LECTURE NOTES ON STOCHASTIC GRADIENT DESCENT *

A. A. ERGÜR, V. T. NGUYEN

Abstract. This scribe contains lecture notes on Stochastic Gradient Descent

Key words. Stochastic Gradient Descent

AMS subject classifications. safely ignore

1. Stochastic Gradient Descent (SGD). Given a dataset $\mathcal{S} = \{\mathbf{x}_i, y_i\}_{i=1}^N$, where each feature vector $\mathbf{x}_i \in \mathbb{R}^D$. We define a loss function $f(.)$ which measures the error between the prediction and the true label for every single sample i-th as

$$f_{\boldsymbol{\theta}} : \mathbb{R}^D \times \mathbb{R} \to \mathbb{R}.$$

The average error which calculated on the entire dataset $\mathcal{S}$ is called cost function, i.e.,

$$C(\boldsymbol{\theta}, \mathbf{X}, \mathbf{y}) = \frac{1}{N} \sum_{i=1}^{N} f(\mathbf{x}_i, y_i; \boldsymbol{\theta}),$$

where $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]^\top, \mathbf{y} = [y_1, \ldots, y_N]^\top$. For training machine learning, we want to find the parameters $\boldsymbol{\theta} \in \Omega$ that minimize the cost function, i.e.,

$$\hat{\boldsymbol{\theta}} = \text{argmin}_{\boldsymbol{\theta} \in \Omega} C(\boldsymbol{\theta}, \mathbf{X}, \mathbf{y})$$

1.1. Recall Gradient descent. Supposedly, $f(\cdot)$ is differentiable, then the gradient of the cost function wrt $\boldsymbol{\theta}$ is given as

$$\nabla_{\boldsymbol{\theta}} C(\boldsymbol{\theta}, \mathbf{X}, \mathbf{y}) = \frac{1}{N} \sum_{i=1}^{N} \nabla_{\boldsymbol{\theta}} f(\mathbf{x}_i, y_i; \boldsymbol{\theta})$$

Then the update parameters follow as

$$\boldsymbol{\theta}^{(k)} = \boldsymbol{\theta}^{(k)} - \alpha^{(k)} \nabla_{\boldsymbol{\theta}} C(\boldsymbol{\theta}, \mathbf{X}, \mathbf{y}).$$

Since computing the gradient on the entire dataset is costly, people refer stochastic methods that approximate the $\nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta}, \mathbf{X}, \mathbf{y})$, with fewer computational effort at every step. That is Stochastic Gradient Descent.

1.2. SGD algorithm. The idea of the stochastic gradient descent algorithm is to build a function $g(\cdot)$ which approximates $C(\cdot)$ and in the mean, $g(\cdot)$ behaves like $C(\cdot)$, i.e.,

$$\mathbb{E}\{g(\mathbf{x}, y; \boldsymbol{\theta})\} = C(\mathbf{x}, y; \boldsymbol{\theta}),$$
$$\mathbb{E}\{\nabla_{\boldsymbol{\theta}} g(\mathbf{x}, y; \boldsymbol{\theta})\} = \nabla_{\boldsymbol{\theta}} C(\mathbf{x}, y; \boldsymbol{\theta}).$$

Then, we can update the model weights as

$$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{k} - \alpha^{(k)} \nabla_{\boldsymbol{\theta}} g(\mathbf{x}^{(k)}, y^{(k)}; \boldsymbol{\theta}^{(k)}). \tag{1.1}$$

*We thank Robbins family for supporting the Algorithmic Foundations of Data Science Course

This manuscript is for review purposes only.

Algorithm 1.1 SGD: Stochastic Gradient Descent

Require: $\alpha^{(k)}$: Learning rate at iteration k
Require: $g(\mathbf{x}, y; \boldsymbol{\theta})$: Stochastic approximation of the cost function
Require: $\boldsymbol{\theta}^{(0)}$: Initial model parameters

```
for k = 0, 1, 2, ... until convergence do
    Sample data point (x^(k), y^(k))
    Compute stochastic gradient ∇_θ g(x^(k), y^(k); θ^(k))
    Update parameters:
        θ^(k+1) ← θ^(k) − α^(k) ∇_θ g(x^(k), y^(k); θ^(k))
end for
return θ^(k+1)
```

$$\boldsymbol{\theta}^{(k+1)} \leftarrow \boldsymbol{\theta}^{(k)} - \alpha^{(k)} \nabla_{\boldsymbol{\theta}} g(\mathbf{x}^{(k)}, y^{(k)}; \boldsymbol{\theta}^{(k)})$$

Stochastic gradient descent for linear regression. Recall that the cost function of linear regression is given as

$$C(\mathbf{X}, \mathbf{y}; \boldsymbol{\theta}) = \frac{1}{N} \sum_{i=1}^{N} (\mathbf{x}_i \boldsymbol{\theta} - y_i)^2 = \frac{1}{N} ||\mathbf{X}\boldsymbol{\theta} - \mathbf{y}||_2^2$$

By calculation, we have

$$\nabla_{\boldsymbol{\theta}} C(\mathbf{X}, \mathbf{y}; \boldsymbol{\theta}) = \frac{2}{N} \left(\mathbf{X}^\top \mathbf{X} \boldsymbol{\theta} - \mathbf{X}^\top \mathbf{y} \right)$$

Instead of using $C(\cdot)$ and its gradient, we can use the alternative cost function $g(\cdot)$ as

$$g(\mathbf{x}, y; \boldsymbol{\theta}; i) = (\mathbf{x}_i^\top \boldsymbol{\theta} - y_i)^2,$$

where i is a random variable that follows a uniform distribution, i.e., $\Pr(i) = \frac{1}{N}$. We can prove that in the mean, the expectation of $g(\cdot)$ equals to $C(\cdot)$ as

$$\begin{aligned}
\mathbb{E}_i\{g(\mathbf{x}, y; \boldsymbol{\theta})\} &= \sum_{i=1}^{N} \Pr(i) (\mathbf{x}_i^\top \boldsymbol{\theta} - y_i)^2 \\
&= \sum_{i=1}^{N} \frac{1}{N} (\mathbf{x}_i^\top \boldsymbol{\theta} - y_i)^2 \\
&= \sum_{i=1}^{N} \frac{1}{N} ||\mathbf{X}\boldsymbol{\theta} - \mathbf{y}||_2^2 \\
&\triangleq C(\mathbf{X}, \mathbf{y}; \boldsymbol{\theta}).
\end{aligned}$$

Similarly, we have $\nabla_{\boldsymbol{\theta}} g(\mathbf{x}, y; \boldsymbol{\theta}; i) = 2\left(\mathbf{x}_i\right)\left(\mathbf{x}_i^\top \boldsymbol{\theta} - y_i\right)$. Then consider the expectation of $\nabla_{\boldsymbol{\theta}} g(\mathbf{x}, y; \boldsymbol{\theta}; i)$ over i, we have:

$$\begin{aligned}
\mathbb{E}_i\{\nabla_{\boldsymbol{\theta}} g(\mathbf{x}, y; \boldsymbol{\theta}; i)\} &= \mathbb{E}_i\{2\left(\mathbf{x}_i\right)\left(\mathbf{x}_i^\top \boldsymbol{\theta} - y_i\right)\} \\
&= \sum_{i=1}^{N} 2\frac{1}{N}\mathbf{x}_i(\mathbf{x}_i^\top \boldsymbol{\theta} - y_i) \\
&= \frac{2}{N}\sum_{i=1}^{N} \mathbf{x}_i(\mathbf{x}_i^\top \boldsymbol{\theta} - y_i) \\
&= \frac{2}{N}\left(\mathbf{X}^\top \mathbf{X}\boldsymbol{\theta} - \mathbf{X}\mathbf{y}\right) \triangleq \nabla_{\boldsymbol{\theta}} C(\mathbf{X}, \mathbf{y}; \boldsymbol{\theta})
\end{aligned}$$

Hinge loss function. Given dataset $\mathcal{S} = \{\mathbf{x}_i, y_i\}_{i=1}^N$, where $\mathbf{x}_i \in \mathbb{R}^D, y_i \in \{-1, 1\}$. We want to find the hyperplane parameterized by $\boldsymbol{\theta} \in \mathbb{R}^D$, such that

$$C(\mathbf{X}, \mathbf{y}; \boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N} \alpha \max\left(1 - y_i \mathbf{x}_i^\top \boldsymbol{\theta}, 0\right) + \frac{\lambda}{2}||\boldsymbol{\theta}||_2^2$$

is minimized, where α, λ are hyper-parameters. Instead of using the above cost function $C(\cdot)$ to do optimization, we can use the following approximation, i.e.,

$$g(\mathbf{x}, y; \boldsymbol{\theta}; i) = \alpha \max\left(1 - y_i \mathbf{x}_i^\top \boldsymbol{\theta}, 0\right) + \frac{\lambda}{2}||\boldsymbol{\theta}||_2^2.$$

Again, the index i is chosen uniformly in $\{1, \ldots, N\}$. Then we have

$$\mathbb{E}_i\{g(\mathbf{x}, y; \boldsymbol{\theta})\} = \sum_{i=1}^{N} \frac{1}{N}\alpha \max\left(1 - y_i \mathbf{x}_i^\top \boldsymbol{\theta}, 0\right) + \frac{\lambda}{2}||\boldsymbol{\theta}||_2^2 \triangleq C(\mathbf{X}, \mathbf{y}; \boldsymbol{\theta}).$$

The gradient of $g(\cdot)$ is given as

$$\nabla_{\boldsymbol{\theta}} g(\mathbf{x}, y; \boldsymbol{\theta}; i) = \alpha \max(-y_i \mathbf{x}_i, \mathbf{0}) + \lambda\boldsymbol{\theta}.$$

The gradient of $C(\cdot)$ is given as

$$\nabla_{\boldsymbol{\theta}} C(\mathbf{X}, \mathbf{y}; \boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N} \alpha \max(-y_i \mathbf{x}_i, \mathbf{0}) + \lambda\boldsymbol{\theta}.$$

Obviously, we have $\mathbb{E}_i\{\nabla_{\boldsymbol{\theta}} g(\mathbf{x}, y; \boldsymbol{\theta}; i)\} \triangleq \nabla_{\boldsymbol{\theta}} C(\mathbf{X}, \mathbf{y}; \boldsymbol{\theta})$.

1.3. Loss function is convex. Given a convex function $f(\cdot)$, and we apply SGD as presented in (1.1), with the approximate function $g(\mathbf{x}, y; \boldsymbol{\theta}; i) = f(\mathbf{x}_i, y_i; \boldsymbol{\theta})$, where the index $i \in [1, N]$ chosen uniformly, and supposedly $\hat{\boldsymbol{\theta}}$ is the optimal point, substracting both sides for the optimal point and take their squared magnitudes, we have

$$||\boldsymbol{\theta}^{(k+1)} - \hat{\boldsymbol{\theta}}||_2^2 = ||\left(\boldsymbol{\theta}^{(k)} - \hat{\boldsymbol{\theta}}\right) - \alpha^{(k)}\nabla_{\boldsymbol{\theta}} f(\mathbf{x}_i^{(k)}, y_i^{(k)}; \boldsymbol{\theta}^{(k)})||_2^2.$$

The above equality is expressed as

$$\begin{aligned}
||\boldsymbol{\theta}^{(k+1)} - \hat{\boldsymbol{\theta}}||_2^2 - ||\boldsymbol{\theta}^{(k)} - \hat{\boldsymbol{\theta}}||_2^2 = &\left(\alpha^{(k)}\right)^2 ||\nabla_{\boldsymbol{\theta}} f(\mathbf{x}_i^{(k)}, y_i^k; \boldsymbol{\theta}^{(k)})||_2^2 \\
&- 2\alpha^{(k)}\nabla_{\boldsymbol{\theta}}^\top f(\mathbf{x}^{(k)}, y^k; \boldsymbol{\theta}^{(k)})(\boldsymbol{\theta}^{(k)} - \hat{\boldsymbol{\theta}}).
\end{aligned}$$

Since

$$\begin{aligned}\mathbb{E}_i\{\nabla_{\boldsymbol{\theta}}^\top f(\mathbf{x}_i^{(k)}, y_i^k; \boldsymbol{\theta}^{(k)})(\boldsymbol{\theta}^{(k)} - \hat{\boldsymbol{\theta}})\} &= \mathbb{E}_i\{\nabla_{\boldsymbol{\theta}}^\top f(\mathbf{x}_i^{(k)}, y_i^{(k)}; \boldsymbol{\theta}^{(k)})\}(\boldsymbol{\theta}^{(k)} - \hat{\boldsymbol{\theta}}) \\ &= \mathbb{E}\left\{\nabla^\top f(\mathbf{x}^{(k)}, y^{(k)}; \boldsymbol{\theta}^{(k)}) \left(\boldsymbol{\theta}^{(k)} - \hat{\boldsymbol{\theta}}\right)\right\},\end{aligned}$$

and

$$\mathbb{E}_i\{||\nabla_{\boldsymbol{\theta}} g(\mathbf{x}, y; \boldsymbol{\theta}^{(k)}); i||^2\} = \mathbb{E}\{||\nabla_{\boldsymbol{\theta}} f(\mathbf{x}, y; \boldsymbol{\theta}^{(k)})||^2\}$$

, we have

...

1.4. Randomized Kaczmarz method. This problem is set up that we are given a training dataset $\{\mathbf{A}, \mathbf{b}\} \in \mathbb{R}^{N\times D} \times \mathbb{R}^N$ which establish the linear system:

$$\mathbf{A}\mathbf{x} = \mathbf{b}.$$

Our goal is to find the best $\hat{\mathbf{x}}$, which is the root of the system. If $\mathbf{b} \in \mathrm{Col}\,(\mathbf{A})$, then the linear system above has one or more solutions; otherwise, there is no solution for the system above. Instead of dealing with the two cases above, we can solve an optimization problem instead, i.e.,

$$\hat{\mathbf{x}} = \mathrm{argmin}_{\mathbf{x}\in\mathbb{R}^D} ||\mathbf{A}\mathbf{x} - \mathbf{b}||_2^2.$$

The Randomized Kaczmarz [6] is an iterative method for solving the complex-value linear system. The Randomized Kaczmarz starts with initializing $\mathbf{x}_0$ to be an arbitrary complex-valued initial approximation, then the update formula is given as

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \frac{b_i - \langle \mathbf{a}_i^{(k)}, \mathbf{x}^{(k)}\rangle}{||\mathbf{a}_i^{(k)}||_2^2}\mathbf{a}_i^{(k)},$$

with $\Pr\{i = k\} = \frac{||\mathbf{a}_k||_2^2}{\sum_{i=1}^N ||\mathbf{a}_k||_2^2}$.

Let $\kappa(\mathbf{A}) = ||\mathbf{A}^{-1}||||\mathbf{A}||_F = \frac{1}{\sigma_{\min}(\mathbf{A})}||\mathbf{A}||_F$, then at the k-th iteration, it is guaranteed that

$$\mathbb{E}\{||\mathbf{x}_k - \hat{\mathbf{x}}||_2^2\} \le \left(1 - \kappa(\mathbf{A})^{-2}\right)^k ||\mathbf{x}_0 - \hat{\mathbf{x}}||_2^2$$

Proof. Since $\sum_{i=1}^m |\langle \mathbf{z}, \mathbf{a}_i\rangle|^2 \le \sigma_{\min}(\mathbf{A})^2||\mathbf{z}||^2$, and $||\mathbf{A}||_F^2 = \sum_{i=1}^m ||a_i||_2^2$ then

$$\begin{aligned}&\sum_{i=1}^m \frac{1}{||\mathbf{A}||_F^2}|\langle \mathbf{z}, \mathbf{a}_i\rangle|^2 \ge \kappa(\mathbf{A})^{-1}||\mathbf{z}||^2 \\ \Leftrightarrow &\sum_{i=1}^m \frac{||\mathbf{a}_i||_2^2}{||\mathbf{A}_F^2||}|\langle \mathbf{z}, \frac{\mathbf{a}_i}{||\mathbf{a}_i||_2}\rangle|^2 \ge \kappa(\mathbf{A})^{-2}||\mathbf{z}||^2.\end{aligned}$$

Let $\Pr\,(e = i) = \frac{||\mathbf{a}_i||_2^2}{\sum_{||\mathbf{a}_i||_2^2}}$, and $\mathbf{u}_e = \frac{\mathbf{a}_e}{||\mathbf{a}_e||}$, then

$$\mathbb{E}\{|\langle \mathbf{z}, \mathbf{u}_e\rangle|^2\} \ge \kappa(\mathbf{A})^{-2}||\mathbf{z}||^2.$$

Note that $\mathbf{x}^{(k)}$ is the projection of $\mathbf{x}$ on the span of $\mathbf{a}_{k-1}$, thus

$$||\mathbf{x}^{(k)} - \hat{\mathbf{x}}||^2 = ||\mathbf{x}^{(k)} - \hat{\mathbf{x}}||^2 - ||\mathbf{x}_{k-1} - \mathbf{x}_k||^2$$

, and since $\hat{\mathbf{x}}$ satisfies $\mathbf{A}\mathbf{x} = \mathbf{b}$, then

$$||\mathbf{x}^{(k-1)} - \mathbf{x}^{(k)}||_2^2 = \left\langle \mathbf{x}^{(k-1)} - \hat{\mathbf{x}}, \frac{\mathbf{a}^{(k)}}{||\mathbf{a}^{(k)}||} \right\rangle.$$

Then we have

$$\mathbb{E}_{\{\mathbf{u}_1,\cdots,\mathbf{u}_k\}}||\mathbf{x}^{(k)} - \hat{\mathbf{x}}||^2 \leq \left(1 - \mathbb{E}\{|\langle \mathbf{x}^{(k-1} - \hat{\mathbf{x}}, \mathbf{u}^{(k)}\rangle|^2\}\right) ||\mathbf{x}^{(k-1)} - \hat{\mathbf{x}}||^2$$

And since $\{\mathbf{u}_i\}$ are independent, then

$$\mathbb{E}\{||\mathbf{x}^{(k)} - \hat{\mathbf{x}}||_2^2\} \leq \left(1 - \kappa(\mathbf{A})^{-2}\right) ||\mathbf{x}^{(k-1)} - \hat{\mathbf{x}}||_2^2$$

holds. ∎

2. Quasi-Newton methods.

2.1. Newton method. The Newton method is a second-order method that uses the Hessian of the objective to compute the update in SGD. At every iteration, the new vector $\boldsymbol{\theta}$ is updated as

$$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} - \alpha_k \left(\nabla^2_{\boldsymbol{\theta}} f(\mathbf{x}^{(k)}, y^{(k)}; \boldsymbol{\theta}^{(k)})\right)^{-1} \nabla_{\boldsymbol{\theta}} f(\mathbf{x}^{(k)}, y^{(k)}; \boldsymbol{\theta}^{(k)})$$

Algorithm 2.1 Newton Method for Optimization

Require: α_k: Stepsize at iteration k
Require: $f(\mathbf{x}, y; \boldsymbol{\theta})$: Objective function
Require: $\boldsymbol{\theta}^{(0)}$: Initial parameter vector

for $k = 0, 1, 2, \ldots$ until convergence **do**
  Sample data point $(\mathbf{x}^{(k)}, y^{(k)})$
  Compute gradient $\nabla_{\boldsymbol{\theta}} f(\mathbf{x}^{(k)}, y^{(k)}; \boldsymbol{\theta}^{(k)})$
  Compute Hessian $\nabla^2_{\boldsymbol{\theta}} f(\mathbf{x}^{(k)}, y^{(k)}; \boldsymbol{\theta}^{(k)})$
  Update parameters:

$$\boldsymbol{\theta}^{(k+1)} \leftarrow \boldsymbol{\theta}^{(k)} - \alpha_k \left(\nabla^2_{\boldsymbol{\theta}} f(\mathbf{x}^{(k)}, y^{(k)}; \boldsymbol{\theta}^{(k)})\right)^{-1} \nabla_{\boldsymbol{\theta}} f(\mathbf{x}^{(k)}, y^{(k)}; \boldsymbol{\theta}^{(k)})$$

end for
return $\boldsymbol{\theta}^{(k+1)}$

The Newton method is good but not an ideal one in terms of efficiency since it requires the computation of the Hessian matrix and its inversion. Instead, people invented intermediate methods in the middle of standard SGD and Newton, called Quasi-Newton. Specifically, they used an approximate version of the Hessian matrix whose inverse is easily computed, i.e.,

$$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} - \alpha^{(k)} \left(\mathbf{B}^{(k)}\right)^{-1} \nabla_{\boldsymbol{\theta}} f(\mathbf{x}^{(k)}, y^{(k)}; \boldsymbol{\theta}^{(k)}).$$

, where the $\mathbf{B}^{(k)}$ an *easier* version of the Hessian matrix $\nabla^2_{\boldsymbol{\theta}} f(\mathbf{x}^{(k)}, y^{(k)}; \boldsymbol{\theta}^{(k)})$, where $(\mathbf{B})^{-1}$ is computed quickly but still approximate well the inverse of the Hessian.

Next, we discuss several versions of this Quasi-Newton method, including Adaptive Preconditioned Gradient Descent (AdaGrad), AdaDelta, and Root Mean Square Propagation.

2.2. Adaptive Preconditioned Gradient Descent. The update of the Adaptive Preconditioned Gradient Descent (aka, AdaGrad) [1] is given as

$$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} - \alpha^{(k)} \mathbf{G}^{-1} \nabla_{\boldsymbol{\theta}} f(\mathbf{x}^k, y^k; \boldsymbol{\theta}^{(k)}), \tag{2.1}$$

where

$$\mathbf{G}^{(t)} = \left(\sum_{i=0}^{t} \nabla_{\boldsymbol{\theta}} f\left(\mathbf{x}^{(i)}, y^{(i)}; \boldsymbol{\theta}^{(i)}\right) \left(f\left(\mathbf{x}^{(i)}, y^{(i)}; \boldsymbol{\theta}^{(i)}\right)\right)^{\top} \right)^{1/2}.$$

A practical version of AdaGrad employs the diagonal matrix $\mathbf{D}^{(t)}$, with entries $\sum_{i=1}^{n} \left(||\nabla_{\boldsymbol{\theta}} f(\mathbf{x}^{(t)}, y^{(t)}; \boldsymbol{\theta}^{(t)})|| + \delta \right)$.

2.2.1. Analysis. Obviously $\mathbf{G}^{(k)}$ is a positive semi-definite matrix. By subtracting both sides of (2.1) for $\hat{\boldsymbol{\theta}}$ and taking the squared $\mathbf{A}$-norm[1], we have

$$\begin{aligned}
\left\| \boldsymbol{\theta}^{(k+1)} - \hat{\boldsymbol{\theta}} \right\|^2_{\mathbf{G}^{(k)}} &= \left\| \boldsymbol{\theta}^{(k)} - \alpha^{(k)} \left(\mathbf{G}^{(k)}\right)^{-1} \nabla_{\boldsymbol{\theta}} f\left(\mathbf{x}^{(k)}, y^{(k)}; \boldsymbol{\theta}^{(k)}\right) - \hat{\boldsymbol{\theta}} \right\|^2_{\mathbf{G}^{(k)}} \\
&= \left\| \boldsymbol{\theta}^{(k)} - \hat{\boldsymbol{\theta}} \right\|^2_{\mathbf{G}^{(k)}} - 2\alpha^{(k)} \left\langle \nabla_{\boldsymbol{\theta}} f\left(\mathbf{x}^{(k)}, y^{(k)}; \boldsymbol{\theta}^{(k)}\right), \boldsymbol{\theta}^{(k)} - \hat{\boldsymbol{\theta}} \right\rangle \\
&\quad + \left(\alpha^{(k)}\right)^2 \left\| \left(\mathbf{G}^{(k)}\right)^{-1} \nabla_{\boldsymbol{\theta}} f\left(\mathbf{x}^{(k)}, y^{(k)}; \boldsymbol{\theta}^{(k)}\right) \right\|^2_{\mathbf{G}^{(k)}}.
\end{aligned}$$

Therefore,

$$\begin{aligned}
\nabla^{\top}_{\boldsymbol{\theta}} f(\mathbf{x}^{(k)}, y^{(k)}; \boldsymbol{\theta}^{(k)})(\boldsymbol{\theta}^{(k)} - \hat{\boldsymbol{\theta}}) &= \frac{1}{2\alpha^{(k)}} \left(||\boldsymbol{\theta}^{(k)} - \hat{\boldsymbol{\theta}}||^2_{\mathbf{G}^{(k)}} - ||\boldsymbol{\theta}^{(k+1)} - \hat{\boldsymbol{\theta}}||^2_{\mathbf{G}^{(k)}} \right) \\
&\quad + \frac{\alpha}{2} || \left(\mathbf{G}^{(k)}\right)^{-1} \nabla_{\boldsymbol{\theta}} f(\mathbf{x}^{(k)}, y^{(k)}; \boldsymbol{\theta}^{(k)}) ||^2_{\mathbf{G}^{(k)}}
\end{aligned}$$

We sum up from $k = 0$, to the current step K, and taking its average, and since

$$f\left(\frac{1}{K+1} \sum_{k=0}^{K} \mathbf{x}^{(k)}, \frac{1}{K+1} \sum_{k=0}^{K} y^{(k)}; \frac{1}{K+1} \sum_{k=0}^{K} \boldsymbol{\theta}^{(k)} \right) \leq \frac{1}{K+1} \sum_{k=0}^{K} f(\mathbf{x}^{(k)}, y^{(k)}; \boldsymbol{\theta}^{(k)}),$$

and supposedly fixed α, i.e., $\alpha^{(k)} = \bar{\alpha} \ \forall \ k \in [0, K]$, then

$$\begin{aligned}
& f\left(\frac{1}{K+1} \sum_{k=0}^{K} \mathbf{x}^{(k)}, \frac{1}{K+1} \sum_{k=0}^{K} y^{(k)}; \frac{1}{K+1} \sum_{k=0}^{K} \boldsymbol{\theta}^{(k)} \right) \\
&\leq \frac{1}{t} \sum_{k=0}^{K} \left[\frac{1}{2\alpha^{(k)}} \left(\|\boldsymbol{\theta}^{(k)} - \hat{\boldsymbol{\theta}}\|^2_{\mathbf{G}^{(k)}} - \|\boldsymbol{\theta}^{(k+1)} - \hat{\boldsymbol{\theta}}\|^2_{\mathbf{G}^{(k)}} \right) \right. \\
&\qquad \left. + \frac{\alpha^{(k)}}{2} \left\| \left(\mathbf{G}^{(k)}\right)^{-1} \nabla_{\boldsymbol{\theta}} f(\mathbf{x}^{(k)}, y^{(k)}; \boldsymbol{\theta}^{(k)}) \right\|^2_{\mathbf{G}^{(k)}} \right] \\
&= \frac{1}{2\bar{\alpha}t} \sum_{k=0}^{K} \left(\|\boldsymbol{\theta}^{(k)} - \hat{\boldsymbol{\theta}}\|^2_{\mathbf{G}^{(k)}} - \|\boldsymbol{\theta}^{(k+1)} - \hat{\boldsymbol{\theta}}\|^2_{\mathbf{G}^{(k)}} \right) \\
&\quad + \frac{\alpha}{2K} \sum_{k=0}^{K} \left\| \left(\mathbf{G}^{(k)}\right)^{-1} \nabla_{\boldsymbol{\theta}} f(\mathbf{x}^{(k)}, y^{(k)}; \boldsymbol{\theta}^{(k)}) \right\|^2_{\mathbf{G}^{(k)}}.
\end{aligned}$$

[1] $||\mathbf{v}||^2_{\mathbf{A}} = \mathbf{v}^{\top} \mathbf{A} \mathbf{v} = ||\mathbf{A}^{1/2} \mathbf{v}||^2_2$

By Elad Hazan's lemma [2], i.e.,

$$\sum_{i=1}^{b} || \left(\mathbf{G}^{(i)}\right)^{-1} \nabla_{\boldsymbol{\theta}} f(\mathbf{x}^{(i)}, y^{(i)}; \boldsymbol{\theta}^{(k)})||^2_{\mathbf{G}^{(k)}} \leq 2\mathrm{Tr}(\boldsymbol{G}^{(b)}),$$

with $\mathbf{G}^{(b)}$ is defined as (2.2), and terms in the summation

$$\sum_{k=0}^{K} \left(||\boldsymbol{\theta}^{(k)} - \hat{\boldsymbol{\theta}}||^2_{\mathbf{G}^{(k)}} - ||\boldsymbol{\theta}^{(k+1)} - \hat{\boldsymbol{\theta}}||^2_{\mathbf{G}^{(k)}}\right)$$

cancel out each other, i.e.,

$$\sum_{k=0}^{K} \left(||\boldsymbol{\theta}^{(k)} - \hat{\boldsymbol{\theta}}||^2_{\mathbf{G}^{(k)}} - ||\boldsymbol{\theta}^{(k+1)} - \hat{\boldsymbol{\theta}}||^2_{\mathbf{G}^{(k)}}\right) = ||\boldsymbol{\theta}^{(0)} - \hat{\boldsymbol{\theta}}||^2_{\mathbf{G}^{(0)}} - ||\boldsymbol{\theta}^{(K+1)} - \hat{\boldsymbol{\theta}}||^2_{\mathbf{G}^{(\mathbf{K})}}$$

Therefore, we have

$$\begin{aligned} & f\left(\frac{1}{K+1}\sum_{k=0}^{K}\mathbf{x}^{(k)}, \frac{1}{K+1}\sum_{k=0}^{K} y^{(k)}; \frac{1}{K+1}\sum_{k=0}^{K}\boldsymbol{\theta}^{(k)}\right) \\ & \leq ||\boldsymbol{\theta}^{(0)} - \hat{\boldsymbol{\theta}}||^2_{\mathbf{G}^{(0)}} - ||\boldsymbol{\theta}^{(K+1)} - \hat{\boldsymbol{\theta}}||^2_{\mathbf{G}^{(\mathbf{K})}} + 2\mathrm{Tr}(\boldsymbol{G}^{(b)}) \end{aligned}$$

2.3. AdaDelta. Idea: Accumulate over a restricted time window.

Let

$$\mathbb{E}_t\{\mathbf{g}^\top \mathbf{g}\}_t = p\mathbb{E}_{t-1}\{\mathbf{g}^\top \mathbf{g}\} + (1-p)\mathbf{g}_t^\top \mathbf{g}_t,$$

then the updated parameters is followed as

$$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} - \alpha^{(k)} \left(\mathbb{E}_t\{\mathbf{g}^\top \mathbf{g}\} + \epsilon\mathbf{I}\right)^{-1/2} \mathbf{g}^{(k)},$$

with $\mathbf{g}^{(k)} \triangleq \nabla_{\boldsymbol{\theta}} f(\mathbf{x}^{(k)}, y^{(k)}; \boldsymbol{\theta}^{(k)})$.

Another idea is picking $\alpha^{(t)}$ to imitate Newton method, i.e., from the Newton method, we have

$$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} - \alpha^{(k)} \left(\mathbf{H}^{(k)}\right)^{-1} \mathbf{g}^{(k)}$$

, where $\mathbf{H}^{(k)} \triangleq \nabla_{\boldsymbol{\theta}} g^{(k)}$. Notice that the vector $\left(\mathbf{H}^{(k)}\right)^{-1} \mathbf{g}^{(k)}$ can be estimated as $\frac{\Delta\boldsymbol{\theta}^{(k)}}{\Delta\mathbf{g}^{(k)}}$, so $\alpha^{(k)}$ can be suggested as $||\Delta\boldsymbol{\theta}^{(k)}||$. From this idea, we pick

$$\alpha^{(k)} = \sqrt{\mathbb{E}_{k-1}\{||\Delta\boldsymbol{\theta}||^2\}},$$

where $\mathbb{E}_k\{||\Delta\boldsymbol{\theta}||^2\} = p\mathbb{E}_{k-1}\{||\Delta\boldsymbol{\theta}||^2\} + (1-p)\mathbb{E}\{||\Delta\boldsymbol{\theta}^{(k)}||^2\}$. Putting together, we have the updated formulaas follows

$$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(\boldsymbol{k})} - \sqrt{\mathbb{E}_{k-1}\{||\Delta\boldsymbol{\theta}||^2\}} \left(\mathbb{E}_t\{\mathbf{g}^\top \mathbf{g} + \epsilon\mathbf{I}\}\right)^{-1/2} \mathbf{g}^{(k)}$$

2.4. Root Mean Square Propagation. RMSProp [3] was provided by Geoffrey Hinton, where at every iteration, the new parameter is updated as:

$$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} - \gamma \left(\mathbb{E}_t\{\mathbf{g}^\top \mathbf{g}\} + \epsilon\mathbf{I}\right)^{-1} \mathbf{g}^{(k)}$$

3. Accelerated descent methods.

3.1. Issue with (Stochastic) Gradient Descent. Get stuck at the local minimum.

3.2. Gradient descent with Momentum. It is also called the Polyak Heavy Ball method.

$$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} - \alpha^{(k)}\mathbf{g}^{(k)} + \mu^{(k)}\left(\boldsymbol{\theta}^{(k)} - \boldsymbol{\theta}^{(k-1)}\right)$$

Issue. Let $f(x, y; \theta)$ is defined by its gradient as

$$\nabla f_\theta (x, y, \theta) = \begin{cases} 25\theta & \text{if } \theta < 1, \\ \theta + 24 & \text{if } 1 \leq \theta < 2, \\ 25\theta - 24 & \text{if } \theta \geq 2. \end{cases}$$

Then the below graph displays the solution of the Polyak Heavy Ball method over 50 iterations, with $\alpha = \frac{1}{9}, \mu = \frac{4}{9}$, and initial $x_0 = 3.3$.

3.3. Nestorov momentum.

$$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} - \mu\left(\boldsymbol{\theta}^{(k)} - \boldsymbol{\theta}^{(k-1)}\right)$$
$$-\gamma\nabla_{\boldsymbol{\theta}} f\left(\boldsymbol{\theta}^{(k)} + \mu\left(\boldsymbol{\theta}^{(k)} - \boldsymbol{\theta}^{(k-1)}\right)\right)$$

3.4. Optimality of Nestorov momentum. Theorem 2.1.7 of [5] said that for any $\boldsymbol{\theta}_0 \in \mathbb{R}^D$ and $k \leq \frac{d-1}{2}$, there exists an L-smooth convex function $f(\cdot)$ such that for any first-order algorithm the output sequence satisfies

$$f(\boldsymbol{\theta}^{(k)}) - f(\hat{\boldsymbol{\theta}}) \geq \frac{3L||\boldsymbol{\theta}_0 - \hat{\boldsymbol{\theta}}||_2^2}{32(k+1)^2}.$$

The goal is now to have an algorithm with

$$|f(\boldsymbol{\theta}^{(k)}) - f(\hat{\boldsymbol{\theta}})| = \mathcal{O}\left(\frac{L}{k^2}\right)$$
$$\Leftrightarrow |f(\boldsymbol{\theta}^{(k)}) - f(\hat{\boldsymbol{\theta}})| \leq \epsilon \quad \text{for } k \geq \sqrt{\frac{L}{\epsilon}}.$$

Nestorov proved if $f(\boldsymbol{\theta}^{(k)}) \leq f(\boldsymbol{\zeta}^{(k)}) - \frac{L}{2}||\nabla f(\boldsymbol{\zeta}^{(k)})||_2^2$, we have the optimal convergence rate $\mathcal{O}\left(\frac{L}{k^2}\right)$

3.5. Accelerated Gradient Descent. The update formula of accelerated Gradient Descent is given as

$$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\zeta}^{(k)} - \epsilon\nabla f(\boldsymbol{\zeta}^{(k)}),$$

where $\boldsymbol{\zeta}^{(k)} = \boldsymbol{\theta}^{(k)} + \frac{k+1}{k+2}(\boldsymbol{\theta}^{(k)} - \boldsymbol{\theta}^{(k-1)})$. The illustration of updating process is demonstrated in Figure 1 (based on the blog of Wibisono[2].

When $||\nabla^2 f|| \leq \frac{1}{\epsilon}$, the optimal rate of accelerated gradient descent is $\mathcal{O}\left(\frac{1}{\epsilon k^2}\right)$ while gradient descent only achieve the optimal rate of $\mathcal{O}\left(\frac{1}{\epsilon k}\right)$.

[2] https://awibisono.github.io/2016/06/20/accelerated-gradient-descent.html

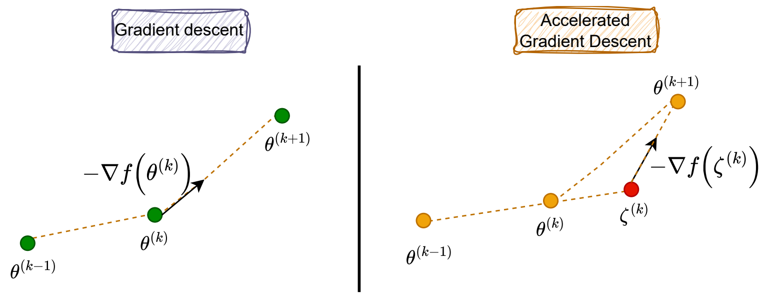

FIG. 1. *Illustration of parameter update of Gradient Descent versus Accelerated GD.*

Proof of convergence rate. Consider a Lyapunov function[3] $f : \mathbb{R}^n \to \mathbb{R}$. Let $\alpha \triangleq \epsilon, \beta \triangleq \frac{k+1}{k+2}$, from accelerated GD method we have

$$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\zeta}^{(k)} - \alpha \nabla f(\boldsymbol{\zeta}^{(k)})$$
$$\boldsymbol{\theta}^{(k)} = \boldsymbol{\zeta}^{(k-1)} - \alpha \nabla f(\boldsymbol{\zeta}^{(k-1)})$$

Subtracting both sides of the two equalities above for the optimal point $\hat{\boldsymbol{\theta}}$, we have

$$\begin{bmatrix} \boldsymbol{\theta}^{(k+1)} - \hat{\boldsymbol{\theta}} \\ \boldsymbol{\theta}^{(k)} - \hat{\boldsymbol{\theta}} \end{bmatrix} = \mathbf{T} \begin{bmatrix} \boldsymbol{\theta}^{(k)} - \hat{\boldsymbol{\theta}} \\ \boldsymbol{\theta}^{(k-1)} - \hat{\boldsymbol{\theta}} \end{bmatrix}.$$

Lemma: $p(\mathbf{T}) \coloneqq \max_{\mathbf{T}\boldsymbol{\theta}=\lambda\boldsymbol{\theta}} |\lambda|$.

Then $p(\mathbf{T}) < p \Leftrightarrow \exists$ a positive semidefinite matrix $\mathbf{K}$ such that

$$p^2\mathbf{K} - \mathbf{T}^\top \mathbf{K}\mathbf{T} \geq 0$$

For $\mathbf{T}$ and corresponding $\mathbf{K}$, define

$$V(\boldsymbol{\theta}) = \boldsymbol{\theta}^\top \mathbf{Q} \boldsymbol{\theta}.$$

We analyze the decrease of $V(\boldsymbol{\theta})$ on $\{\boldsymbol{\theta}^{(t)}\}_{t=0}^k$. By using the lemma above, we can show that

$$||\boldsymbol{\theta}^{(k)} - \hat{\boldsymbol{\theta}}|| \leq \sqrt{2\text{cond}(\mathbf{Q})}p(\mathbf{T})^k||\boldsymbol{\theta}^{(0)} - \hat{\boldsymbol{\theta}}||^k.$$

The optimal values of α, β are given as

$$\alpha = \left(\frac{2}{\sqrt{\sigma_1} + \sqrt{\sigma_n}}\right)^2, \quad \beta = \left(\frac{\sqrt{\sigma_1} - \sqrt{\sigma_n}}{\sqrt{\sigma_1} + \sqrt{\sigma_n}}\right)^2,$$

where σ_1, σ_n is the largest and smallest singular values. In practice, these values are chosen as fixed $\beta = 0.99$ and tune value of α.

4. Adaptive Moment Estimation (ADAM).

The Adaptive Moment Estimation (ADAM) [4] combines all advantages of the momentum method, and the AdaGrad, RMSProp.

5. Exercises.

[3] https://en.wikipedia.org/wiki/Lyapunov_function

Algorithm 4.1 Adam: Adaptive Moment Estimation

Require: α: Stepsize
Require: $\beta_1, \beta_2 \in [0,1)$: Exponential decay rates for moment estimates
Require: $f(\boldsymbol{\theta})$: Stochastic objective function with parameters $\boldsymbol{\theta}$
Require: Initial parameters $\boldsymbol{\theta}_0$

```
m_0 ← 0 {Initialize 1st moment vector}
v_0 ← 0 {Initialize 2nd moment vector}
t ← 0 {Initialize timestep}
while θ_t not converged do
    t ← t + 1
    g_t ← ∇_θ f_t(θ_{t-1}) {Compute gradients}
    m_t ← β_1 m_{t-1} + (1 − β_1) g_t {Update biased 1st moment}
    v_t ← β_2 v_{t-1} + (1 − β_2) g_t^2 {Update biased 2nd moment}
    m̂_t ← m_t / (1 − β_1^t) {Bias-corrected 1st moment}
    v̂_t ← v_t / (1 − β_2^t) {Bias-corrected 2nd moment}
    θ_t ← θ_{t-1} − α · m̂_t / (√v̂_t + ε) {Update parameters}
end while
return θ_t {Resulting parameters}
```

Exercise 1. Suppose we have n data points with each having 100 features and unit norm: $\mathbf{x}_i \in \mathbb{R}^{100}, ||x_i||_2 = 1, \forall\, i \in [n]$. We want to classify these n data points into 5 categories. We'll use logistic regression for this purpose. Analyze the number of expected steps for stochastic gradient descent to make the loss go below 0.01.

Solution:

Let the softmax logistic regression loss be defined as:

$$L(w) = \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, w^T x_i),$$

where ℓ is the cross-entropy loss between the predicted probabilities and the true labels $y_i \in \{1,2,3,4,5\}$. Let assume: The gradient $\nabla L(w)$ is L-Lipschitz continuous. We use a constant learning rate η. The stochastic gradients have bounded variance: $\mathbb{E}\left[\|\nabla L(w) - \nabla \tilde{L}(w)\|^2\right] \leq \sigma^2$. Desired squared gradient norm $\mathbb{E}[\|\nabla L(w)\|^2] \leq \varepsilon = 0.01$.

From the convergence theory of SGD for non-convex objectives, the number of iterations T required to achieve this level of accuracy is:

$$T = \mathcal{O}\left(\frac{1}{\varepsilon^2}\right)$$

Substituting the target accuracy:

$$T = \mathcal{O}\left(\frac{1}{0.01^2}\right) = \mathcal{O}(10^4)$$

Therefore, SGD is expected to require approximately 10,000 iterations to reduce the loss gradient norm below 0.01.

Exercise 2. Suppose we have n data points $\mathbf{x}_i \in \mathbb{R}^{100}, ||\mathbf{x}_i|| = 1, \forall\, i \in [n]$ and an objective function $h(\cdot)$ is written as

$$h(\mathbf{x}) = \sum_{i=1}^{n} f(\mathbf{x}_i).$$

That is $f(\cdot)$, a specific function is applied to the data point $\mathbf{x}_i$ separately, one by one. We assume Gaussian noise in our computation, that is

$$\left(\nabla f(\mathbf{x}_i)\right)_j = \left(\nabla f(\mathbf{x}_i)\right)_j + (\boldsymbol{\epsilon}_i)_j, \quad \forall\ i \in [n], j \in [100],$$

where $(\boldsymbol{\epsilon}_j)_j \sim \mathcal{N}(0, \sigma^2)$. We sample m data points from our database uniformly randomly without replacement, suppose this set of data points is called $\mathcal{S}$. Let

$$g(\mathbf{x}) = \frac{1}{m}\sum_{i \in \mathcal{S}} \nabla f(\mathbf{x}_i).$$

Please derive a bound on the error term

$$||\nabla f(\mathbf{x})|| - \mathbb{E}_{\mathcal{S}}\{g(\mathbf{x})\}.$$

Solution: By linearity of expectation, we have:

$$\mathbb{E}_S[g(x)] = \frac{1}{m}\sum_{i \in S} \mathbb{E}[\nabla f(x_i)] = \nabla f(x)$$

So the expected bias is zero, and we compute the variance of the estimator:

$$\mathbb{E}_S\left[\|\nabla f(x) - g(x)\|^2\right] = \mathbb{E}_S\left[\left\|\frac{1}{m}\sum_{i \in S} \xi_i\right\|^2\right]$$

Each noise vector $\xi_i \in \mathbb{R}^{100}$ has components $\xi_{ij} \sim \mathcal{N}(0, \sigma^2)$, so:

$$\mathbb{E}\left[\|\xi_i\|^2\right] = \sum_{j=1}^{100} \mathbb{E}[\xi_{ij}^2] = 100 \cdot \sigma^2$$

Since we are averaging m i.i.d. noise vectors:

$$\mathrm{Var}(g(x)) = \frac{1}{m^2} \cdot m \cdot \mathbb{E}[\|\xi_i\|^2] = \frac{100\sigma^2}{m}$$

Taking the square root to get the standard deviation (expected deviation):

$$\mathbb{E}\left[\|\nabla f(x) - g(x)\|\right] \leq \sqrt{\frac{100 \cdot \sigma^2}{m}} = \frac{10\sigma}{\sqrt{m}}$$

Thus, the expected error norm is bounded by:

$$\mathbb{E}_S\left[\|\nabla f(x) - g(x)\|\right] \leq \frac{10\sigma}{\sqrt{m}}.$$

REFERENCES

[1] J. Duchi, E. Hazan, and Y. Singer, Adaptive subgradient methods for online learning and stochastic optimization., Journal of machine learning research, 12 (2011).

[2] E. Hazan et al., Introduction to online convex optimization, Foundations and Trends® in Optimization, 2 (2016), pp. 157–325.

[3] G. Hinton, N. Srivastava, and K. Swersky, Neural networks for machine learning lecture 6a overview of mini-batch gradient descent, Cited on, 14 (2012), p. 2.

[4] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980, (2014).

[5] Y. Nesterov et al., Lectures on convex optimization, vol. 137, Springer, 2018.

[6] T. Strohmer and R. Vershynin, A randomized kaczmarz algorithm with exponential convergence, Journal of Fourier Analysis and Applications, 15 (2009), pp. 262–278.