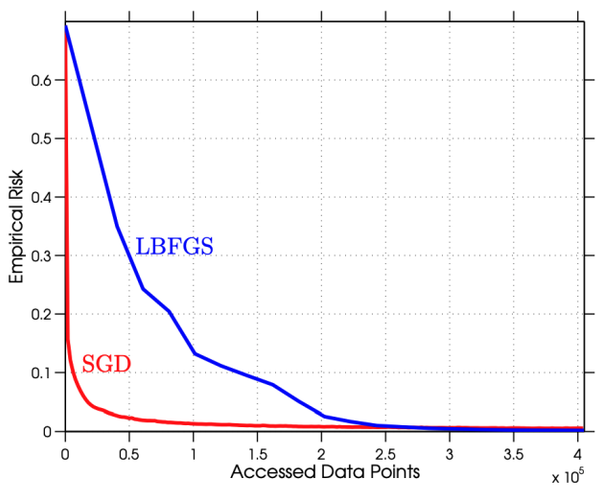

Fig. 1. Binary classification problem using logistic regression and the RCV1 dataset. Borrowed from Bottou, Curtis and Nocedal manuscript on "Optimization methods for large-scale machine learning" [1].

The theoretical motivation that stems from using SGD lies in that, while we often do not achieve in theory the same convergence rate than full gradient descent, SGD removes the n factor in the total complexity, creating an interesting trade-off for the practitioner. Moreover, it turns out that in the stochastic optimization setting, SGD yields the same convergence rate, even if we work in the *expected risk case.* Having such strong guarantees in the full gradient descent is not possible, as the latter is not even viable without the ability to compute the full gradient on, let's say, countably infinite data points that can be generated from the data distribution.

Some examples for SGD for common functions. We will consider the classical examples of linear and logistic regression.

Linear regression: In this well-studied setting, we can easily decompose the full gradient objective $\frac{1}{2}\|y - Ax\|_2^2$ into a summation of smaller objectives:

$$f(x) := \tfrac{1}{2}\|y - Ax\|_2^2 = \sum_{i=1}^{n} \tfrac{1}{2} \left(y_i - \alpha_i^\top x\right)^2 := \sum_{i=1}^{n} f_i(x).$$

Then, SGD looks like the following:

$$x_{t+1} = x_t - \eta \nabla f_{i_t}(x_t) = x_t + \eta \alpha_{i_t}(y_{i_t} - \alpha_{i_t}^\top x_t)$$

Logistic regression: In logistic regression, the objective function is similarly the summation of objectives obtained from samples:

$$f(x) := \tfrac{1}{n} \sum_{i=1}^{n} \log(1 + \exp(-y_i \alpha_i^\top x)) := \frac{1}{n} \sum_{i=1}^{n} f_i(x)$$

As we shown in Chapter 2:

$$\nabla f_{i_t}(x_t) = \frac{-y_i}{1 + \exp(y_{i_t} \alpha_{i_t}^\top x_t)} \alpha_{i_t}$$

Then, SGD looks like the following:

$$x_{t+1} = x_t - \eta \nabla f_{i_t}(x_t) = x_t + \eta \frac{y_i}{1 + \exp(y_{i_t} \alpha_{i_t}^\top x_t)} \alpha_{i_t}$$

Basics of theory on SGD. Let us first define the notion of *unbiasedness*:

Definition 1. (Unbiasedness) *In statistics, we call the variable $\hat{\theta}$ an unbiased estimation of θ if:*

$$\mathbb{E}[\hat{\theta}] = \theta.$$

We will use the above definition to show that stochastic gradients are unbiased estimators of the full gradient. In particular, consider the case where $\nabla f(x) = \frac{1}{n} \sum_{i=1}^{n} f_i(x)$, similar to the empirical risk minimization case. Then:

Claim 1. *Let i be selected uniformly at random from the set $[n]$. Then, the atomic gradient $\nabla f_i(x)$ is an unbiased estimator of the true gradient, $\nabla f(x)$.*

Proof:

$$\begin{aligned} \mathbb{E}_i\left[\nabla f_i(x)\right] &= \sum_{j=1}^{n} \mathbb{P}\left[j = i\right] \nabla f_j(x) \\ &= \frac{1}{n} \sum_{j=1}^{n} \nabla f_j(x) = \nabla f(x). \end{aligned}$$

□

In this section, we would like to characterize theoretically the performance of SGD:

$$x_{t+1} = x_t - \eta_{i_t} \nabla f_{i_t}(x_t).$$

In order to limit the harmful effect of stochasticity, it is required to assume that the variance of $\nabla f_{i_t}(x_t)$ (its norm) is being bounded, or in other words one of the following inequalities is being assumed:

- There exist some $M, M_f \geq 0$ such that:

$$\mathbb{E}_{i_t}\left[\|\nabla f_{i_t}(x_t)\|_2^2\right] \leq M + M_f \|\nabla f(x_t)\|_2^2;$$

or

- There exists a $C \geq 0$ such that:

$$\mathbb{E}_{i_t}\left[\|\nabla f_{i_t}(x_t)\|_2^2\right] \leq C.$$

The former means that norm of each individual gradient is not much greater than norm of the full gradient, while the latter implies that norm of each individual gradient is bounded with a constant.

SGD for smooth and strongly convex f with constant step size. We will focus on the following claim:

Claim 2. *Assume that f is a i) differentiable, ii) L-smooth, and iii) a μ-strongly convex function. There exist a constant $\eta \geq 0$ such that applying SGD from point x_0 with step size η, then:*

$$\mathbb{E}\left[f(x_{t+1}) - f(x^\star)\right] \leq (1-\mu\eta)^t \left(f(x_0) - f(x^\star)\right) + \mathcal{O}(\eta)$$

where $x^\star$ is the global minimizer of f.

Proof: By using the assumption of Lipschitz gradients, we have:

$$\begin{aligned} f(x_{t+1}) &\leq f(x_t) + \langle \nabla f(x_t), x_{t+1} - x_t \rangle + \tfrac{L}{2}\|x_{t+1} - x_t\|_2^2 \\ &= f(x_t) + \langle \nabla f(x_t), x_t - \eta_t \nabla f_{i_t}(x_t) - x_t \rangle \\ &\qquad + \tfrac{L}{2}\|x_t - \eta_t \nabla f_{i_t}(x_t) - x_t\|_2^2 \\ &= f(x_t) - \eta \langle \nabla f(x_t), \nabla f_{i_t}(x_t) \rangle + \tfrac{L}{2}\eta^2 \|\nabla f_{i_t}(x_t)\|_2^2 \end{aligned}$$

Let's assume that indices $i_0, \ldots, i_{t-1}$ have been selected uniformly at random. So, taking the expectation with respect to i_t given $i_0, \ldots, i_{t-1}$ (for brevity we write $\mathbb{E}_{i_t}[.]$ instead of $\mathbb{E}_{i_t|i_{t-1},\ldots,i_0}[.]$):

$$\begin{aligned} \mathbb{E}_{i_t}\left[f(x_{t+1})\right] &\leq f(x_t) - \eta \langle \nabla f(x_t), \mathbb{E}_{i_t}\left[\nabla f_{i_t}(x_t)\right] \rangle \\ &\qquad + \tfrac{L}{2}\eta^2 \mathbb{E}_{i_t}\left[\|\nabla f_{i_t}(x_t)\|_2^2\right] \\ &\leq f(x_t) - \eta \|\nabla f(x_t)\|_2^2 \\ &\qquad + \tfrac{L}{2}\eta^2 (M + M_f \|\nabla f(x_t)\|_2^2) \end{aligned}$$

which is the result of unbiasedness and boundedness of each individual gradient norm. Hence:

$$\begin{aligned} \mathbb{E}_{i_t}\left[f(x_{t+1}) - f(x^\star)\right] &\leq \left(f(x_t) - f(x^\star)\right) + \tfrac{L\eta^2 M}{2} \\ &\qquad - (\eta - \tfrac{L\eta^2 M_f}{2}) \|\nabla f(x_t)\|_2^2 \quad (\star) \end{aligned}$$

By strong convexity, we have:

$$\begin{aligned} f(x_t) \leq f(x^\star) + \langle \nabla f(x^\star), x_t - x^\star \rangle + \tfrac{1}{2\mu}\|\nabla f(x_t) - \nabla f(x^\star)\|_2^2 \\ \Rightarrow f(x_t) - f(x^\star) \leq \tfrac{1}{2\mu}\|\nabla f(x_t)\|_2^2 \end{aligned}$$

Furthermore, for $\eta \leq \frac{2}{LM_f}$, it is guaranteed that $\eta - \frac{L\eta^2 M_f}{2} \geq 0$. Combining the two above facts in $(\star)$, we have:

$$\begin{aligned} \mathbb{E}_{i_t}\left[f(x_{t+1}) - f(x^\star)\right] &\leq \left(f(x_t) - f(x^\star)\right) + \tfrac{L\eta^2 M}{2} \\ &\qquad - 2\mu \left(\eta - \tfrac{L\eta^2 M_f}{2}\right) \left(f(x_t) - f(x^\star)\right) \\ &= \left(1 - 2\mu\left(\eta - \tfrac{L\eta^2 M_f}{2}\right)\right) \left(f(x_t) - f(x^\star)\right) \\ &\qquad + \tfrac{L\eta^2 M}{2} \\ \text{(Assume } \eta = \tfrac{1}{LM_f}\text{ :)} &= (1-\mu\eta)(f(x_t) - f(x^\star)) + \tfrac{L\eta^2 M}{2} \end{aligned}$$

Repeating the chain for $i_{t-1}, \ldots, i_0$, we have:

$$\begin{aligned} \mathbb{E}\left[f(x_{t+1}) - f(x^\star)\right] &\leq (1-\mu\eta)^t (f(x_0) - f(x^\star)) \\ &\qquad + \sum_{j=0}^{t} (1-\mu\eta)^j \tfrac{L\eta^2 M}{2} \\ &= (1-\mu\eta)^t (f(x_0) - f(x^\star)) \\ &\qquad + \tfrac{L\eta^2 M}{2} \cdot \tfrac{1-(1-\mu\eta)^{t+1}}{1-1+\mu\eta} \\ \text{(Assume } \eta \leq \frac{1}{\mu}\text{ :)} &= (1-\mu\eta)^t (f(x_0) - f(x^\star)) \\ &\qquad + \tfrac{L\eta M}{2\mu} (= \mathcal{O}(\eta)) \end{aligned}$$

Note that the above result holds for $\eta \leq \min\left\{\frac{1}{LM_f}, \frac{1}{\mu}\right\}$. □

According to the above proof, we observe:

1. Fast linear convergence is connected with smaller $(1-\mu\eta)$ which is subject to selecting (valid) larger value of η.
2. After large enough iterations, the algorithm converges around a neighborhood of radius $\mathcal{O}(\eta)$.
3. When we do full gradient descent, $M = 0, M_f = 1$.
4. Smaller step sizes (η) yield better convergence, although slower, so there is a trade-off between accuracy and speed of the algorithm.

SGD for smooth and strongly convex f with decreasing step sizes. Here, we assume the simpler case:

$$\mathbb{E}_{i_t}\left[\|\nabla f_{i_t}(x_t)\|_2^2\right] \leq G^2$$

Claim 3. *Assume that f is a i) differentiable, ii) L-smooth, and iii) μ-strongly convex function. If we apply SGD starting from point x_1 with step size $\eta_t = \frac{1}{\mu t}$, then:*

$$\mathbb{E}\left[\|x_{t+1} - x^\star\|_2^2\right] \leq \frac{\max\{\|x_1 - x^\star\|_2^2, \frac{G^2}{\mu^2}\}}{t+1} =: \frac{\Delta}{t+1}$$

where $x^\star$ is the global minimizer of f.

Proof: We know that:

$$\begin{aligned} \mathbb{E}_{i_t}\left[\|x_{t+1} - x^\star\|_2^2\right] &\leq \|x_t - x^\star\|_2^2 + \eta_t^2 \mathbb{E}_{i_t}\left[\|\nabla f_{i_t}(x_t)\|_2^2\right] \\ &\qquad - 2\eta_t \left\langle \mathbb{E}_{i_t}\left[\nabla f_{i_t}(x_t)\right], x_t - x^\star \right\rangle \end{aligned}$$

By strong convexity:

$$\langle \nabla f(x) - \nabla f(x^\star), x - x^\star \rangle = \langle \nabla f(x), x - x^\star \rangle \geq \mu \|x - x^\star\|_2^2$$

Then:

$$\begin{aligned} \mathbb{E}_{i_t}\left[\|x_{t+1} - x^\star\|_2^2\right] &\leq \|x_t - x^\star\|_2^2 + \eta_t^2 G^2 - 2\eta_t\mu \|x_t - x^\star\|_2^2 \\ &= (1 - 2\eta_t\mu)\|x_t - x^\star\|_2^2 + \eta_t^2 G^2 \quad (\star\star) \end{aligned}$$

Finally, in order to prove the claim 3, let's use induction. First, it is trivial that:

$$\mathbb{E}\left[\|x_1 - x^\star\|_2^2\right] = \|x_1 - x^\star\|_2^2 \leq \frac{\max\{\|x_1 - x^\star\|_2^2, \frac{G^2}{\mu^2}\}}{1}$$

Assume that for t:

$$\mathbb{E}\left[\|x_t - x^\star\|_2^2\right] \leq \frac{\max\{\|x_1 - x^\star\|_2^2, \frac{G^2}{\mu^2}\}}{t} = \frac{\Delta}{t}$$

Using the chain of expectations, according to assumption of the induction and $(\star\star)$, we have:

$$\begin{aligned} \mathbb{E}\left[\|x_{t+1} - x^\star\|_2^2\right] &\leq \left(1 - \frac{2}{t}\right)\mathbb{E}\left[\|x_t - x^\star\|_2^2\right] + \frac{G^2}{\mu^2 t^2} \\ &\leq \left(1 - \frac{2}{t}\right)\frac{\Delta}{t} + \frac{G^2}{\mu^2 t^2} \\ &\leq \left(\frac{1}{t} - \frac{2}{t^2}\right)\Delta + \frac{\Delta}{t^2} \\ &= \left(\frac{1}{t} - \frac{1}{t^2}\right)\Delta \\ &\leq \frac{\Delta}{t+1} \end{aligned}$$

□

Comparing GD and SGD. We studied both gradient descent and its stochastic version carefully. As we showed in claim 2, for the ideal case of smooth and strong convex function, having the linear convergence rate, there is a trade-off between the accuracy and the speed. Then in claim 3, using a decreasing learning rate, we guaranteed accuracy of SGD but lost the linear convergence rate. While, in previous chapters, we showed that GD, in addition to keeping the accuracy, exploits a linear convergence rate. So, number of samples plays a critical role in making the decision whether to choose GD or SGD. In other words, although, GD converges with the less number of iterations to global minimizer of the function, but each iteration can consume a considerable portion of the time. SGD, needs more iterations to converge to the global minimum with guaranty but it needs less complexity per iteration. Thus, here, the theory, justifies what we intuitively expect from stochastic and full gradient descent.

Note that instead of these two methods, we can use a mini-batch of samples to do a stochastic gradient descent in each iteration, but using the expectation analysis regime, the rate of convergence does not change. So, is there any gap between theoretical analysis and practice?

Coordinate descent. Similar to what we did for samples, the idea of coordinate descent is to update just one feature (or a subset of features) in each iteration. This method seems interesting specially for the cases with $p \gg 1$. *What if we just computed the gradient on only one feature per iteration?* I.e., we perform:

$$(x_{t+1})_{i_t} = (x_t)_{i_t} - \eta_t \nabla_{i_t} f(x_t),$$

where $i_t \in [p]$ and is selected randomly. Regarding the fact that in each iteration, we just compute the gradient with respect to one coordinate, $\nabla_{i_t} f(x_t)$, each iteration is cheaper than its counterpart in GD. (Something like $O\left(\mathtt{T}_{\nabla_i f(\cdot)} \cdot \log \frac{1}{\varepsilon}\right)$, is expected.)

Appendix

1. **L. Bottou, F. Curtis, and J. Nocedal. Optimization methods for large-scale machine learning. Siam Review, 60(2):223–311, 2018.**