Optimization for Data Science
ETH Zürich, FS 2023 261-5110-00L

Lecture 12: Finite Sum Optimization

Variance-reduced Stochastic Methods

Bernd Gärtner
Niao He

https://www.ti.inf.ethz.ch/ew/courses/ODS23/index.html
May 8, 2023

Lecture Outline

Recap

- Stochastic Optimization:

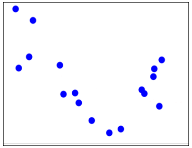
$$\min_{\mathbf{x}\in\mathbb{R}^d} \quad F(\mathbf{x}) := \mathbb{E}_{\boldsymbol{\xi}}[f(\mathbf{x},\boldsymbol{\xi})] \tag{SO}$$

- Finite Sum Optimization (special case):

this is a sepcial case of the stochastic optimization.

$$\min_{\mathbf{x}\in\mathbb{R}^d} \quad F(\mathbf{x}) := \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{x}) \tag{FS}$$

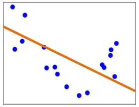
Example: Supervised Learning



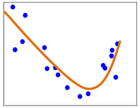
$(\mathbf{x}_i, y_i), i = 1, \dots, n$

Data

- Linear model: $h_{\mathbf{w}}(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$



- Nonlinear model: $h_{\mathbf{w}}(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x})$



- Multi-layer network model: $h_{\mathbf{w}}(\mathbf{x}) = W_3^T g_2(W_2^T g_1(W_1^T \mathbf{x}))$

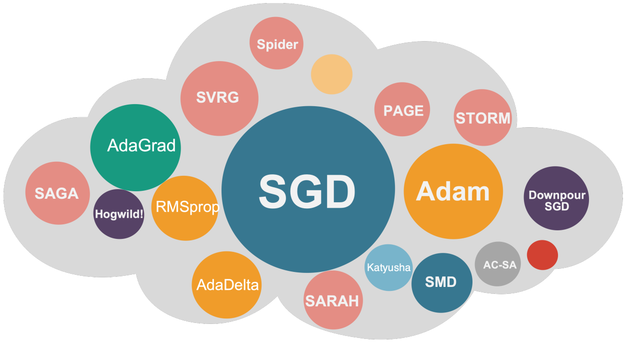
Model

$$\min_{\mathbf{w}} \mathbb{E}_{\mathbf{x},y} [\ell(h_{\mathbf{w}}(\mathbf{x}), y)]$$

$$\min_{\mathbf{w}} \frac{1}{n} \sum_{i=1}^{n} \ell(h_{\mathbf{w}}(\mathbf{x}_i), y_i)$$

Optimization

The Zoo of Stochastic Gradient Based Methods



- SGD
- Adaptive SGD
- SGD with variance reduction (This Lecture!)
- ... ...

SGD Recap

$$\min_{\mathbf{x}\in\mathbb{R}^d} \quad F(\mathbf{x}) := \mathbb{E}_{\boldsymbol{\xi}}[f(\mathbf{x},\boldsymbol{\xi})]$$

might need to put this one in the exam sheet.

SGD: $\mathbf{x}_{t+1} = \mathbf{x}_t - \gamma_t \nabla f(\mathbf{x}_t, \boldsymbol{\xi}_t)$, where $\boldsymbol{\xi}_t \overset{iid}{\sim} P(\boldsymbol{\xi})$

	Stepsize	Rate	Measure
Nonconvex	$\gamma_t = O\left(\frac{1}{\sqrt{t}}\right)$	$O\left(\frac{1}{T^{1/4}}\right)$	gradient norm
Convex	$\gamma_t = O\left(\frac{1}{\sqrt{t}}\right)$	$O\left(\frac{1}{\sqrt{T}}\right)$	function value gap
Strongly Convex	$\gamma_t = O\left(\frac{1}{\mu t}\right)$	$O\left(\frac{1}{T}\right)$	function value gap

Generic Adaptive Scheme

The following scheme encapsulates popular adaptive methods in a unified framework. [Reddi, Kale, & Kumar (2018)]

$$\begin{aligned}
\mathbf{g}_t &= \nabla f(\mathbf{x}_t, \boldsymbol{\xi}_t) \\
\mathbf{m}_t &= \phi_t(\mathbf{g}_1, \ldots, \mathbf{g}_t) \\
V_t &= \psi_t(\mathbf{g}_1, \ldots, \mathbf{g}_t) \\
\hat{\mathbf{x}}_t &= \mathbf{x}_t - \alpha_t V_t^{-1/2} \mathbf{m}_t \\
\mathbf{x}_{t+1} &= \operatorname*{argmin}_{\mathbf{x} \in X} \{ (\mathbf{x} - \hat{\mathbf{x}}_t)^T V_t^{1/2} (\mathbf{x} - \hat{\mathbf{x}}_t) \}
\end{aligned}$$

Popular Examples

- SGD

$$\phi_t(\mathbf{g}_1,\ldots,\mathbf{g}_t)=\mathbf{g}_t,\quad \psi_t(\mathbf{g}_1,\ldots,\mathbf{g}_t)=\mathbb{I}$$

- AdaGrad

$$\phi_t(\mathbf{g}_1,\ldots,\mathbf{g}_t)=\mathbf{g}_t,\quad \psi_t(\mathbf{g}_1,\ldots,\mathbf{g}_t)=\frac{\mathrm{diag}(\sum_{\tau=1}^t \mathbf{g}_\tau^2)}{t}$$

- Adam

$$\phi_t(\mathbf{g}_1,\ldots,\mathbf{g}_t)=(1-\alpha)\sum_{\tau=1}^t \alpha^{t-\tau}\mathbf{g}_\tau,\quad \psi_t(\mathbf{g}_1,\ldots,\mathbf{g}_t)=(1-\beta)\mathrm{diag}(\sum_{\tau=1}^t \beta^{t-\tau}\mathbf{g}_\tau^2)$$

In other words, $\mathbf{m}_t=\alpha\mathbf{m}_{t-1}+(1-\alpha)\mathbf{g}_t$, $V_t=\beta V_{t-1}+(1-\beta)\mathrm{diag}(\mathbf{g}_t^2)$.
(All operations on vectors are element-wise)

ADAM

ADAM ≈ RMSProp + Momentum (>100K citations)

$$\begin{cases} \mathbf{v}_t & = \beta \mathbf{v}_{t-1} + (1-\beta)\nabla f(\mathbf{x}_t, \boldsymbol{\xi}_t)^2 \\ \mathbf{m}_t & = \alpha \mathbf{m}_{t-1} + (1-\alpha)\nabla f(\mathbf{x}_t, \boldsymbol{\xi}_t) \\ \mathbf{x}_{t+1} & = \mathbf{x}_t - \frac{\gamma_0}{\varepsilon + \sqrt{\tilde{\mathbf{v}}_t}} \tilde{\mathbf{m}}_t \end{cases}$$

- Exponential decay of previous information $\mathbf{m}_t, \mathbf{v}_t$.
- Note $\tilde{\mathbf{v}}_t = \frac{\mathbf{v}_t}{1-\beta^t}$ and $\tilde{\mathbf{m}}_t = \frac{\mathbf{m}_t}{1-\alpha^t}$ are bias-corrected estimates.
- In practice, α and β are chosen to be close to 1.

The Non-Convergence of Adam

Counterexample: consider a one-dimensional problem:

$$X = [-1, 1],\ f(x, \xi) = \begin{cases} Cx, & \text{if } \xi = 1 \\ -x, & \text{if } \xi = 0 \end{cases},\ P(\xi = 1) = p = \frac{1+\delta}{C+1}.$$

- Here $F(x) = \mathbb{E}[f(x, \xi)] = \delta x$ and $x^* = -1$.
- Adam step is $x_{t+1} = x_t - \gamma_0 \Delta_t$ with $\Delta_t = \frac{\alpha m_t + (1-\alpha) g_t}{\sqrt{\beta v_t + (1-\beta) g_t^2}}$
- For large enough $C > 0$, one can show that $\mathbb{E}[\Delta_t] \leq 0$.
- Adam steps keep drifting away from the optimal solution $x^* = -1$.

A Convergent Adam-type Algorithm

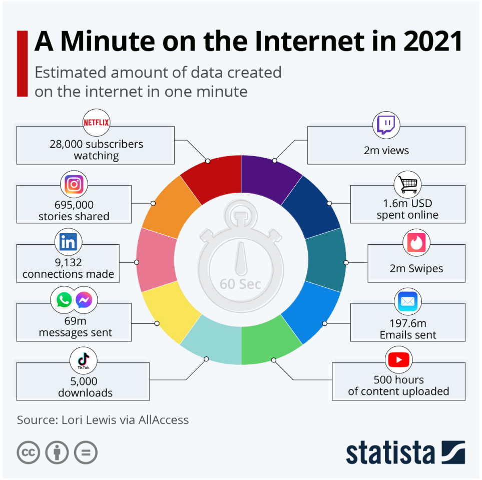
AMSGrad [Reddi, Kale, & Kumar (2018)]

Algorithm 2 AMSGRAD

```
Input: $x_1 \in \mathcal{F}$, step size $\{\alpha_t\}_{t=1}^T$, $\{\beta_{1t}\}_{t=1}^T$, $\beta_2$
Set $m_0 = 0$, $v_0 = 0$ and $\hat{v}_0 = 0$
for $t = 1$ to $T$ do
    $g_t = \nabla f_t(x_t)$
    $m_t = \beta_{1t} m_{t-1} + (1 - \beta_{1t}) g_t$
    $v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$
    $\hat{v}_t = \max(\hat{v}_{t-1}, v_t)$ and $\hat{V}_t = \mathrm{diag}(\hat{v}_t)$
    $x_{t+1} = \Pi_{\mathcal{F}, \sqrt{\hat{V}_t}}(x_t - \alpha_t m_t / \sqrt{\hat{v}_t})$
end for
```

- Use maximum value for normalizing the running average of the gradient.
- Ensure non-increasing stepsize and avoid pitfalls of Adam and RMSProp.
- Allow long-term memory of past gradients.

Modern Big Data Challenge



Big n !

- Cannot afford computing the gradient
- Cannot afford going through data many times

SGD vs. GD for Finite Sum Problem

$$\min_{\mathbf{x}\in\mathbb{R}^d} \quad F(\mathbf{x}) := \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{x})$$

Table: Complexity for smooth and strongly convex problems: $\kappa = L/\mu$

	iteration complexity	per-iteration cost	total cost
GD			
SGD			

- GD converges faster but with expensive iteration cost
- SGD converges slowly but with cheap iteration cost
- SGD is more appealing for large n and moderate accuracy ϵ.

Can we achieve the best of both worlds?

- GD: deterministic, linear rate, $O(n)$ iteration cost, fixed stepsize.
- SGD: stochastic, sublinear rate, $O(1)$ iteration cost, diminishing stepsize.

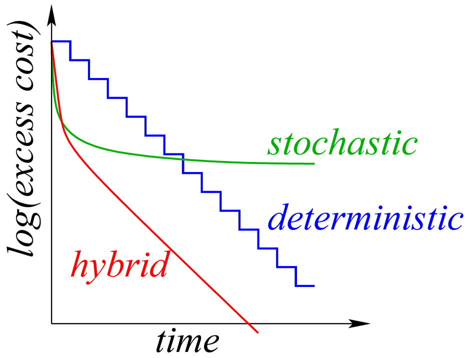


Figure from Bach's NeurIPS 2016 tutorial

Stochastic Varianced-reduced Methods

Stochastic variance-reduced methods are as cheap to update as SGD, but have as fast convergence as full gradient descent.

Popular algorithms:

- SAG (stochastic average gradient) [Le Roux et al., 2012]
- SVRG (stochastic variance-reduced gradient) [Johnson and Zhang, 2013]
- SDCA (stochastic dual coordinate ascent) [Shalev-Shwartz and Zhang, 2013]
- SAGA (stochastic average gradient amélioré) [Defazio et al., 2014]
- Many many others: MISO, Finito, Catalyst-SVRG, S2GD, etc.
- Recent variants for nonconvex setting: SPIDER, SARAH, STORM, PAGE, etc.

Preview of VR Methods

Algorithm	# of Iterations	Per-iteration Cost
GD	$O\left(\kappa \log \frac{1}{\epsilon}\right)$	$O(n)$
SGD	$O\left(\frac{\kappa}{\epsilon}\right)$	$O(1)$
VR	$O\left((n+\kappa) \log \frac{1}{\epsilon}\right)$	$O(1)$

Table: Complexity of strongly convex and smooth finite-sum optimization

Preview of VR Methods

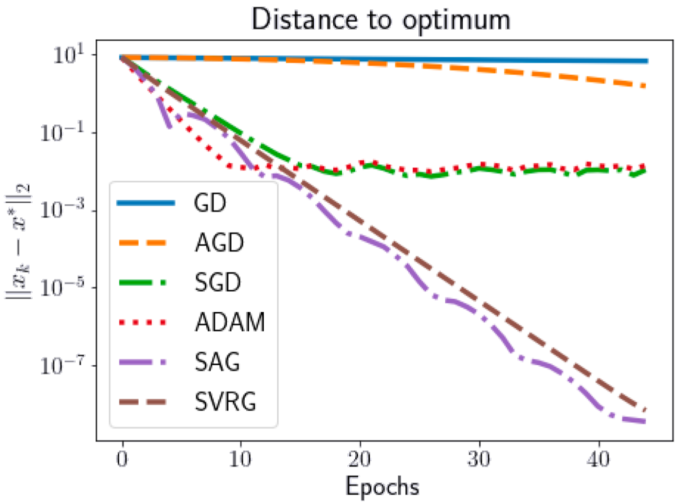


Figure: Logistic regression on `mushrooms` dataset with $n = 8124$ [Gow20]

Lecture Outline

Classical Variance Reduction Techniques

$$\min_{\mathbf{x}\in\mathbb{R}^d} \quad F(\mathbf{x}) := \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{x})$$

- Mini-batching: Use the average of gradients from a random subset

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \gamma_t \frac{1}{|B_t|}\sum_{i\in B_t} \nabla f_i(\mathbf{x}_t)$$

  NB: Variance reduction comes at a computational cost.

- Momentum: add momentum to the gradient step

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \gamma_t \hat{\mathbf{m}}_t, \text{ where } \hat{\mathbf{m}}_t = c \cdot \sum\nolimits_{\tau=1}^{t} \alpha^{t-\tau} \nabla f_{i_\tau}(\mathbf{x}_\tau)$$

  NB: Here $\mathbf{m}_t$ is the weighted average of the past stochastic gradients.

A Modern Variance Reduction Technique

Suppose we want to estimate $\theta = \mathbb{E}[X]$ where X is a random variable.
Consider the **point estimator** for θ:

$$\hat{\theta} := X - (Y - \mathbb{E}[Y])$$

- $\mathbb{E}[\hat{\theta}] = \theta$.
- $\mathbb{V}[\hat{\theta}]$ is less than $\mathbb{V}[X]$ if Y is highly positively correlated with X.

A Modern Variance Reduction Technique

Suppose X is positively correlated with Y and we can compute $\mathbb{E}[Y]$.

Point Estimator:

$$\hat{\theta}_\alpha = \alpha(X - Y) + \mathbb{E}[Y], \quad (0 \leq \alpha \leq 1).$$

$$\mathbb{E}[\hat{\theta}_\alpha] = \alpha \mathbb{E}[X] + (1 - \alpha)\mathbb{E}[Y]$$
$$\mathbb{V}[\hat{\theta}_\alpha] = \alpha^2 (\mathbb{V}[X] + \mathbb{V}[Y] - 2\mathrm{Cov}[X, Y])$$

- If covariance is sufficiently large, then $\mathbb{V}[\hat{\theta}_\alpha] \leq \mathbb{V}[X]$.

Clicker Question

this part does not know, it seems have some missing about statistics.

Recall $\hat{\theta}_\alpha = \alpha(X - Y) + \mathbb{E}[Y]$ and $\mathrm{Cov}[X, Y] > 0$.
Which one of the following statements about $\hat{\theta}_\alpha$ is NOT correct?

A. If $\alpha = 1$, the estimator is unbiased.

B. If $\mathbb{E}[Y] = \mathbb{E}[X]$, the estimator is unbiased for any α.

C. The bias increases as α increases from 0 to 1.

D. The variance increases as α increases from 0 to 1.

Motivation

Q: Can we design cheap gradient estimators with reduced variance?

Key Idea: if $\mathbf{x}_t$ is not too far away from previous iterates, then we can leverage previous gradient information to construct positively correlated control variates.

- SGD: estimate $\nabla F(\mathbf{x}_t)$ by $\nabla f_{i_t}(\mathbf{x}_t)$
- VR: estimate $\nabla F(\mathbf{x}_t)$ by $\mathbf{g}_t := \alpha(\nabla f_{i_t}(\mathbf{x}_t) - Y) + \mathbb{E}[Y]$ such that

$$\mathbb{E}[\|\mathbf{g}_t - \nabla F(\mathbf{x}_t)\|^2] \to 0, \text{ as } t \to \infty. \qquad \text{(VR property)}$$

So how to design Y?

Design Ideas

Goal: Construct Y that is positively correlated to $X = \nabla f_{i_t}(\mathbf{x}_t)$:

Choice I: $Y = \nabla f_{i_t}(\mathbf{x}^\star)$, where $\mathbf{x}^\star$ is the optimal solution

- $\mathbb{E}[Y] = 0$, unrealistic but conceptually useful

Choice II: $Y = \nabla f_{i_t}(\bar{\mathbf{x}}_{i_t})$, where $\bar{\mathbf{x}}_i$ is the last point for which we evaluated $\nabla f_i(\bar{\mathbf{x}}_i)$

- $\mathbb{E}[Y] = \frac{1}{n}\sum_{i=1}^n \nabla f_i(\bar{\mathbf{x}}_i)$, requires storage of $\{\bar{\mathbf{x}}_i\}_{i=1}^n$ or $\{\nabla f_i(\bar{\mathbf{x}}_i)\}_{i=1}^n$

Choice III: $Y = \nabla f_{i_t}(\tilde{\mathbf{x}})$, where $\tilde{\mathbf{x}}$ is some fixed reference point

- $\mathbb{E}[Y] = \frac{1}{n}\sum_{i=1}^n \nabla f_i(\tilde{\mathbf{x}})$, requires computing the full gradient at $\tilde{\mathbf{x}}$

Lecture Outline

Variance Reduction Techniques for Finite Sum Problems

Goal: estimate $\theta = \nabla F(\mathbf{x}_t)$, $X = \nabla f_{i_t}(\mathbf{x}_t)$

- SGD: $\mathbf{g}_t = \nabla f_{i_t}(\mathbf{x}_t)$ $[\alpha = 1, Y = 0]$
- SAG: $\mathbf{g}_t = \frac{1}{n}(\nabla f_{i_t}(\mathbf{x}_t) - \mathbf{v}_{i_t}) + \frac{1}{n}\sum_{i=1}^n \mathbf{v}_i$ $[\alpha = \frac{1}{n}, Y = \mathbf{v}_{i_t}]$
- SAGA: $\mathbf{g}_t = (\nabla f_{i_t}(\mathbf{x}_t) - \mathbf{v}_{i_t}) + \frac{1}{n}\sum_{i=1}^n \mathbf{v}_i$ $[\alpha = 1, Y = \mathbf{v}_{i_t}]$

  Here $\{\mathbf{v}_i, i = 1, \ldots, n\}$ are the past stored gradients for each component.
- SVRG: $\mathbf{g}_t = \nabla f_{i_t}(\mathbf{x}_t) - \nabla f_{i_t}(\tilde{\mathbf{x}}) + \nabla F(\tilde{\mathbf{x}})$ $[\alpha = 1, Y = \nabla f_{i_t}(\tilde{\mathbf{x}})]$

Stochastic Average Gradient (SAG)

Idea: keep track of the average of $\mathbf{v}_i$ as an estimate of the full gradient

$$\mathbf{g}_t = \frac{1}{n}\sum_{i=1}^{n} \mathbf{v}_i^t \qquad \approx \qquad \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(\mathbf{x}_t) = \nabla F(\mathbf{x}_t)$$

- The past gradients are updated as:

$$\mathbf{v}_i^t = \begin{cases} \nabla f_{i_t}(\mathbf{x}_t), & \text{if } i = i_t, \\ \mathbf{v}_i^{t-1}, & \text{if } i \neq i_t. \end{cases}$$

- Equivalently, we have

$$\mathbf{g}_t = \mathbf{g}_{t-1} - \frac{1}{n}\mathbf{v}_{i_t}^{t-1} + \frac{1}{n}\nabla f_{i_t}(\mathbf{x}_t)$$

Stochastic Average Gradient (SAG, continued)

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \frac{\gamma}{n}\sum_{i=1}^{n} \mathbf{v}_i^t, \text{ where } \mathbf{v}_i^t = \begin{cases} \nabla f_{i_t}(\mathbf{x}_t), & \text{if } i = i_t \\ \mathbf{v}_i^{t-1}, & \text{otherwise} \end{cases}$$

Algorithm SAG (Le Roux et al., 2012)

```
1: Initialize $\mathbf{v}_i = 0, i = 1, \ldots, n$
2: for $t = 1, 2, \ldots, T$ do
3:     Randomly pick $i_t \in \{1, 2, \ldots, n\}$
4:     $\mathbf{g}_t = \mathbf{g}_{t-1} - \frac{1}{n}\mathbf{v}_{i_t}$
5:     $\mathbf{v}_{i_t} = \nabla f_{i_t}(\mathbf{x}_t)$
6:     $\mathbf{g}_t = \mathbf{g}_t + \frac{1}{n}\mathbf{v}_{i_t}$
7:     $\mathbf{x}_{t+1} = \mathbf{x}_t - \gamma \mathbf{g}_t$
8: end for
```

- Biased gradient
- Cheap iteration cost
- $O(nd)$ memory cost
- Hard to analyze

Stochastic Average Gradient (SAG, continued)

- **Linear convergence:** The first stochastic methods to enjoy linear rate using a constant stepsize for strongly-convex and smooth objectives.

  If F is μ-strongly convex and each f_i is L_i-smooth and convex, setting $\gamma = 1/(16L_{\max})$, one can show that

  $$\mathbb{E}[F(\mathbf{x}_t) - F(\mathbf{x}^*)] \le C \cdot \left(1 - \min\{\frac{1}{8n}, \frac{\mu}{16L_{\max}}\}\right)^t.$$

  Here $L_{\max} := \max\{L_1, \dots, L_n\}$.

- **Memory cost:** $O(n)$ times higher than SGD/SVRG
- **Per-iteration cost**: one gradient evaluation
- Total complexity: $O\left((n + \kappa_{\max})\log(\frac{1}{\epsilon})\right)$.

SAGA

SAGA (Defazio, Bach, Lacoste-Julien, 2016):

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \gamma \left[(\nabla f_{i_t}(\mathbf{x}_t) - \mathbf{v}_{i_t}^{t-1}) + \frac{1}{n} \sum_{i=1}^{n} \mathbf{v}_i^{t-1} \right]$$

- Unbiased update, while SAG is biased
- Same $O(nd)$ memory cost as SAG
- Similar linear convergence rate as SAG, but has a much simpler proof

Stochastic Variance Reduced Gradient (SVRG)

Key idea: Build covariates based on fixed reference point; balance the frequency of reference point update and the variance reduction.

Algorithm Stochastic Variance Reduced Gradient (Johnson & Zhang '13)

```
1: for $s = 1, 2, \ldots$ do
2:     Set $\tilde{\mathbf{x}} = \tilde{\mathbf{x}}^{s-1}$ and compute $\nabla F(\tilde{\mathbf{x}}) = \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(\tilde{\mathbf{x}})$        (update snapshot)
3:     Initialize $\mathbf{x}_0 = \tilde{\mathbf{x}}$
4:     for $t = 0, 1, \ldots, m-1$ do
5:         Randomly pick $i_t \in \{1, 2, \ldots, n\}$ and update
6:             $\mathbf{x}_{t+1} = \mathbf{x}_t - \eta\left(\nabla f_{i_t}(\mathbf{x}_t) - \nabla f_{i_t}(\tilde{\mathbf{x}}) + \nabla F(\tilde{\mathbf{x}})\right)$        (cheap cost)
7:     end for
8:     Update $\tilde{\mathbf{x}}^s = \frac{1}{m}\sum_{t=0}^{m-1} \mathbf{x}_t$
9: end for
```

SVRG: Key Features

Intuition: the closer $\tilde{\mathbf{x}}$ is to $\mathbf{x}_t$, the smaller the variance of the gradient estimator

$$\mathbb{E}[\|\mathbf{g}_t - \nabla F(\mathbf{x}_t)\|^2] \leq \mathbb{E}[\|\nabla f_{i_t}(\mathbf{x}_t) - \nabla f_{i_t}(\tilde{\mathbf{x}})\|^2] \leq L_{\max}^2 \|\mathbf{x}_t - \tilde{\mathbf{x}}\|^2$$

Two-loop structure:

- Outer loop: update reference point and compute its full gradient at $O(n)$ cost
- Inner loop: update iterates with variance-reduced gradient for m steps
- Total of $O(n + 2m)$ component gradient evaluations at each epoch

Compare to SAG/SAGA

$(+)$ Cheap memory cost, no need to store past gradients or past iterates

$(-)$ More parameter tuning, two gradient computation per iteration

Convergence of SVRG

Theorem 12.1 (Johnson & Zhang, 2013)

Assume each $f_i(\mathbf{x})$ is convex and L_i-smooth, $F(\mathbf{x})$ is μ-strongly convex. Assume m is sufficiently large and $\eta < \frac{1}{2L_{max}}$ such that $\rho = \frac{1}{\mu\eta(1-2\eta L_{max})m} + \frac{2\eta L_{max}}{1-2\eta L_{max}} < 1$, then

$$\mathbb{E}[F(\tilde{\mathbf{x}}^s) - F(\mathbf{x}^*)] \le \rho^s\,[F(\tilde{\mathbf{x}}^0) - F(\mathbf{x}^*)].$$

- **Linear convergence:** choose $m = O(\frac{L_{\max}}{\mu}), \eta = O(\frac{1}{L_{\max}})$ such that $\rho \in (0, \frac{1}{2})$.
- **Total complexity**:

$$O\left((2m+n)\log\frac{1}{\epsilon}\right) = O\left(\left(n + \frac{L_{\max}}{\mu}\right)\log\frac{1}{\epsilon}\right).$$

SVRG vs. SAG/SAGA

this table seems important for true or false questions.

Table: Comparisons between SVRG and SAG/SAGA

	SVRG	**SAG/SAGA**
memory cost	$O(d)$	$O(nd)$
epoch-based	yes	no
# gradients per step	at least 2	1
parameters	stepsize & epoch length	stepsize
unbiasedness	yes	yes/no
total complexity	$O\left((n+\kappa_{\max})\log\frac{1}{\epsilon}\right)$	$O\left((n+\kappa_{\max})\log\frac{1}{\epsilon}\right)$

Loopless-SVRG: [Hofmann et al., 2015][Kovalev el al., 2020]

Numerical Illustration

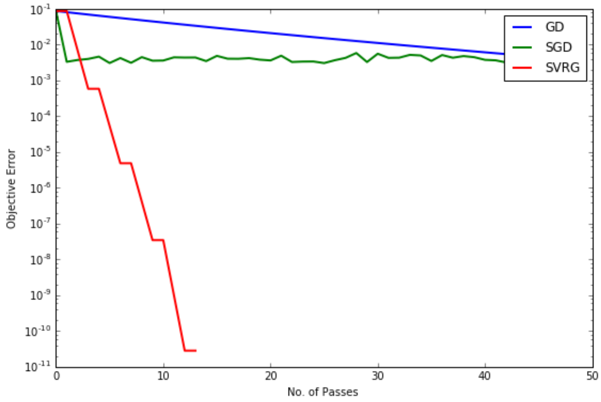


Figure: Numerical illustration among GD, SGD, SVRG on logistic regression.

Convergence Analysis of SVRG: Key Lemma

Lemma 12.2 (Exercise, use smoothness and convexity)

$$\frac{1}{n}\sum_{i=1}^{n}\|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{x}^*)\|_2^2 \leq 2L_{max}(F(\mathbf{x}) - F(\mathbf{x}^*))$$

Lemma 12.3 (Bound of variance)

Denote $\mathbf{g}_t = \nabla f_{i_t}(\mathbf{x}^t) - \nabla f_{i_t}(\tilde{\mathbf{x}}) + \nabla F(\tilde{\mathbf{x}})$. *We have*

$$\mathbb{E}[\|\mathbf{g}_t\|_2^2] \leq 4L_{max}[F(\mathbf{x}_t) - F(\mathbf{x}^*) + F(\tilde{\mathbf{x}}) - F(\mathbf{x}^*)].$$

- Follows from the previous lemma using a suitable decomposition by introducing $\nabla f_{i_t}(\mathbf{x}^*)$ in $\mathbf{g}_t$.

Convergence Analysis of SVRG: Proof

For notation simplicity, denote $L = L_{\max}$. From Lemma 12.3, we have

$$
\begin{aligned}
&\mathbb{E}\left[\|\mathbf{x}_{t+1} - \mathbf{x}^*\|_2^2\right] \\
&= \|\mathbf{x}_t - \mathbf{x}^*\|_2^2 - 2\eta(\mathbf{x}_t - \mathbf{x}^*)^T \mathbb{E}\left[\mathbf{g}_t\right] + \eta^2 \mathbb{E}\left[\|\mathbf{g}_t\|_2^2\right] \\
&\leq \|\mathbf{x}_t - \mathbf{x}^*\|_2^2 - 2\eta(1 - 2L\eta)(F(\mathbf{x}_t) - F(\mathbf{x}^*)) + 4L\eta^2\left[F(\tilde{\mathbf{x}}) - F(\mathbf{x}^*)\right]
\end{aligned}
$$

We can then establish the contraction after telescoping the sum and invoking the definition for $\tilde{\mathbf{x}}$.

Convergence Analysis of SVRG: Proof (continued)

Then we have

$$\begin{aligned}
&\mathbb{E}\left[\|\mathbf{x}_m - \mathbf{x}^*\|^2\right] + 2\eta(1-2L\eta)m\mathbb{E}\left[F(\tilde{\mathbf{x}}^s) - F(\mathbf{x}^*)\right] \\
&\leq \mathbb{E}\left[\|\mathbf{x}_m - \mathbf{x}^*\|^2\right] + 2\eta(1-2L\eta)\sum_{t=0}^{m-1}\mathbb{E}\left[F(\mathbf{x}_t) - F(\mathbf{x}^*)\right] && \text{(by convexity)} \\
&\leq \mathbb{E}\left[\|\mathbf{x}_0 - \mathbf{x}^*\|^2\right] + 4Lm\eta^2\mathbb{E}\left[F(\tilde{\mathbf{x}}^{s-1}) - F(\mathbf{x}^*)\right] && \text{(by telescoping)} \\
&= \mathbb{E}\left[\|\tilde{\mathbf{x}}^{s-1} - \mathbf{x}^*\|^2\right] + 4Lm\eta^2\mathbb{E}\left[F(\tilde{\mathbf{x}}^{s-1}) - F(\mathbf{x}^*)\right] && \text{(by definition of } \mathbf{x}_0\text{)} \\
&\leq \frac{2}{\mu}\mathbb{E}\left[F(\tilde{\mathbf{x}}^{s-1}) - F(\mathbf{x}^*)\right] + 4Lm\eta^2\mathbb{E}\left[F(\tilde{\mathbf{x}}^{s-1}) - F(\mathbf{x}^*)\right] && \text{(by } \mu \text{ strong-convexity)}
\end{aligned}$$

This further implies

$$\mathbb{E}\left[F(\tilde{\mathbf{x}}^s) - F(\mathbf{x}^*)\right] \leq \left[\frac{1}{\mu\eta(1-2L\eta)m} + \frac{2L\eta}{1-2L\eta}\right]\mathbb{E}\left[F(\tilde{\mathbf{x}}^{s-1}) - F(\mathbf{x}^*)\right].$$

□

Summary: Finite Sum Optimization

$$\min_{\mathbf{x}\in\mathbb{R}^d} \quad F(\mathbf{x}) := \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{x})$$

(f_i is L_i-smooth and convex, F is L-smooth and μ-strongly convex)

Algorithm	# of Iterations	Per-iteration Cost
GD	$O\left(\kappa \log \frac{1}{\epsilon}\right)$	$O(n)$
SGD	$O\left(\frac{\kappa_{\max}}{\epsilon}\right)$	$O(1)$
SAG/SAGA/SVRG	$O\left((n+\kappa_{\max}) \log \frac{1}{\epsilon}\right)$	$O(1)$

Table: Complexity of finite-sum optimization, $\kappa = \frac{L}{\mu}$, $\kappa_{\max} = \frac{L_{\max}}{\mu}$

Remarks

- Variance reduction technique is crucial for finite sum problems.
- In general, $L \leq L_{\max} \leq nL$. VR methods are always superior in terms of total runtime than GD.
- If $L_i = L, \forall i$, then $\kappa = \kappa_{\max}$, VR methods are much faster than GD especially when $\kappa = O(n)$.
- SGD has much worse dependency on ϵ than VR methods, which explain its poor performance when ϵ is small.

Can we further improve the VR methods?

- Non-uniform sampling: improve to $O\left((n + \kappa_{\text{avg}}) \log \frac{1}{\epsilon}\right)$

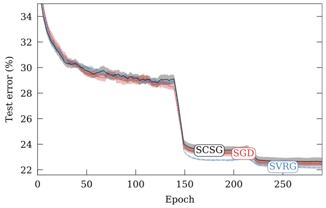
$$P(i_t = i) = \frac{L_i}{\sum_{i=1}^{n} L_i}$$

- Incorporating acceleration: can improve to $O\left((n + \sqrt{n\kappa_{\max}}) \log \frac{1}{\epsilon}\right)$.

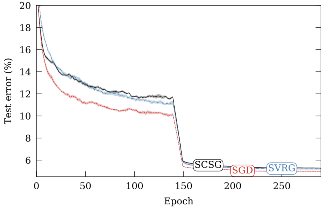
- Lower complexity bound: $O\left((n + \sqrt{n\kappa_{\max}}) \log \frac{1}{\epsilon}\right)$ for the strongly-convex and smooth finite-sum problems considered (Woodworth and Srebro, 2016; Lan and Zhou, 2018)

Limitations

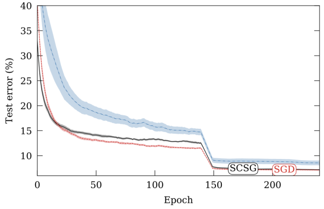
- Challenges with practical implementations: learning rate and sampling
- VR may be ineffective for training neural networks [Defazio and Bottou, 2019].



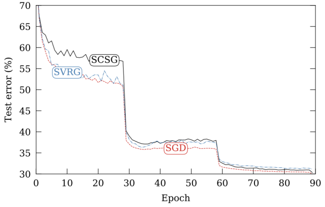
(a) LeNet on CIFAR10



(b) DenseNet on CIFAR10



(c) ResNet-110 on CIFAR10



(d) ResNet-18 on ImageNet

Lecture Outline

Nonconvex SGD vs. GD

For smooth functions with finite-sum structure $F(\mathbf{x}) := \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{x})$

- GD finds a point with $\|\nabla F(\bar{\mathbf{x}})\| \leq \epsilon$ in $O(n/\epsilon^2)$ gradient evaluations.
- SGD finds a point with $\mathbb{E}[\|\nabla F(\bar{\mathbf{x}})\|] \leq \epsilon$ in $O(1/\epsilon^4)$ gradient evaluations.

Q. Can we improve the complexity bound with variance reduction techniques?

- Algorithms: SPIDER [Fang et al., 2018], SARAH [Nguyen et al., 2017], STORM [Cutkosky & Orabona, 2019], ...
- Complexity: $O(\min\{\sqrt{n}/\epsilon^2, \epsilon^{-3}\})$ for *average-smooth* functions.
- Average-smoothness: $\mathbb{E}_i\|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{y})\|^2 \leq L^2\|\mathbf{x} - \mathbf{y}\|^2$.

Variance-reduced Methods for Nonconvex Optimization

$$\min_{\mathbf{x}\in\mathbb{R}^d} \quad F(\mathbf{x}) := \mathbb{E}_{\boldsymbol{\xi}}[f(\mathbf{x},\boldsymbol{\xi})] \qquad \left[\text{ or } F(\mathbf{x}) := \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{x})\right]$$

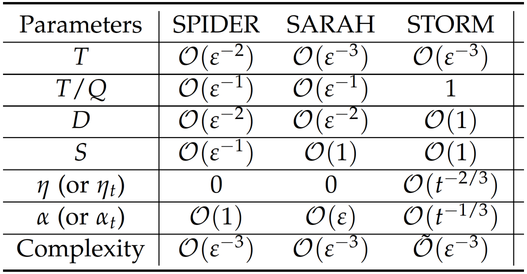
SGD with variance-reduced recursive gradient estimator:

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \alpha \mathbf{g}_t,$$

$$\mathbf{g}_t = \begin{cases} \frac{1}{D}\sum_{i=1}^{D} \nabla f(\mathbf{x}_t, \boldsymbol{\xi}_t^i), & \text{if } t \equiv 0 (\text{mod } Q) \\ (1-\eta)(\mathbf{g}_{t-1} - \frac{1}{S}\sum_{j=1}^{S}\nabla f(\mathbf{x}_{t-1},\boldsymbol{\xi}_t^j)) + \frac{1}{S}\sum_{j=1}^{S}\nabla f(\mathbf{x}_t,\boldsymbol{\xi}_t^j), & \text{otherwise.} \end{cases}$$

Variance-reduced SGD for Nonconvex Optimization

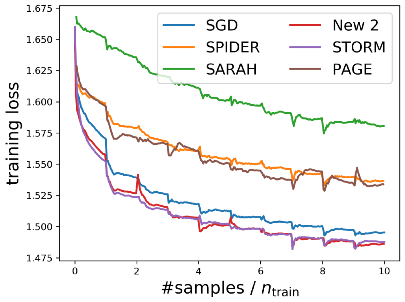
- Total gradient complexity: $O(T(2S + \frac{D}{Q}))$



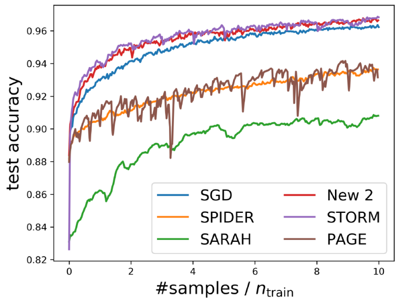
Parameters	SPIDER	SARAH	STORM
T	$\mathcal{O}(\varepsilon^{-2})$	$\mathcal{O}(\varepsilon^{-3})$	$\mathcal{O}(\varepsilon^{-3})$
T/Q	$\mathcal{O}(\varepsilon^{-1})$	$\mathcal{O}(\varepsilon^{-1})$	1
D	$\mathcal{O}(\varepsilon^{-2})$	$\mathcal{O}(\varepsilon^{-2})$	$\mathcal{O}(1)$
S	$\mathcal{O}(\varepsilon^{-1})$	$\mathcal{O}(1)$	$\mathcal{O}(1)$
η (or η_t)	0	0	$\mathcal{O}(t^{-2/3})$
α (or α_t)	$\mathcal{O}(1)$	$\mathcal{O}(\varepsilon)$	$\mathcal{O}(t^{-1/3})$
Complexity	$\mathcal{O}(\varepsilon^{-3})$	$\mathcal{O}(\varepsilon^{-3})$	$\tilde{\mathcal{O}}(\varepsilon^{-3})$

Figure: Parameter choices of different VR-SGD methods: T stands for iteration complexity, T/Q for number of epochs, D for batch size at checkpoints, S for batch size at other iterations, η for the momentum parameter and α for the stepsize

Variance-reduced SGD for Nonconvex Optimization



(a) Training Loss



(b) Test Accuracy

Figure 5.4: The performances of different algorithms for the multi-class classification task on the MNIST dataset using a three-layer neural network. Both metrics are plotted against number of samples used during training with $n_{\text{train}} = 50000$ as the training size.

Figure source: [Zha21]

Bibliography (cont'd)

R. M. Gower, M. Schmidt, F. Bach, P. Richtarik
Variance-reduced methods for machine learning.
Proceedings of the IEEE, 108(11), 1968-1983, 2022.

D. Kovalev, S. Horváth, P. Richtárik.
Don't jump through hoops and remove those loops: SVRG and Katyusha are better without the outer loop.
In Algorithmic Learning Theory, 2020.

L. Zhang
Variance Reduction for Non-Convex Stochastic Optimization: General Analysis and New Applications.
ETH Master Thesis, 2021.