

Modern Stochastic Methods

Yu-Xiang Wang
CS292F

(Based on Ryan Tibshirani's 10-725)

Last time: Exponential Concavity and Online Newton Step

A weaker assumption under which you can get $\log T$ regret.

- An algorithm with online “quasi-newton” update.
- Exponentially Concave functions and its “directional / restricted” strongly convex interpretation.
- Sherman-Morrison-Woodbury Identity for a fast $O(n^2)$ implementation for the Online Newton Step.

Other topics in online learning not covered:

- Bandits problems — reduction to full-info OGD by doing a stochastic estimate of the gradient.
- Projection-free algorithms — from $O(n^3)$ to $O(nnz)$ for matrix problems.
- Adaptive online learning — adaptive regret, dynamic regret and other variants.

One of these topics in the final lecture.

Recall: Stochastic gradient descent

Consider minimizing an average of functions

$$\min_x \frac{1}{n} \sum_{i=1}^n f_i(x)$$

As $\nabla \sum_{i=1}^n f_i(x) = \sum_{i=1}^n \nabla f_i(x)$, gradient descent or GD repeats:

$$x^{(k)} = x^{(k-1)} - t_k \cdot \frac{1}{n} \sum_{i=1}^n \nabla f_i(x^{(k-1)}), \quad k = 1, 2, 3, \dots$$

In comparison, **stochastic gradient descent** or SGD repeats:

$$x^{(k)} = x^{(k-1)} - t_k \cdot \nabla f_{i_k}(x^{(k-1)}), \quad k = 1, 2, 3, \dots$$

where $i_k \in \{1, \dots, n\}$ is randomly chosen index at iteration k . Note $\mathbb{E}[\nabla f_{i_k}(x)] = \nabla f(x)$, so we use **unbiased estimate** of full gradient

Recall: Convergence rates

Recall the following:

Condition	GD rate	SGD rate
Convex	$O(1/\sqrt{k})$	$O(1/\sqrt{k})$
+ Lipschitz gradient	$O(1/k)$	$O(1/\sqrt{k})$
+ Strongly convex	$O(c^k)$	$O(1/k)$

Notes:

- In GD, we can take fixed step sizes in the latter two cases
- In SGD, we always take diminishing step sizes to control the variance (of the gradient estimate)

Recall: End of the story?

Is this the end of the story? SGD simply **cannot adapt** to strong convexity, and this is the best we can hope for?

For a while, the answer was believed to be yes, as Nemirovski and others established matching **lower bounds** ... however this was for general stochastic problem, where

$$f(x) = \int F(x, \xi) dP(\xi)$$

For finite sums (our focus)

$$f(x) = \frac{1}{n} \sum_{i=1}^n f_i(x)$$

new wave of **variance reduction** work shows we can modify SGD to converge much faster, basically retaining the properties of GD

Outline

Rest of today:

- Fast Gradient methods: (SAG, SAGA, SDCA, SVRG)
- Acceleration and momentum
- Adaptive step sizes (AdaGrad, ADAM, SignSGD)

Stochastic average gradient

Stochastic average gradient or SAG (Schmidt, Le Roux, and Bach 2013) is a breakthrough method in stochastic optimization:

- Maintain table, containing gradient g_i of f_i , $i = 1, \dots, n$
- Initialize $x^{(0)}$, and $g_i^{(0)} = \nabla f_i(x^{(0)})$, $i = 1, \dots, n$
- At steps $k = 1, 2, 3, \dots$, pick random $i_k \in \{1, \dots, n\}$, then let

$$g_{i_k}^{(k)} = \nabla f_{i_k}(x^{(k-1)}) \quad (\text{most recent gradient of } f_{i_k})$$

Set all other $g_i^{(k)} = g_i^{(k-1)}$, $i \neq i_k$, i.e., these stay the same

- Update

$$x^{(k)} = x^{(k-1)} - t_k \cdot \frac{1}{n} \sum_{i=1}^n g_i^{(k)}$$

Notes:

- Key of SAG is to allow each f_i , $i = 1, \dots, n$ to communicate a part of the gradient estimate at each step
- This basic idea can be traced back to incremental aggregated gradient (Blatt, Hero, Gauchman, 2006)
- SAG gradient estimates are **no longer unbiased**, but they have **greatly reduced variance**
- Isn't it expensive to average all these gradients? Basically **just as efficient** as SGD, as long we're clever:

$$x^{(k)} = x^{(k-1)} - t_k \cdot \underbrace{\left(\frac{g_{i_k}^{(k)}}{n} - \frac{g_{i_k}^{(k-1)}}{n} + \underbrace{\frac{1}{n} \sum_{i=1}^n g_i^{(k-1)}}_{\text{old table average}} \right)}_{\text{new table average}}$$

SAG convergence analysis

Assume that $f(x) = \frac{1}{n} \sum_{i=1}^n f_i(x)$, where each f_i is differentiable, and ∇f_i is Lipschitz with constant L

Denote $\bar{x}^{(k)} = \frac{1}{k} \sum_{\ell=0}^{k-1} x^{(\ell)}$, the average iterate after $k - 1$ steps

Theorem (Schmidt, Le Roux, Bach): SAG, with a fixed step size $t = 1/(16L)$, and the initialization

$$g_i^{(0)} = \nabla f_i(x^{(0)}) - \nabla f(x^{(0)}), \quad i = 1, \dots, n$$

satisfies

$$\mathbb{E}[f(\bar{x}^{(k)})] - f^* \leq \frac{48n}{k} (f(x^{(0)}) - f^*) + \frac{128L}{k} \|x^{(0)} - x^*\|_2^2$$

where the expectation is taken over random choices of indices

Notes:

- Result stated in terms of the average iterate $\bar{x}^{(k)}$, but also can be shown to hold for best iterate $x_{\text{best}}^{(k)}$ seen so far
- This is $O(1/k)$ convergence rate for SAG. Compare to $O(1/k)$ rate for GD, and $O(1/\sqrt{k})$ rate for SGD
- But, the **constants are different!** Bounds after k steps:

$$\text{GD : } \frac{L}{2k} \|x^{(0)} - x^*\|_2^2$$

$$\text{SAG : } \frac{48n}{k} (f(x^{(0)}) - f^*) + \frac{128L}{k} \|x^{(0)} - x^*\|_2^2$$

- So first term in SAG bound suffers from factor of n ; authors suggest smarter initialization to make $f(x^{(0)}) - f^*$ small (e.g., they suggest using result of n SGD steps)

Convergence under strong convexity

Assume further that each f_i is strongly convex with parameter m

Theorem (Schmidt, Le Roux, Bach): SAG, with a step size $t = 1/(16L)$ and the same initialization as before, satisfies

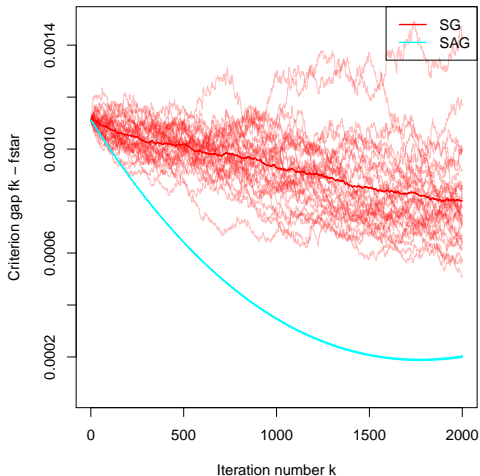
$$\mathbb{E}[f(x^{(k)})] - f^* \leq \left(1 - \min\left\{\frac{m}{16L}, \frac{1}{8n}\right\}\right)^k \times \left(\frac{3}{2}(f(x^{(0)}) - f^*) + \frac{4L}{n}\|x^{(0)} - x^*\|_2^2\right)$$

Notes:

- This is **linear** convergence rate $O(c^k)$ for SAG. Compare this to $O(c^k)$ for GD, and only $O(1/k)$ for SGD
- Like GD, we say SAG is **adaptive to strong convexity** (achieves better rate with same settings)
- Proofs of these results **not easy**: 15 pages, computed-aided!

Example: logistic regression

Back to our logistic regression, SGD versus SAG, over 30 reruns of these randomized algorithms:



Notes:

- SAG does well, but did not work out of the box; required a specific setup
- Took one full cycle of SGD (one pass over the data) to get $\beta^{(0)}$, and then started SGD and SAG both from $\beta^{(0)}$. This **warm start helped** a lot
- SAG initialized at $g_i^{(0)} = \nabla f_i(\beta^{(0)})$, $i = 1, \dots, n$, computed during initial SGD cycle. Centering these gradients was much worse (and so was initializing them at 0)
- Tuning the fixed step sizes for SAG was very finicky; here now hand-tuned to be about as large as possible before it diverges
- Authors of SAG conveyed that this algorithm will work the best, relative to SGD, for ill-conditioned problems (the current problem not being ill-conditioned at all)

SAGA

SAGA (Defazio, Bach, and Lacoste-Julien 2014) is a follow-up on the SAG work:

- Maintain table, containing gradient g_i of f_i , $i = 1, \dots, n$
- Initialize $x^{(0)}$, and $g_i^{(0)} = \nabla f_i(x^{(0)})$, $i = 1, \dots, n$
- At steps $k = 1, 2, 3, \dots$, pick random $i_k \in \{1, \dots, n\}$, then let

$$g_{i_k}^{(k)} = \nabla f_{i_k}(x^{(k-1)}) \quad (\text{most recent gradient of } f_{i_k})$$

Set all other $g_i^{(k)} = g_i^{(k-1)}$, $i \neq i_k$, i.e., these stay the same

- Update

$$x^{(k)} = x^{(k-1)} - t_k \cdot \left(g_{i_k}^{(k)} - g_{i_k}^{(k-1)} + \frac{1}{n} \sum_{i=1}^n g_i^{(k-1)} \right)$$

Notes:

- SAGA gradient estimate $g_{i_k}^{(k)} - g_{i_k}^{(k-1)} + \frac{1}{n} \sum_{i=1}^n g_i^{(k-1)}$, versus SAG gradient estimate $\frac{1}{n} g_{i_k}^{(k)} - \frac{1}{n} g_{i_k}^{(k-1)} + \frac{1}{n} \sum_{i=1}^n g_i^{(k-1)}$
- Recall, SAG estimate is biased; remarkably, SAGA estimate is **unbiased**! Simple explanation: consider family of estimators

$$\theta_\alpha = \alpha(X - Y) + \mathbb{E}(Y)$$

for $\mathbb{E}(X)$, where $\alpha \in [0, 1]$, and X, Y are presumed correlated. We have

$$\mathbb{E}(\theta_\alpha) = \alpha\mathbb{E}(X) + (1 - \alpha)\mathbb{E}(Y)$$

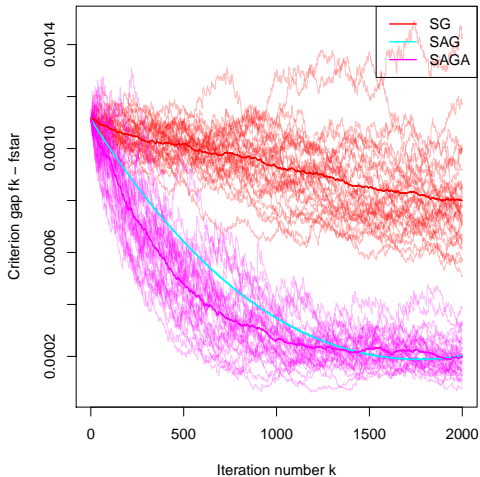
$$\text{Var}(\theta_\alpha) = \alpha^2(\text{Var}(X) + \text{Var}(Y) - 2\text{Cov}(X, Y))$$

SAGA uses $\alpha = 1$ (unbiased), SAG uses $\alpha = 1/n$ (biased)

- SAGA matches convergence rates of SAG, with simpler proofs

Example: logistic regression

Back to our logistic regression example, now adding SAGA to mix:



Notes:

- SAGA does well, but again it required somewhat specific setup
- As before, took one full cycle of SGD (one pass over the data) to get $\beta^{(0)}$, and then started SGD, SAG, SAGA all from $\beta^{(0)}$. This **warm start helped** a lot
- SAGA initialized at $g_i^{(0)} = \nabla f_i(\beta^{(0)})$, $i = 1, \dots, n$, computed during initial SGD cycle. Centering these gradients was much worse (and so was initializing them at 0)
- Tuning the fixed step sizes for SAGA was fine; seemingly on par with tuning for SGD, and more robust than tuning for SAG
- Interestingly, the SAGA criterion curves look like SGD curves (realizations being jagged and highly variable); SAG looks very different, and this really emphasizes the fact that its updates have **much lower variance**

Many, many others

A lot of recent work revisiting stochastic optimization:

- SDCA (Shalev-Schwartz, Zhang, 2013): applies coordinate ascent to the dual of ridge regularized problems, and uses randomly selected coordinates. Effective primal updates are similar to SAG/SAGA
(Primal dual convergence. Example of dual separation!)
- SVRG (Johnson, Zhang, 2013): like SAG/SAGA, but does not store a full table of gradients, just an average, and updates this occasionally
- There's also S2GD (Konecny, Richtarik, 2014), MISO (Mairal, 2013), Finito (Defazio, Caetano, Domke, 2014), etc.
- Both the SAG and SAGA papers give very nice reviews and discuss connections

SVRG: Stochastic Variance Reduced Gradient

- Initialize $\tilde{x} = x_0$.
- For $s = 1, 2, \dots$ (Outer Loop)
 1. Take a snapshot $\tilde{x} = \tilde{x}_{s-1}$
 2. Calculate the full gradient: $\tilde{g} = \frac{1}{n} \sum_{i=1}^n \nabla f_i(\tilde{x})$.
 3. Set $x_0 = \tilde{x}$.
 4. For $t = 1, 2, \dots, N$ (Inner Loop)
 - a. Randomly sample $i_t \in \{1, 2, \dots, n\}$
 - b. $x_t = x_{t-1} - \eta(\nabla f_{i_t}(x_{t-1}) - \nabla f_{i_t}(\tilde{x}) + \tilde{g})$
 5. Set $\tilde{x}_s = x_t$ with t chosen at random from $\{0, 1, \dots, N - 1\}$.

Convergence: When $N = 32L/m$ and $\eta = 1/8L$, the sequence converges at a rate $\mathbb{E}F_n(\tilde{x}_s) - F_n^* \leq (\frac{2}{3})^s (F(x_0) - F^*)$. Or if we rewrite it in terms of the number of incremental gradient oracle calls

$$O((n + L/m) \log(1/\epsilon)).$$

Ideas in the convergence proof

1. Variance reduction: $\mathbb{E}[\|\tilde{g}_t\|^2] \rightarrow 0$ as the number of iterations t gets larger.
2. Eventually \tilde{g} is as good as $\nabla F(x_t)$. Thus, we can use a constant learning rate!
3. Analyze the outer loop convergence.

Optimality and acceleration

For finite sums, Lan and Zhou (2015) (also Woodworth and Srebro 2016) prove lower bounds, that do not match to upper bounds from SAG, SAGA (and others). E.g., for strongly convex setting:

- SAG, SAGA, SDCA, SVRG (and others) have iteration complexity:

$$O\left(\left(n + \frac{L}{m}\right) \log(1/\epsilon)\right)$$

- Lower bound:

$$O\left(\left(n + \sqrt{\frac{nL}{m}}\right) \log(1/\epsilon)\right)$$

Can we do better? Yes! Use **acceleration** (Lan and Zhou 2015, Lin et al., 2015)

Momentum and beyond

Variance reduction + acceleration completely solve the finite sum case. Beyond this, the story is much more complicated ...

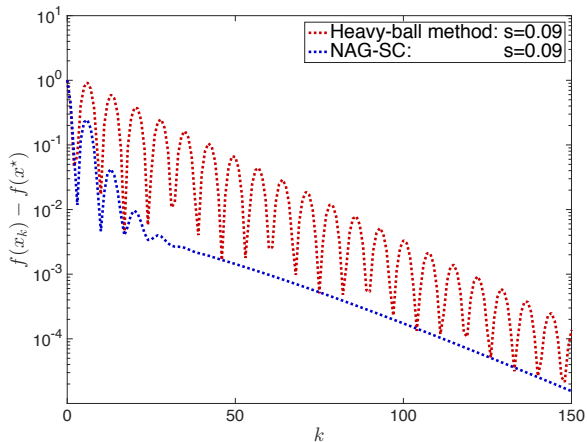
- Recall, for general stochastic setting, the performance of SGD cannot be improved (matching lower bounds in Nemirovski et al. 2009)
- Acceleration is less used for nonconvex problems (?), but a related technique is often used: **momentum**
- Predates acceleration by nearly two decades (Polyak, 1964). In practice, Polyak's heavy ball method can work really well:

$$x^{(k)} = x^{(k-1)} + \alpha(x^{(k-1)} - x^{(k-2)}) - t_k \nabla f_{i_k}(x^{(k-1)})$$

but it can also be somewhat fragile

- Open problem: when and why does this work?

Polyak's heavy ball versus Nesterov acceleration, in optimizing a convex quadratic (from Shi et al., 2018):



Adaptive step sizes

Another big topic in stochastic optimization these days: **adaptive step sizes**

To motivate, let's consider a logistic regression problem, where x_{ij} are binary, and many of them are zero. E.g., classifying whether a given movie review is positive or negative:

Piece of subtle art. Maybe a masterpiece. Doubtlessly a special story about the ambiguity of existence.

Some words are common (blue) and uninformative and some rare (green) and informative. Here:

- x_{ij} represents whether the j th word is present in i th review
- y_i represents the i th review is positive or negative (sentiment)

Recall we have $f_i(\beta) = -y_i x_i^T \beta + \log(1 + \exp(x_i^T \beta))$, and

$$\nabla f_i(\beta) = \left(-y_i + \frac{1}{1 + \exp(-x_i^T \beta)} \right) x_i$$

Observation: $x_{ij} = 0$ implies that $\nabla_j f_i(\beta) = 0$. Also $\|\nabla f_i(\beta)\|_2$ is large when i th review is misclassified

So what does SGD do?

- Gives equal weight to common and to rare informative words
- Diminishing step sizes t_k means the rare informative features are learned very slowly ...

To escape this long wait, we'll have to adapt the step sizes to pick up the informative features

AdaGrad

AdaGrad (Duchi, Hazan, and Singer 2010): very popular adaptive method. Let $g^{(k)} = \nabla f_{i_k}(x^{(k-1)})$, and update for $j = 1, \dots, p$:

$$x_j^{(k)} = x_j^{(k-1)} - \alpha \cdot \frac{g_j^{(k)}}{\sqrt{\sum_{\ell=1}^k (g_j^{(\ell)})^2}}$$

Notes:

- AdaGrad does not require tuning learning rate: $\alpha > 0$ is fixed constant, learning rate decreases naturally over iterations
- Learning rate of rare informative features diminishes slowly
- Can drastically improve over SGD in sparse problems
- More recent variations Adam, RMSProp, etc., very popular in training deep nets

Adam and SignSGD

- ADAM: (Kingma and Ba, ICLR'2015)

$$x_j^{(k)} = x_j^{(k-1)} - \alpha \cdot \frac{m_j^{(k)}}{\sqrt{v_j^{(k)} + \epsilon}}$$

where $m^{(k)} = \beta_1 m^{(k-1)} + (1 - \beta_1)g^{(k)}$,
 $v_j^{(k)} = \beta_2 v_j^{(k-1)} + (1 - \beta_2)[g^{(k)}]^2$.

- SignSGD with momentum: (Bernstein, W., Azizzadenesheli, Anandkumar, ICML'18)

$$x_j^{(k)} = x_j^{(k-1)} - \alpha \cdot \text{sign}(m_j^{(k)}).$$

32x savings in the communication! Elegant convergence bound. Related to the steepest descent method with an ℓ_∞ -norm (Section 9.4, BV book).

SignSGD in practice is very similar to ADAM

Imagenet with ResNet50. Staircasing Learning Rate.

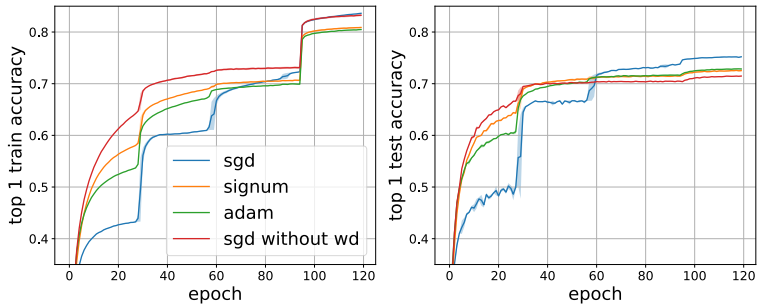


Figure extracted from (Bernstein, W., Azizzadenesheli, Anandkumar, ICML'18)

References and further reading

References for finite sum:

- M. Schmidt and N. Le Roux and F. Bach (2013), “Minimizing finite sums with the stochastic average gradient”
- S. Shalev-Shwartz and T. Zhang (2013). “Stochastic dual coordinate ascent methods for regularized loss minimization”.
- R. Johnson and T. Zhang. (2013). “Accelerating stochastic gradient descent using predictive variance reduction”.
- A. Defasio and F. Bach and S. Lacoste-Julien (2014), “SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives”

More on Allen-Zhu's excellent ICML'17 tutorial.

References and further reading

Adaptive Gradient Methods References.

- J. Duchi and E. Hazan and Y. Singer (2010), “Adaptive subgradient methods for online learning and stochastic optimization”
- D.P. Kingma and J.L. Ba (2015) “ADAM: A Method for Stochastic Optimization”
- J. Bernstein, Y.X. Wang and K. Azizzadenesheli and A. Anandkumar. (2018) “SIGNSGD: compressed optimisation for non-convex problems.”

Other pioneering work.

- G. Lan and Y. Zhou (2015), “An optimal randomized incremental gradient method”.
- A. Nemirovski and A. Juditsky and G. Lan and A. Shapiro (2009), “Robust stochastic optimization approach to stochastic programming”.