Review of Probability and Stochastic gradient descent (SGD)

August 11, 2019

1 Motivation

Given data $(x_i, y_i)_{i=1}^m$ ($x_i \in \mathbb{R}^n$ and $y_i \in \mathbb{R}$), we wanted to minimize

(1) (Linear-least squares) $\min_\theta \|X\theta - y\|_2^2 = \sum_{i=1}^m (x_i^T\theta - y_i)^2$

(2) (Logistic regression) $\min_\theta \sum_{i=1}^m \log(1+\exp(\theta^T x_i)) - y_i\theta^T x_i$

Observe both of these take the form

$$\text{(Empirical loss/large finite sums)} \qquad \min_\theta f(\theta) = \frac{1}{m}\sum_{i=1}^m f_i(\theta) = \frac{1}{m}\sum_{i=1}^m \ell(h(\theta, x_i), y_i). \qquad (\star)$$

- h is the prediction function; parameterized by θ and ℓ is the loss function
  - Linear least squares: $h(\theta, x) = X\theta$, $\ell(;y) = \|\cdot - y\|^2$
  - Logistic regression: $h(\theta, x) = \theta^T x$ and $\ell(\cdot; y) = \log(1+\exp(\cdot)) - y\cdot$
- Many ML problems take this form; Much more complicated prediction functions and loss function.
- Think m is large- the number of samples is big
- Part of a more general form:

$$\text{(Expected loss)} \qquad \min_\theta f(\theta) = \mathbf{E}_\xi \ell \circ h(\theta;\xi) \qquad (\star\star)$$

(i) ξ is a random variable

(ii) ℓ is the loss function and h is predictor θ on a sample ξ

(iii) More interested in $(\star\star)$ but can't solve it so we approx. it by empirical loss $(\star)$

$$\boxed{\textbf{Problem:}\text{ Because } m \text{ is large, we can't compute } \nabla f(\theta) \text{ or even evaluate } f(\theta).}$$

Remark 1. For $(\star\star)$, convexity/differentiability (resp.) means $\ell \cdot h(\cdot;\xi)$ is convex/differentiable for all ξ

$$\boxed{\textbf{Question:}\text{ How do we minimize } (\star) \text{ and } (\star\star)?}$$

$\Rightarrow$ Use probability! Instead of computing $\nabla f(\theta)$ we randomly choose sample x_i (or ξ_i) and use $\nabla f_i(\theta)$ or $\nabla \ell(\theta;\xi_i)$ as a surrogate for the full gradient.

2 Stochastic Gradient Descent (SGD)

This is a very old method established by Robbins and Monro [1951]: gradients do not need to be computed exactly in order to guarantee progress toward optimum. They observed that as long as the gradients are correct *on average*, the error introduced by the gradient approximations will eventually vanish.

Recall we are interested in solving

$$\min_\theta f(\theta) := \mathbf{E}_\xi[\ell(\theta;\xi)] \quad \text{and} \quad \min_\theta \; f(\theta) := \frac{1}{m}\sum_{i=1}^m f_i(\theta) = \frac{1}{m}\sum_{i=1}^m \ell(\theta,\xi_i).$$

For simplicity, lets assume $f(\theta)$ is has L- smooth.

$$\boxed{\text{What optimization algorithm might we run?-gradient descent}}$$

$$\theta_{k+1} = \theta_k - \alpha_k g(\theta_k), \quad \text{where } g(\theta_k) \text{ is an estimator of the gradient.}$$

Definition 2.1 (Stochastic oracle)**.** A *stochastic oracle* for a convex function $f : \mathbb{R}^m \to \mathbb{R}^n$ inputs a random variable Θ and outputs a random variable $g(\Theta,\xi)$ such that $\mathbf{E}[g(\Theta,\xi)|\sigma(\Theta)] \in \partial f(\Theta)$ ($\mathbf{E}[g(\Theta,\xi)|\sigma(\Theta)] = \nabla f(\Theta)$). This is an *unbiased estimator* of gradient.

$$\boxed{\text{Interested in the number of calls to the stochastic oracle!!}}$$

$$\boxed{\textbf{Question}\text{: How to do we select } g(\Theta,\xi)?}$$

Under natural technical conditions,

Algorithm 1: Stochastic gradient descent for expected loss

```
initialize: $\theta_0 \in \mathbb{R}^n$, $\alpha_0 \in \mathbb{R}$ (not random)
for $k = 0, 1, \ldots$
        Draw $\xi$ randomly from the unknown distribution
        Compute $g(\theta_k,\xi) = \nabla_x \ell(\theta_k,\xi)$
        Set $\theta_{k+1} = \theta_k - \alpha_k \nabla_x \ell(\theta_k,\xi)$
        Update $\alpha_{k+1}$
end
```

First, let's check that in both cases we are getting an unbiased estimator of the gradient:

- (Empirical risk) $\mathbf{E}[\nabla f_I(\Theta_k)|\sigma(\Theta_k)] = \frac{1}{m}\sum_{i=1}^m \nabla f_i(\Theta_k)$.
- (Expected loss) Under some technical assumptions (ξ and Θ are independent and interchanging of derivatives), $\mathbf{E}[\nabla \ell(\Theta_k;\xi)|\sigma(\Theta_k)] = \nabla \mathbf{E}_\xi[\ell(\cdot;\xi)](\Theta_k)$

$$\boxed{\text{What can go wrong?}}$$

```
Algorithm 2: Stochastic gradient descent for empirical risk
initialize: $x_0 \in \mathbb{R}^n$, $\alpha_0 \in \mathbb{R}$ (not random)
for $k = 0, 1, \ldots$
        Draw uniformly at random $I \in [m]$
        Compute $g(\theta, \xi) := \nabla f_I(\theta)$
        Set $\theta_{k+1} = \theta_k - \alpha_k \nabla f_I(x)$
        Update $\alpha_{k+1}$
end
```

- Stepsize- fixed or decreasing. We can not expect a fixed stepsize because even if at the optimal we will still take a step.
- Control the fluctuations of $g(\theta, \xi)$- (nonsmooth case) we will have to assume $\mathbf{E}[\|g(\theta,\xi)\|^2 \,|\theta] < B^2$ and in the smooth case $\mathbf{E}[\|g(\theta,\xi) - \nabla f(\theta)\|^2 \,|\theta] \le \sigma^2$
- Difference between Algorithm 1 and Algorithm 2: Suppose we only have m iid samples $\xi_1, \ldots, \xi_m$ and we run Algorithm 1. How many times can we run Algorithm 1? How many times can we run Algorithm 2? For Algorithm 1, we can only run it m times. Why? Consider Θ_{m+1}. Then Θ_{m+1} contains information about $\xi_1, \ldots, \xi_m$. Therefore, if I generate my next ξ it will not be independent of Θ_{m+1} and thus the conditional expectation doesn't match. Whereas in Algorithm 2 can run it infinitely often.

Class	Stepsize	Rate
L-smooth, $\mathbf{E}[\|\nabla f(\Theta) - g(\Theta,\xi)\|^2 \,\|\Theta] \le \sigma^2$	$\min\{1/L, \frac{\varepsilon^2}{L\sigma}\}$	$\mathbf{E}[\|\nabla f(\Theta_k)\|^2] \le \varepsilon^2$
convex (not diff), $\mathbf{E}[\|g(\Theta)\|^2 \,\|\Theta] \le B^2$	$\frac{R}{B}\sqrt{\frac{2}{T}}$	$\mathbf{E}\left[f\left(\frac{1}{T}\sum_{k=1}^T \Theta_k\right)\right] - f^* \le RB\sqrt{\frac{2}{T}}$
ℓ-strongly cvx (not diff), $\mathbf{E}[\|g(\Theta)\|^2 \,\|\Theta] \le B^2$	$\frac{2}{\ell(k+1)}$	$\mathbf{E}\left[f\left(\sum_{k=1}^T \frac{2k}{T(T+1)}\Theta_k\right)\right] - f^* \le \frac{2B^2}{\ell(k+1)}$
cvx, L smooth, $\mathbf{E}[\|\nabla f(\Theta) - g(\Theta,\xi)\|^2 \,\|\Theta] \le \sigma^2$	$\frac{1}{L+1/\eta}$, $\eta = \frac{R}{\sigma}\sqrt{\frac{2}{T}}$	$\mathbf{E}f\left(\frac{1}{T}\sum_{k=1}^T \Theta_k\right) - f^* \le R\sigma\sqrt{\frac{2}{T}} + \frac{LR^2}{T}$
ℓ strong cvx, L-smooth, $\mathbf{E}[\|\nabla f(\Theta) - g(\Theta,\xi)\|^2 \,\|\Theta] \le \sigma^2$	$\frac{2}{\ell\sigma(1+k)}$	$\mathbf{E}[f(\Theta_k) - f^*] \le O(1)\frac{1}{1+k}$

Here $R = \|x_1 - x^*\|$.

Lemma 2.2. *Suppose f is L-smooth. Then the iterates of SGD satisfy the following inequality:*

$$\mathbf{E}[f(\Theta_{k+1})|\Theta_k] - f(\Theta_k) \le -\alpha_k \|\nabla f(\Theta_k)\|^2 + \frac{L}{2}\alpha_k^2 \mathbf{E}[\|g(\Theta_k, \xi_k)\|^2 \,|\Theta_k].$$

Proof. By L-smoothness of the function f, we have that

$$\begin{aligned} f(\theta_{k+1}) - f(\theta_k) &\le \nabla f(\theta_k)^T(\theta_{k+1} - \theta_k) + \frac{L}{2}\|\theta_{k+1} - \theta_k\|^2 \\ &= -\alpha_k \nabla f(\theta_k)^T g(\theta_k, \xi_k) + \frac{L}{2}\alpha_k^2 \|g(\theta_k, \xi_k)\|^2 \end{aligned}$$

Let's take the conditional expectation with respect to Θ_k:

$$\begin{aligned} \mathbf{E}[f(\Theta_{k+1})|\Theta_k] - f(\Theta_k) &\le -\alpha_k \nabla f(\Theta_k)^T \mathbf{E}[g(\Theta_k, \xi_k)|\Theta_k] + \frac{L}{2}\alpha_k^2 \mathbf{E}[\|g(\Theta_k, \xi_k)\|^2 \,|\Theta_k] \\ \text{(unbiased gradient estimator)} \quad &= -\alpha_k \|\nabla f(\Theta_k)\|^2 + \frac{L}{2}\alpha_k^2 \mathbf{E}[\|g(\Theta_k, \xi_k)\|^2 \,|\Theta_k] \end{aligned}$$

□

In order to deter the harmful affects of the second term, we impose an assumption on the boundedness of the variance of $g(\Theta)$:

$$\mathbf{E}[\|g(\Theta,\xi)\|^2 \,|\Theta] - \|\nabla f(\Theta)\|^2 \leq \sigma^2. \tag{2.1}$$

Assumption 2.3 (Strong convexity)**.** *A function f is strongly convex if there exists a constant $\ell > 0$ such that*

$$f(y) \geq f(\theta) + \nabla f(\theta)^T (y-\theta) + \frac{\ell}{2}\|y-\theta\|^2 \quad \textit{for all } y, \theta.$$

In particular, this implies that

$$2\ell(f(\theta) - f^*) \leq \|\nabla f(\theta)\|^2.$$

Theorem 2.4 (Strong cvx, "SGD convergence" with fixed stepsize)**.** *Suppose f is ℓ-strongly convex and L-smooth. Let $\mathbf{E}[\|g(\Theta,\xi)\|^2 \,|\Theta] \leq \sigma^2 + \|\nabla f(\Theta)\|^2$ and $0 < \alpha \leq \frac{1}{L}$ ($\alpha_k = \alpha$ for all k). Then*

$$\begin{aligned}\mathbf{E}[f(\Theta_k) - f^*] &\leq \frac{\alpha L \sigma^2}{2\ell} + (1-\alpha\ell)^{k-1}\left(f(\theta_1) - f^* - \frac{\alpha L\sigma^2}{2\ell}\right)\\ &\Rightarrow_{k\to\infty} \frac{\alpha L\sigma^2}{2\ell}.\end{aligned}$$

Proof. The previous Lemma and (2.1) we have the following bound:

$$\begin{aligned}\mathbf{E}[f(\Theta_{k+1})|\Theta_k] - f(\Theta_k) &\leq -\alpha\|\nabla f(\Theta_k)\|^2 + \frac{L}{2}\alpha^2\|\nabla f(\Theta_k)\|^2 + \frac{L\alpha^2\sigma^2}{2}\\ &\leq -\frac{\alpha}{2}\|\nabla f(\Theta_k)\|^2 + \frac{L\alpha^2\sigma^2}{2}\\ \text{strong convexity} \qquad &\leq -\alpha\ell(f(\Theta_k) - f^*) + \frac{L\alpha^2\sigma^2}{2}.\end{aligned}$$

By subtracting $\frac{L\alpha\sigma^2}{2\ell}$ and f^* from both sides, we have that

$$\begin{aligned}\mathbf{E}[f(\Theta_{k+1})|\Theta_k] - f^* - \frac{L\alpha\sigma^2}{2\ell} &\leq (1-\alpha\ell)(f(\Theta_k) - f^*) - \frac{L\alpha\sigma^2}{2\ell} + \frac{L\alpha^2\sigma^2}{2}\\ &= (1-\alpha\ell)\left(f(\Theta_k) - f^* - \frac{L\alpha\sigma^2}{2\ell}\right) + \frac{L\alpha^2\sigma^2}{2}.\end{aligned}$$

Taking expectations and iterating, we obtain the result. □

Rmks:

- The convergence rate breaks into two components:

$$\mathbf{E}[f(\Theta_k) - f^*] \leq \underbrace{\frac{\alpha L\sigma^2}{2\ell}}_{\text{fixed neighborhood}} + \underbrace{(1-\alpha\ell)^{k-1}\left(f(\theta_1) - f^* - \frac{\alpha L\sigma^2}{2\ell}\right)}_{\text{Deterministic linear rate}}$$

- For a fixed stepsize α, SGD converges to a neighborhood determined by the variance of the gradients and the size of the stepsize. To obtain convergence, we need either (1) α to decrease or (2) the variance to decrease.

By running the argument with stepsize α_k instead of α, one can show that the expected optimality gap satisfies

$$\mathbf{E}[f(\Theta_k) - f^*] \leq \mathcal{O}\left(\frac{1}{k}\right)$$

provided the stepsize α_k satisfies

$$\sum_{k=1}^{\infty} \alpha_k = \infty \quad \text{and} \quad \sum_{k=1}^{\infty} \alpha_k^2 < \infty.$$

Question: The convergence rate for SGD is slow compared to full gradient descent, what could we do to improve this rate?

We could add more samples. What does this do...reduces the variance. Suppose instead of uniformly choosing index $I \in [m]$, we choose $|S|$ number of indices in $[m]$ (Here I am assuming $|S| << m$):

$$\theta_{k+1} = \theta_k - \alpha_k \frac{1}{|S|} \sum_{I \in S} \nabla f_I(\theta_k).$$

When we do this, we reduce the variance σ^2, particularly,

$$\begin{aligned} \mathbf{E}[\| \frac{1}{|S|} \sum_{I \in S} f_I(\theta) - \nabla f(\theta) \|^2 | \theta] &= \frac{1}{|S|^2} \mathbf{E}[\| \sum_{I \in S} (\nabla f_i(\theta) - \nabla f(\theta)) \|^2 | \theta] \\ \text{independence} \quad &= \frac{1}{|S|} \mathbf{E}[\| \nabla f_I(\Theta) - \nabla f(\Theta) \|^2 | \theta] \leq \frac{\sigma^2}{|S|} =: \sigma^2_{\text{mini}} \end{aligned}$$

Great! We reduced variance, but do we lose anything? Yes, we do. Each iteration is much more expense. Suppose we can only call the stochastic oracle T times. Thus, it is $T/|S|$ iterations of minibatch we can call.

- For minibatch SGD with $\sigma^2 = \frac{\sigma^2}{|S|}$ and fixed $\alpha > 0$,

$$\begin{aligned} \mathbf{E}[f(\Theta_k) - f^*] &\leq \frac{\alpha L \sigma^2}{2\ell |S|} + (1 - \alpha \ell))^{k-1} \left(f(\theta_1) - f^* - \frac{\alpha L \sigma^2}{2\ell |S|} \right) = \frac{\alpha L \sigma^2}{2\ell |S|} + \varepsilon. \\ \Rightarrow \quad \frac{T}{|S|} &= \frac{1}{\alpha \ell} \log(1/\varepsilon) \end{aligned}$$

- For simple SGD we could set $\alpha = \alpha/|S|$

$$\begin{aligned} \mathbf{E}[f(\Theta_k) - f^*] &\leq \frac{\alpha L \sigma^2}{2\ell |S|} + \left(1 - \frac{\alpha \ell}{|S|}\right)^{k-1} \left(f(\theta_1) - f^* - \frac{\alpha L \sigma^2}{2\ell |S|} \right) = \frac{\alpha L \sigma^2}{2\ell |S|} + \varepsilon. \\ \Rightarrow \quad T &= \frac{|S|}{\alpha \ell} \log(1/\varepsilon) \end{aligned}$$

So it doesn't pay in this case to use minibatching. The speed up you get the optimization problem due to reducing variance does not outweigh the cost.

2.1 Practical consideration

- minibatching may be beneficial if the cost per iteration to compute the minibatch is reduced for e.g. distribute across multiple processors. But, your minibatch $|S| << m$.
- Although SGD converges with stepsize decreasing, people generally run SGD until they see it stabilize. Then they shrink the step size parameter by some constant.

2.2 Why does SGD work so well in ML? Possible explanations

Intuitive Motivations:

- SGD uses information more efficiently! Consider a training set, called $\mathcal{S}$, which consists of ten copies of the set $\mathcal{S}_0$. A minimizer of the empirical risk over the larger set $\mathcal{S}$ is clearly a minimizer of the smaller set $\mathcal{S}_0$. If we apply "batch" approach to minimize the empirical risk, each iteration would be *10* times more expensive. SGD performs the same computation in both scenarios.
- Convergence rate in strongly convex case. SGD has a sublinear convergence of $1/\varepsilon$ where as full gradient descent has convergence $\log(1/\varepsilon)$ but costs m number of stochastic gradients. The total number cost of full gradient is $m\log(1/\varepsilon)$. However if m is very large then these might be comparable.
- **Formalize: Work Complexity for Large-Scale Learning.** Suppose we have *infinite* number of samples but a limited computational budget, $\mathcal{T}_{\max}$.

  Question: Why might stochastic optimization be better than deterministic?

  Let $\theta^* \in$ argmin R (Expected loss) and $\theta_m \in$ argmin R_m (Empirical loss with m samples), $\tilde{\theta_m}$ output of the optimization algorithm on the empirical loss problem. Set ε be the accuracy of our optimization on the empirical loss, $\mathbf{E}[R_m(\tilde{\theta_m}) - R_m(\theta_m)] \leq \varepsilon$.

$$E[R(\tilde{\theta}_m)] = \underbrace{R(\theta^*)}_{\text{approximation error-choosing } h} + \underbrace{\mathbf{E}[R_m(\theta_m) - R(\theta^*)]}_{\text{estimation error-samples}} + \underbrace{\mathbf{E}[R_m(\tilde{\theta}_m) - R_m(\theta_m)]}_{\text{Optimization error}}$$

$$\approx R(\theta^*) + \frac{1}{m} + \varepsilon \qquad \text{(by Law of Large Numbers).}$$

  Goal: Minimize $\mathbf{E}[R(\tilde{\theta}_m) - E(\theta^*)]$ subject to $\mathcal{T} \leq \mathcal{T}_{\max}$.

  - SGD: ε accuracy requires a total computing time of $\mathcal{T} = \frac{1}{\varepsilon} \Rightarrow \varepsilon = \frac{1}{\mathcal{T}_{\max}}$.

$$\min_m \mathbf{E}[R(\tilde{\theta}_m) - R(\theta^*)] = \min_m \frac{1}{m} + \frac{1}{\mathcal{T}_{\max}} = \frac{1}{\mathcal{T}_{\max}}.$$

    So should choose m to be as large as possible to minimizes the above!

  - Gradient Descent: ε accuracy requires a total computing time of $\mathcal{T} = m\log(1/\varepsilon) \Rightarrow m = \mathcal{T}_{\max}/\log(1/\varepsilon)$.

$$\min_\varepsilon \mathbf{E}[R(\tilde{\theta}_m) - R(\theta^*)] = \min_\varepsilon \frac{\log(1/\varepsilon)}{\mathcal{T}_{\max}} + \varepsilon = \frac{\log(\mathcal{T}_{\max})}{\mathcal{T}_{\max}} + \frac{1}{\mathcal{T}_{\max}}.$$

- Observed that SGD "generalizes" well. Namely SGD converges to "better optimum" then other more advance methods.

2.3 SGD variants and adaptive stepsizes

2.3.1 SGD with momentum

Introduce a velocity vector

$$\begin{aligned} v_k &= \alpha v_{k-1} - \varepsilon g(\theta_k, \xi_k) \\ \theta_{k+1} &= \theta_k + v_k \end{aligned}$$

$\alpha \geq \varepsilon$ then the current iterate affected more by past gradients. Typical α are 0.8 or 0.9.

2.3.2 AdaGrad-Adaptive Stepsizes

Motivation-Up to now we've assigned the same learning rate to all features. If the contours of the function f looks like this

Easy to minimize!

If the contours looks like this,

Hard to minimize!

If the features vary in importance and frequency, maybe we should vary the stepsize to reflect this!

This idea is related to preconditioning, i.e. you want to find a transformation which takes you from hard to easy case.

Idea For each "feature" downscale by the square-root of sum of squares of its historical values. Set $r_0 = 0$ $\delta > 0$ to prevent it from being 0 and α global learning rate.

$$\begin{aligned} \text{(accumulates past gradients)} \quad r_k &= r_{k-1} + g(\theta_k, \xi_k) \odot g(\theta_k, \xi_k) \\ \theta_{k+1} &= \theta_k - \frac{\alpha}{\delta + \sqrt{r_k}} \odot g(\theta_k, \xi_k). \end{aligned}$$

Agressive stepsize goes to 0. Comes from finding the optimum learning rate using regret bounds. RMSProp/Adam variants of this.

Side note: Where does this come from?
Regret minimization: The algorithm predicts θ_k and incurs loss of $\ell_{i_k}(\theta_k)$

$$R(T) = \sum_{k=1}^{T} f_{i_k}(\theta_k) - \inf_{\theta} \sum_{k=1}^{T} f_{i_k}(\theta).$$

- SGD:

$$\theta_{k+1} = \theta_k - \alpha g(\theta_k) \quad \Rightarrow \quad \sum_{k=1}^{T} f_{i_k}(\theta_k) - f_{i_k}(\theta^*) \le \frac{1}{2\alpha} \|\theta_1 - \theta^*\|^2 + \frac{\alpha}{2} \sum_{k=1}^{T} \|g(\theta_k)\|^2$$

- Preconditioning:

$$\theta_{k+1} = \theta_k - \alpha A^{-1} g(\theta_k) \quad \Rightarrow \quad \sum_{k=1}^{T} f_{i_k}(\theta_k) - f_{i_k}(\theta^*) \le \frac{1}{2\alpha} \|\theta_1 - \theta^*\|^2 + \frac{\alpha}{2} \sum_{k=1}^{T} \|g(\theta_k)\|_{A^{-1}}^2$$

What if we minimize upper bound on regret w.r.t. A, $\min_A g(\theta_k)^T A^{-1} g(\theta_k)$. The solution is $A = c\left(\sum_{k=1}^{T} g(\theta_k) g(\theta_k)^T\right)^{1/2}$. Can't actually use this because it has future gradients in it- i.e. at iteration 2 you don't have the gradient at iteration 10. Instead approximate this by $A_k = \left(\sum_{\tau=1}^{k} g(\theta_\tau) g(\theta_\tau)^T\right)^{1/2}$ and then take the diagonal.

3 Noise Reduction

Let's focus on the empirical risk problems:

$$\min_{\theta} \frac{1}{m} \sum_{i=1}^{m} f_i(\theta) =: f(\theta)$$

where f_i's are L-smooth and f is ℓ-strongly convex.

Looking back at SGD we have

$$\mathbf{E}[f(\Theta_k) - f^*] \le \frac{\alpha L \sigma^2}{2\ell} + (1 - \alpha\ell)^{k-1} \left(f(\theta_1) - f^* - \frac{\alpha L \sigma^2}{2\ell}\right).$$

Noisy gradients do affect SGD which prevents us from converging in the fixed stepsize case or gives us sublinear rate in the diminishing stepsize rate. With GD, we obtain a linear rate $\mathcal{O}(m\frac{L}{\ell}\log(1/\varepsilon))$.

Question: Can one get something in between?

3.1 SVRG.

We clearly see that when m is much much larger than κ then SGD is better. When κ is much much larger than m, we saw that doing the naive thing of reducing variance by increasing batch size did not work. Can we do something more clever in this regime to beat SGD?

Rather than reducing variance by computing *new* stochastic gradient information, like in minibatching, can we *reuse/revise* previously computed information? After all, if the current gradient isn't too far from the previous then the gradient information might still be good. What's the cost?- either increase a little bit in computation of SGD or increase the storage.

Idea of SVRG (increase the cost): Center the output of the stochastic oracle in order to reduce variance:

- SGD: feed in $\nabla f_I(\theta)$
- SVRG: $\nabla f_i(\theta) - \nabla f_i(\hat{\theta}) + \nabla f(\hat{\theta})$ (increase in computational cost because computing full gradient).

Why? $\nabla f_i(\theta) - \nabla f_i(\hat{\theta})$ is small when θ and $\hat{\theta}$ are close like near the optimum and should have small variance. Moreover, $\nabla f(\hat{\theta})$ is small near the optimum.

Lemma 3.1.

$$\mathbf{E}[\|\nabla f_I(\theta) - \nabla f_I(\theta^*)\|^2] \leq 2L(f(\theta) - f(\theta^*))$$

Proof. Set $z := \theta^* - \frac{1}{L}\left(\nabla f_I(\theta) - \nabla f_I(\theta^*)\right)$. By L-smoothness, we have that

$$f_I(z) - f_I(\theta) \leq \nabla f_I(\theta)^T(z-\theta) + \frac{L}{2}\|z-\theta\|^2,$$

and by convexity,

$$f_I(\theta^*) - f_I(z) \leq \nabla f_I(\theta^*)^T(\theta^* - z).$$

Using these two bounds, we derive the following result

$$\begin{aligned}
f_I(\theta^*) - f_I(\theta) &= f_I(\theta^*) - f_I(z) + f_I(z) - f_I(\theta) \\
&\leq \nabla f_I(\theta^*)^T(\theta^* - z) + \nabla f_I(\theta)^T(z-\theta) + \frac{L}{2}\|z-\theta\|^2 \\
\text{(add/sub } \theta) \quad &= \nabla f_I(\theta^*)^T(\theta^* - \theta) + (\nabla f_I(\theta) - \nabla f_I(\theta^*))^T(z-\theta) + \frac{L}{2}\|z-\theta\|^2 \\
\text{(plug in } z) \quad &= \nabla f_I(\theta^*)^T(\theta^* - \theta) - \frac{1}{2L}\|\nabla f_I(\theta) - \nabla f_I(\theta^*)\|^2
\end{aligned}$$

By taking expectations conditioned on θ and by observing that $\mathbf{E}[\nabla f_I(\theta^*)|\theta] = \nabla f(\theta^*) = 0$, we get that

$$\frac{1}{2L}\mathbf{E}[\|\nabla f_I(\theta) - \nabla f_I(\theta^*)\|^2|\theta] \leq f(\theta) - f(\theta^*)$$

□

However, $\nabla f(\hat{\theta})$ is expensive to compute (requires m gradient computations) so we should only update it very infrequently. This leads to the following epoch idea.

```
Algorithm 3: SVRG
  initialize: $\theta_0 \in \mathbb{R}^n$, $\alpha_0 > 0$ (not random)
  for $s = 1, 2, \ldots$
        Step 1: Compute the batch gradient $\nabla f(\hat{\theta}^{(s)})$
        Set $\hat{\theta}^{(s)} = \theta_1^{(s)}$
        for $t = 1, \ldots, k$
            Step 2: Set
                $$\theta_{t+1}^{(s)} = \theta_t^{(s)} - \alpha \left( \nabla f_{I_t^{(s)}}(\theta_t^{(s)}) - \nabla f_{I_t^{(s)}}(\hat{\theta}^{(s)}) + \nabla f(\hat{\theta}^{(s)}) \right)$$
 where $I_t$ is chosen uniformly from $[m]$.
        end
        Update
                $$\hat{\theta}^{(s+1)} = \frac{1}{k} \sum_{t=1}^{k} \theta_t^{(s)}.$$
  end
```

Theorem 3.2 (Convergence of SVRG)**.** *Suppose $f_1, \ldots, f_m$ are L-smooth functions and f is ℓ strongly convex. Then SVRG with $\alpha = \frac{1}{10L}$ and $k = 20\frac{L}{\ell}$ (number of inner iterations) satisfies*

$$\mathbf{E}[f(\hat{\theta}^{(s+1)})] - f^* \leq 0.9^s (f(\hat{\theta}^{(s)}) - f^*).$$

Proof. Fix $s \geq 1$. I will drop the s in the notation for simplification. By expanding the norm, we obtain

$$\|\theta_{t+1} - \theta^*\|^2 = \|\theta_t - \theta^*\|^2 - \alpha 2 v_t^T (\theta_t - \theta^*) + \alpha^2 \|v_t\|^2$$

where

$$v_t = \nabla f_{I_t}(\theta_t) - \nabla f_{I_t}(\hat{\theta}) + \nabla f(\hat{\theta}).$$

Using the previous Lemma, $\mathbf{E}[\|X - \mathbf{E}(X)\|^2] \leq \mathbf{E}[\|X\|^2]$, $\mathbf{E}[\nabla f_{I_t}(\theta^*)] = \nabla f(\theta^*) = 0$, and $(a+b)^2 \leq 2a^2 + 2b^2$, we obtain

$$\begin{aligned}
\mathbf{E}_{I_t}[\|v_t\|^2] &= \mathbf{E}_{I_t}[\left\|\nabla f_{I_t}(\theta_t) - \nabla f_{I_t}(\theta^*) - \nabla f_{I_t}(\hat{\theta}) + \nabla f(\hat{\theta}) + \nabla f_{I_t}(\theta^*)\right\|^2] \\
&\leq 2\mathbf{E}_{I_t}[\|\nabla f_{I_t}(\theta_t) - \nabla f_{I_t}(\theta^*)\|^2] + 2\mathbf{E}_{I_t}[\left\|\nabla f_{I_t}(\hat{\theta}) - \nabla f_{I_t}(\theta^*) - \nabla f(\hat{\theta})\right\|^2] \\
&\leq 2\mathbf{E}_{I_t}[\|\nabla f_{I_t}(\theta_t) - \nabla f_{I_t}(\theta^*)\|^2] + 2\mathbf{E}_{I_t}[\left\|\nabla f_{I_t}(\hat{\theta}) - \nabla f_{I_t}(\theta^*)\right\|^2] \\
&\leq 4L \left(f(\theta_t) - f(\theta^*) + f(\hat{\theta}) - f(\theta^*) \right)
\end{aligned}$$

Also observe that $\mathbf{E}_{I_t}[v_t^t(\theta_t - \theta^*)] = \nabla f(\theta_t)^T(\theta_t - \theta^*) \geq f(\theta_t) - f(\theta^*)$ by convexity. Plugging this in, we get

$$\mathbf{E}_{I_t}[\|\theta_{t+1} - \theta^*\|^2] \leq \|\theta_t - \theta^*\|^2 - 2\alpha(1 - 2L\alpha)(f(\theta_t) - f(\theta^*)) + 4\alpha^2 L(f(\hat{\theta}) - f(\theta^*)).$$

I'm now going to introduce back the s! Taking conditional expectation (conditioned on $\hat{\theta}$) and summing up from $t = 1, \dots, k$

$$\begin{aligned}
\mathbf{E}[\left\|\theta_{k+1}^{(s)} - \theta^*\right\|^2 |\hat{\theta}^{(s)}] &\le \left\|\theta_1^{(s)} - \theta^*\right\|^2 - 2\alpha(1-2L\alpha)\mathbf{E}[\sum_{t=1}^{k} f(\theta_t^{(s)}) - f(\theta^*)|\hat{\theta}^{(s)}] + 4\alpha^2 Lk\mathbf{E}[f(\hat{\theta}^{(s)}) - f(\theta^*)|\hat{\theta}^{(s)} \\
(\theta_1 = \hat{\theta}^{(s)}) \quad &= \left\|\hat{\theta}^{(s)} - \theta^*\right\|^2 - 2\alpha(1-2L\alpha)\mathbf{E}[\sum_{t=1}^{k} f(\theta_t^{(s)}) - f(\theta^*)|\hat{\theta}^{(s)}] + 4\alpha^2 Lk\mathbf{E}[f(\hat{\theta}^{(s)}) - f(\theta^*)|\hat{\theta}^{(s)} \\
\text{(strong convexity)} \quad &\le \left(\frac{2}{\ell} + 4\alpha^2 Lk\right)\left(f(\hat{\theta}^{(s)}) - f(\theta^*)\right) - 2\alpha(1-2L\alpha)\mathbf{E}[\sum_{t=1}^{k} f(\theta_t^{(s)}) - f(\theta^*)|\hat{\theta}^{(s)}]
\end{aligned}$$

By convexity, $\mathbf{E}\left[f\left(\frac{1}{k}\sum_{i=1}^{k}\theta_t^{(s)}\right)|\hat{\theta}^{(s)}\right] \le \frac{1}{k}\mathbf{E}[\sum_{t=1}^{k} f(\theta_t^{(s)}) - f(\theta^*)|\hat{\theta}^{(s)}]$. Hence because of our choice of α, we have that

$$\mathbf{E}\left[f(\hat{\theta}^{(s+1)})|\hat{\theta}^{(s)}\right] = \mathbf{E}\left[\left(\frac{1}{k}\sum_{i=1}^{k}\theta_t^{(s)}\right)|\hat{\theta}^{(s)}\right] \le \left(\frac{1}{\alpha\ell(1-2L\alpha)k} + \frac{2\alpha L}{1-2L\alpha}\right)\left(f(\hat{\theta}^{(s)}) - f(\theta^*)\right)$$

Just want to choose k and α so that $\left(\frac{1}{\alpha\ell(1-2L\alpha)k} + \frac{2\alpha L}{1-2L\alpha}\right) < 1$. Plugging in $\alpha = \frac{1}{10L}$ and $k = \frac{20L}{\ell}$, we get the result. $\square$

This is great! We now have a linear rate of convergence just like full gradient descent, but did we reduce the cost?

Let's count the cost (i.e number of stochastic gradient evaluations) to compute $\hat{\theta}$ (Note this is the iterate which is converging): Step 2 requires 2 stochastic gradient evaluations while Step 1 requires m (a full gradient) so each iteration costs us $k \cdot 2 + m$. Therefore, the number of iterations to make $\mathbf{E}[f(\hat{\theta}^{(s)})] - f(\theta^*) < \varepsilon$ requires $\mathcal{O}\left((\kappa + m)\log(1/\varepsilon)\right)$ number of stochastic gradient evaluations ($k \approx \kappa$).

Comparison.

Algorithm	Stochastic gradient evaluations ($< \varepsilon$)
SGD, decreasing stepsize	$\mathcal{O}(\kappa/\varepsilon)$
GD, fixed stepsize	$\mathcal{O}\left(\kappa\log(1/\varepsilon)\right)$
SVRG/SAGA/SAG	$\mathcal{O}\left((\kappa + m)\log(1/\varepsilon)\right)$

Following similar analysis as before, when $m >> \kappa$ is very large, SVRG are comparable to GD and therefore can not beat SGD in this regime. On the other hand, when $\kappa >> m$, the regime where GD may be superior, and perhaps easier to tune.

SAGA/SAG. This method is closer in spirit to SGD than SVRG was. Here we are not going to compute batch gradients (expect possibly at the initial point); instead, in each iteration, we compute a stochastic $g(\theta_k)$ as the average of the stochastic gradients evaluated at previous iterates. Specifically, at each iteration k, the method will have stored $\nabla f_i(\theta_{[i]})$ for $i = 1, \dots, m$ where $\theta_{[i]}$ represents the last iterate at which ∇f_i was evaluated. An integer $j \in [m]$ is chosen uniformly and the stochastic vector is set to be

$$g(\theta_k) = \nabla f_j(\theta_k) - \nabla f_j(\theta_{[j]}) + \frac{1}{m}\sum_{i=1}^{m}\nabla f_i(\theta_{[i]}).$$

Note this is an unbiased estimator of the gradient and the variance that are expected to be less than the stochastic gradients in SGD. So what's the problem?- At each iteration, you have to store *m-stochastic gradients*. Note: if $f_i(theta_k) = \ell(x_i^T theta_k)$, then $\nabla f_i(\theta_k) = \hat{f}'(x_i^T \theta_k)x_i$. Since the feature vectors are already available in storage, one need only store the scalar $\hat{f}'(x_i^T \theta_k)$. Such functional forms of f_i occur in logistic and least squares.