Modern Stochastic Methods

Ryan Tibshirani
(notes by Sashank Reddi and Ryan Tibshirani)
Convex Optimization 10-725

Last time: conditional gradient method

For the problem

$$\min_x \ f(x) \ \text{ subject to } \ x \in C$$

where f is convex, smooth and C is a convex set, the Frank-Wolfe method chooses an initial $x^{(0)}$ and repeats for $k = 1, 2, 3, \ldots$

$$s^{(k-1)} \in \operatorname*{argmin}_{s \in C} \ \nabla f(x^{(k-1)})^T s$$
$$x^{(k)} = (1 - \gamma_k) x^{(k-1)} + \gamma_k s^{(k-1)}$$

Here γ_k is a step size, either prespecified (as in $\gamma_k = 2/(k+1)$) or chosen by line search. Convergence is similar to projected gradient

For many problems, linear minimization over C is simpler or more efficient than projection onto C, hence the appeal of Frank-Wolfe

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:

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

In comparison, stochastic gradient descent or SGD repeats:

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

where $i_k \in \{1, \ldots n\}$ 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

Mini-batches

Also common is mini-batch stochastic gradient descent, where we choose a random subset $I_k \subseteq \{1, \ldots n\}$, of size $|I_k| = b \ll n$, and repeat:

$$x^{(k)} = x^{(k-1)} - t_k \cdot \frac{1}{b} \sum_{i \in I_k} \nabla f_i(x^{(k-1)}), \quad k = 1, 2, 3, \ldots$$

Again, we are approximating full graident by an unbiased estimate:

$$\mathbb{E}\left[\frac{1}{b} \sum_{i \in I_k} \nabla f_i(x)\right] = \nabla f(x)$$

Using mini-batches reduces the variance of our gradient estimate by a factor $1/b$, but is also b times more expensive

Example: logistic regression

Given $(x_i, y_i) \in \mathbb{R}^p \times \{0, 1\}$, $i = 1, \ldots n$, recall logistic regression:

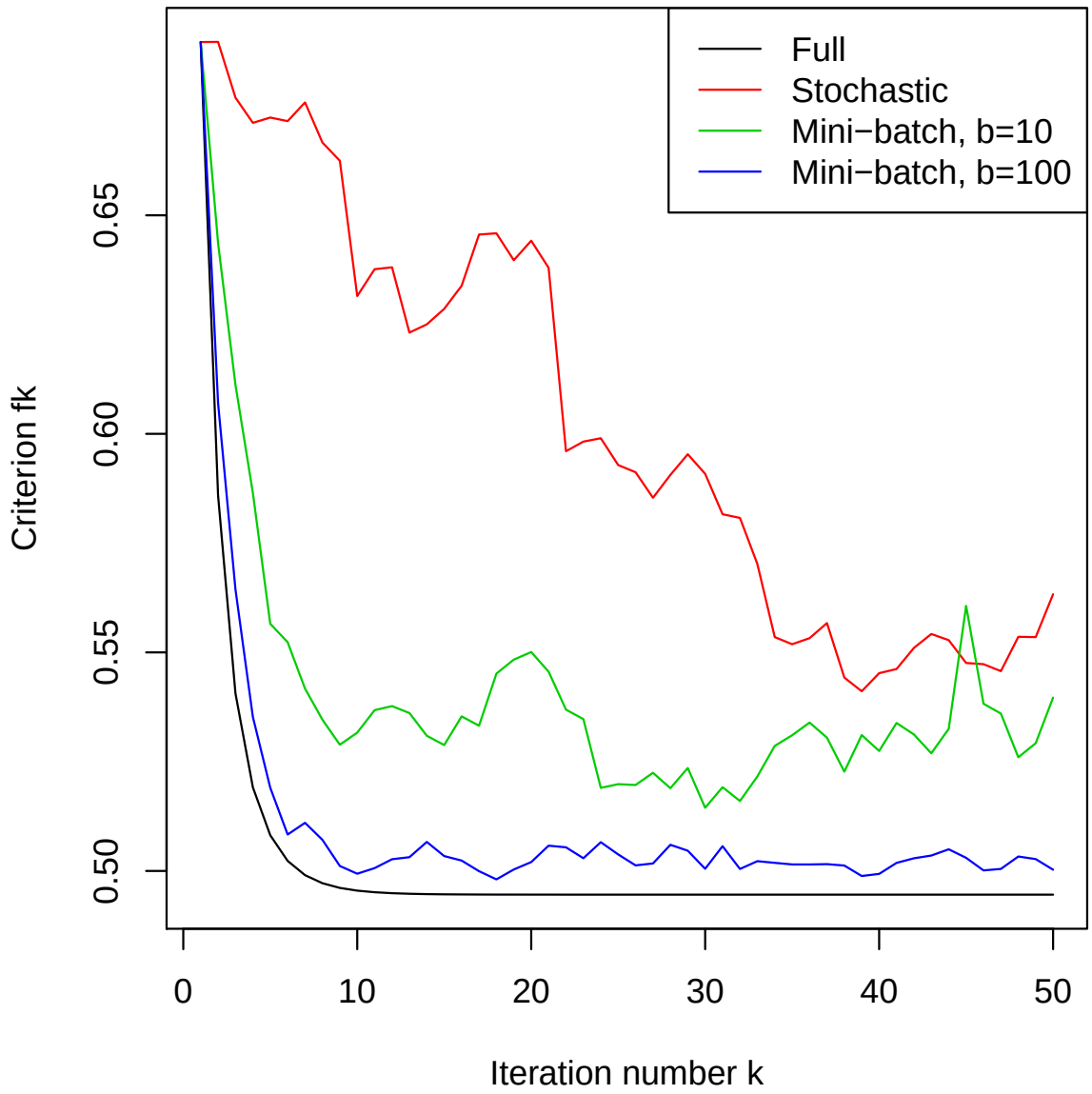
$$\min_\beta \; f(\beta) = \frac{1}{n} \sum_{i=1}^n \underbrace{\Big(- y_i x_i^T \beta + \log(1 + \exp(x_i^T \beta)) \Big)}_{f_i(\beta)}$$

Gradient computation $\nabla f(\beta) = \frac{1}{n} \sum_{i=1}^n \big(y_i - p_i(\beta)\big) x_i$ is doable when n is moderate, but not when n is huge

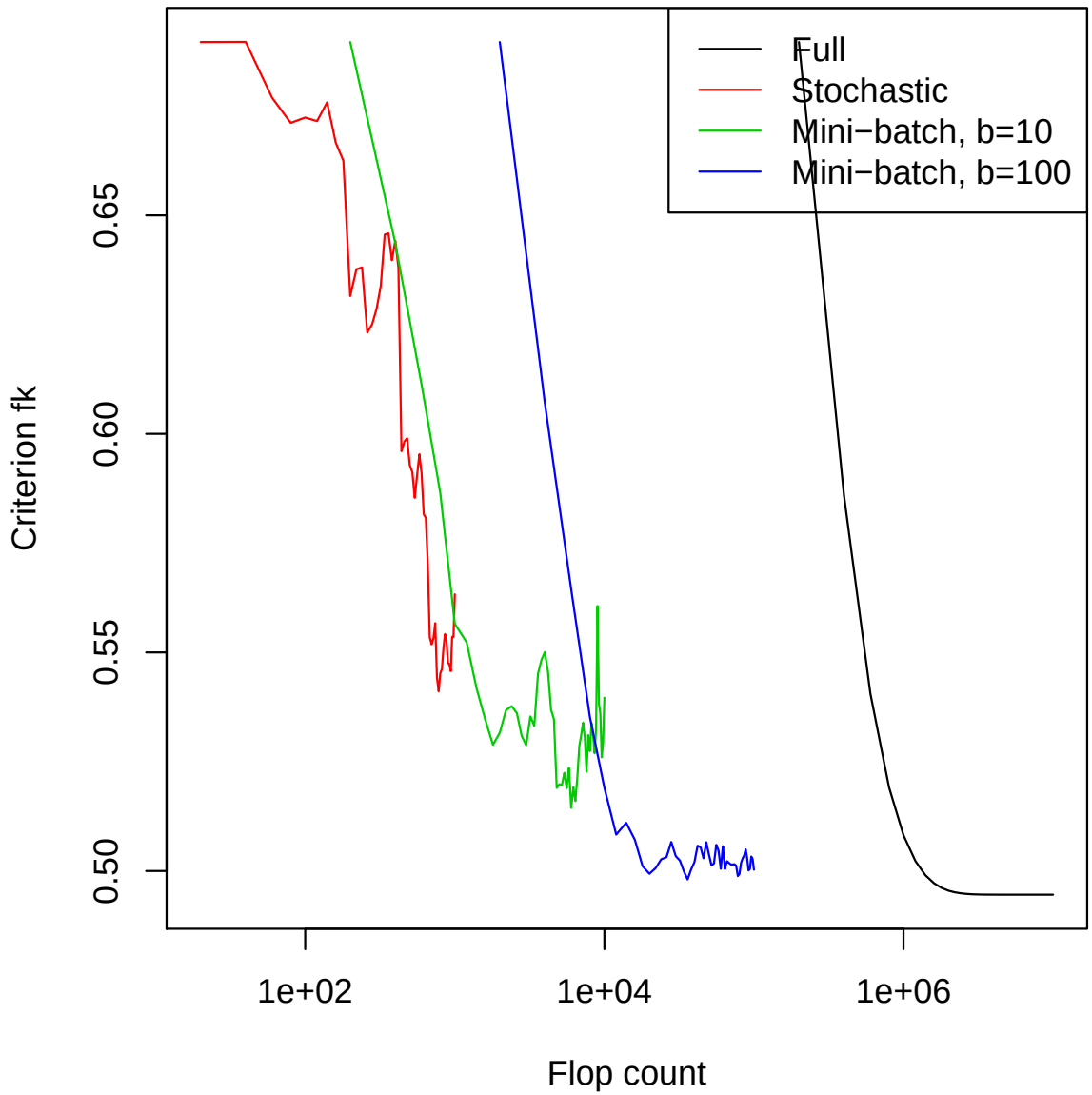
Full gradient (also called batch) versus stochastic gradient:

- One batch update costs $O(np)$
- One mini-batch update costs $O(bp)$
- One stochastic update costs $O(p)$

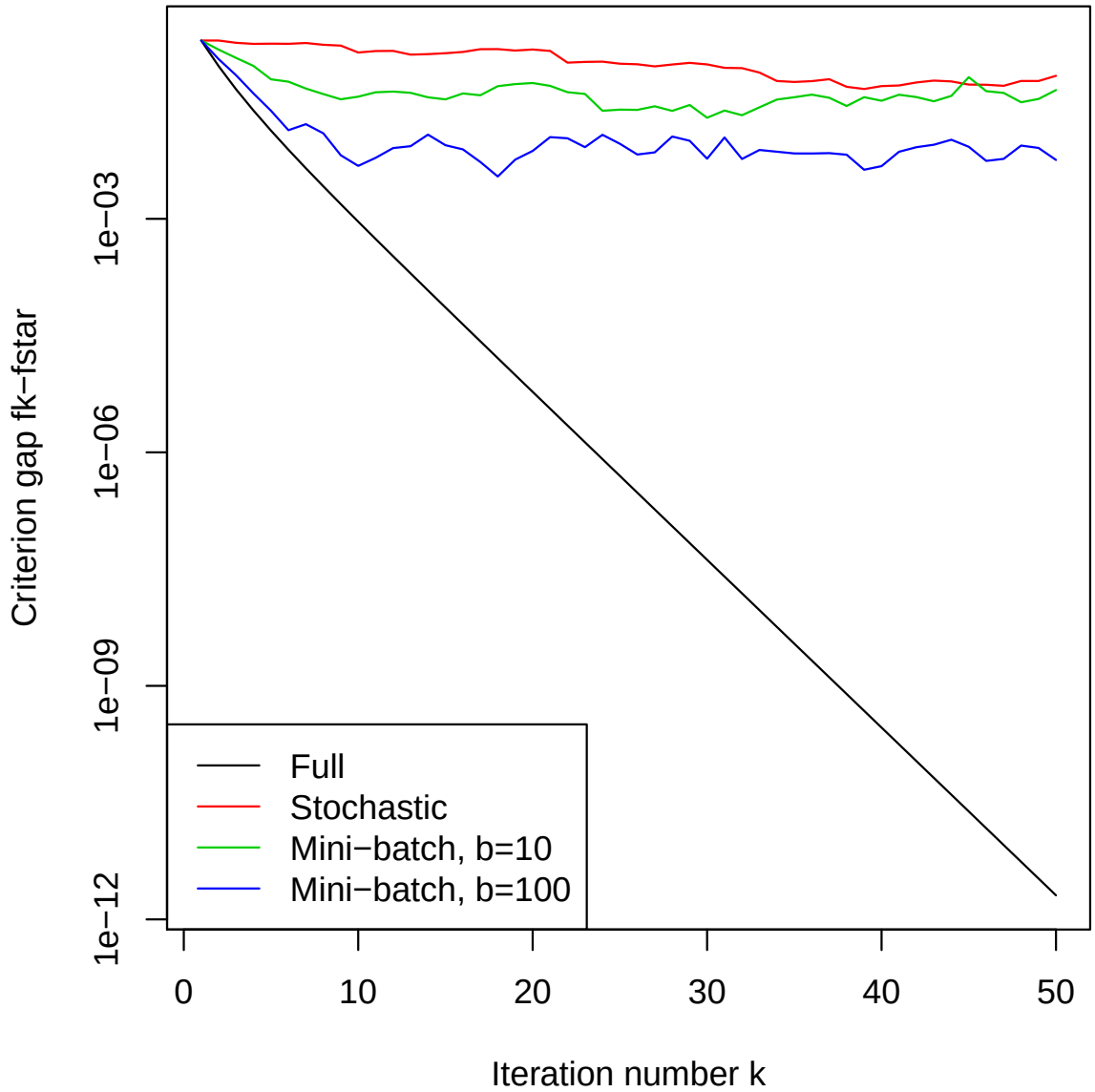
Example with $n = 10,000$, $p = 20$, all methods use fixed step sizes:



What's happening? Now let's parametrize by flops:



Finally, looking at suboptimality gap (on log scale):



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)
- Mini-batches are a wash in terms of flops (but still popular practice)

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:

- Variance reduction (SAG, SAGA)
- Acceleration and momentum
- Adaptive step sizes (AdaGrad)

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

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

  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, $i = 1, \ldots n$ 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

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

satisfies

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

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 $x^{(k)}_{\text{best}}$ 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:

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

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

- So first term in SAG bound suffers from factor of n; authors suggest smarter initialization to make $f(x^{(0)}) - f^\star$ 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

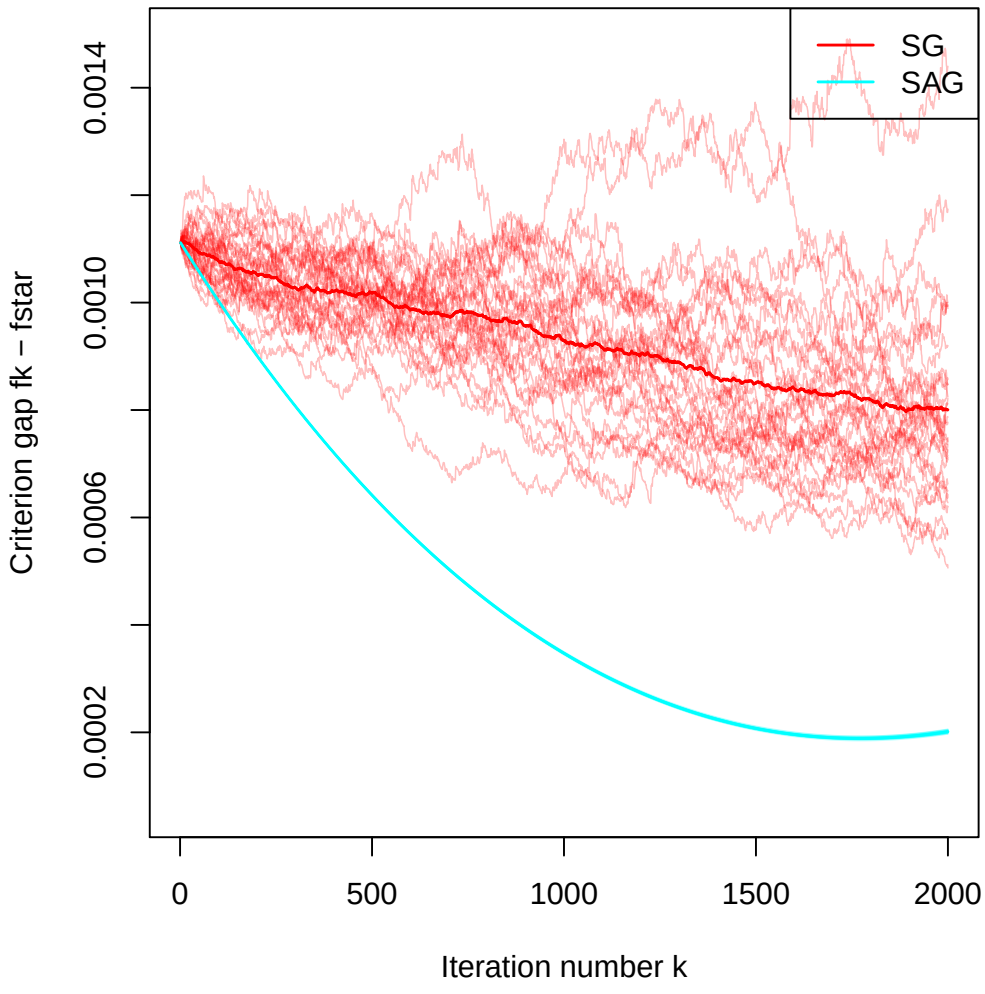
$$\mathbb{E}[f(x^{(k)})] - f^\star \leq \left(1 - \min\left\{\frac{m}{16L}, \frac{1}{8n}\right\}\right)^k \times \left(\frac{3}{2}(f(x^{(0)}) - f^\star) + \frac{4L}{n}\|x^{(0)} - x^\star\|_2^2\right)$$

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 $g_i^{(0)} = \nabla f_i(\beta^{(0)})$, $i = 1, \ldots n$, 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, $i = 1, \ldots n$
- Initialize $x^{(0)}$, and $g_i^{(0)} = \nabla f_i(x^{(0)})$, $i = 1, \ldots n$
- At steps $k = 1, 2, 3, \ldots$, pick random $i_k \in \{1, \ldots n\}$, then let

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

  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

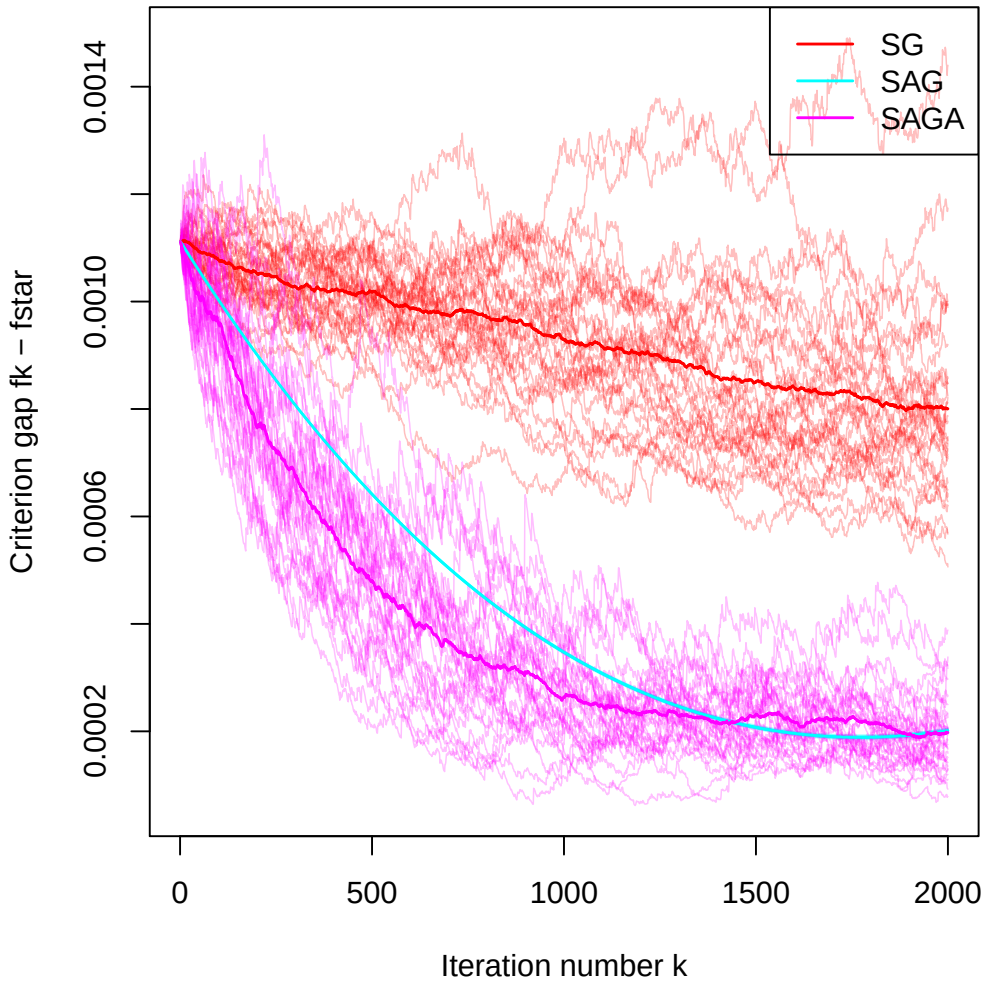
$$\begin{aligned} \mathbb{E}(\theta_\alpha) &= \alpha\mathbb{E}(X) + (1-\alpha)\mathbb{E}(Y) \\ \mathrm{Var}(\theta_\alpha) &= \alpha^2\big(\mathrm{Var}(X) + \mathrm{Var}(Y) - 2\mathrm{Cov}(X, Y)\big) \end{aligned}$$

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 $g_i^{(0)} = \nabla f_i(\beta^{(0)})$, $i = 1, \ldots n$, 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
- 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

Optimality and acceleration

For finite sums, Lan and Zhou (2015) (also Woodworth and Srebro 2016) prove lower bounds, that do no match to upper bounds from SAG, SAGA (and others). E.g., for strongly convex setting:

- SAG, SAGA (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

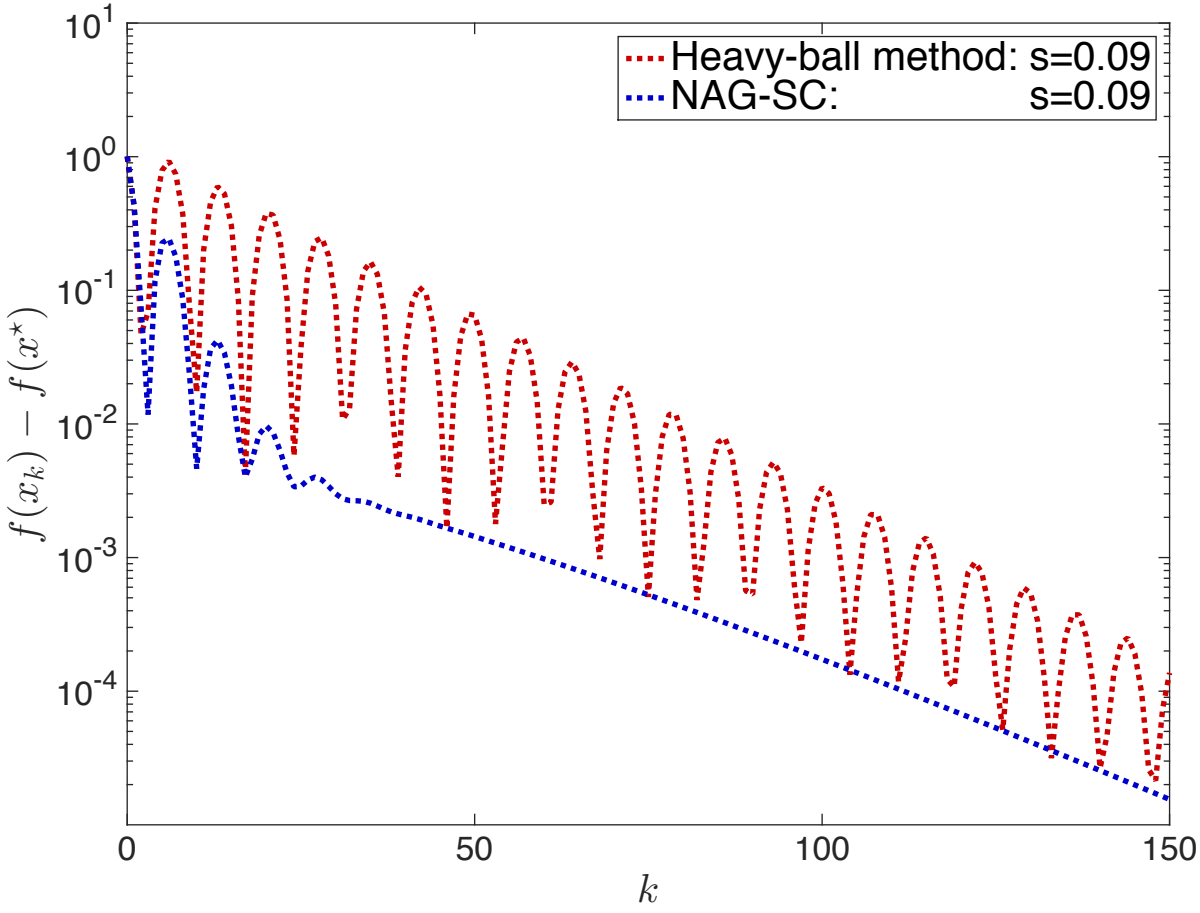
Variance reduction + acceleration completely solve the finite sum case. Beyond this, the story is much more complicated ...

- 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 jth word is present in ith review
- y_i represents the ith 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 ith 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 $j = 1, \ldots p$:

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

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

References and further reading

- J. Duchi and E. Hazan and Y. Singer (2010), "Adaptive subgradient methods for online learning and stochastic optimization"
- A. Defasio and F. Bach and S. Lacoste-Julien (2014), "SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives"
- 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"
- M. Schmidt and N. Le Roux and F. Bach (2013), "Minimizing finite sums with the stochastic average gradient"