Variance reduction for stochastic gradient methods

Axel Böhm

January 7, 2022

The finite sum problem

A common Task in (supervised) machine learning:

$$\min_{x\in\mathbb{R}^d} f(x) := \frac{1}{n}\sum_{i=1}^{n} \underbrace{f_i(x)}_{\text{loss for } i\text{-th sample}} + \underbrace{\psi(x)}_{\text{regularizer}}$$

where the i-th sample is (a_i, y_i).

Examples:

- linear regression: $f_i(x) = (a_i^T x - y_i)^2$, and $\psi = 0$
- logistic regression: $f_i(x) = \log(1 + e^{-y_i a_i^T x})$, and $\psi = 0$ "sigmoid function" and logistic loss.
- Lasso: f_i as for linear regression but $\psi(x) = \|x\|_1$
- SVM: $f_i(x) = \max\{0, 1 - y_i a_i^T x\}$ and $\psi(x) = \|x\|^2$

Gradient descent

Algorithm (batch) GD

1: **for** $k = 1, 2, \ldots$ **do**
2: $\quad x_{k+1} = x_k - \alpha_k \nabla f(x_k)$

- gradient can be computed via

$$\nabla f(x) = \nabla \left(\sum_{i=1}^{n} f_i(x) \right) = \sum_{i=1}^{n} \nabla f_i(x_k)$$

- good convergence properties
- can be **expensive** if n is large!

Stochastic gradient descent

Algorithm SGD

1: **for** $k = 1, 2, \ldots$ **do**
2: pick i_k uniform at random in $[n]$
3: $x_{k+1} = x_k - \alpha_k \nabla f_{i_k}(x_k)$

We already noticed that:

- unbiased: $\mathbb{E}[\nabla f_{i_k}(x)] = \sum_{i=1}^n \mathbb{P}[i = i_k] \nabla f_i(x) = \sum_{i=1}^n \frac{1}{n} \nabla f_i(x)$
- large stepsizes fail to suppress noise in the stoch. gradients $\rightarrow$ leads to oscillations
- decreasing stepsizes mitigate this problem but **slows down** convergence (too *conservative*)

Recall SGD

template

$$x_{k+1} = x_k - \alpha_k g_k$$

- g_k is an unbiased estimator of the true gradient $\nabla F(x_k)$
- convergence depends on **variance** $\mathbb{E}[\|g_k - \nabla F(x_k)\|] \leq \sigma_g$ (not strictly necessary)
- vanilla SGD uses $g_k = \nabla f_{i_k}(x_k)$
  issue: σ_g is non-negligible even close to the solution
- **Q:** can we choose g_k in a different way to reduce variability?

Minibatching

Algorithm minibatch SGD

1: **for** $k = 1, 2, \ldots$ **do**
2: pick I_k random subset of $[n]$ with $|I_k| = b$
3: $x_{k+1} = x_k - \alpha_k \sum_{i \in I_k} \nabla f_i(x_k)$

- typically we make a (uniform) *random* choice $i_k \in [n] = \{1, \ldots, n\}$ (or random reshuffling)
- by increasing the size to a **random subset** $I_k \subset [n]$ of size $b \ll n$ we can
  - decrease variance
  - increase cost only moderately,
  - no improvement in the rate

A simple idea

Consider

- estimator X for parameter μ ($\mathbb{E}[X] = \mu$ and $\mathbb{V}[X] = \sigma^2$)
- want to keep unbiased but **reduce variance**
- find Y such that $\mathbb{E}[Y] = 0$ but $\mathrm{Cov}(X, Y)$ **is large** and define

$$\tilde{X} := X - Y$$

- remains unbiased
- $\mathbb{V}[\tilde{X}]$ can be much smaller than $\mathbb{V}[X]$ if X, Y are highly correlated

$$\mathbb{V}[\tilde{X}] = \mathbb{V}[X] + \mathbb{V}[Y] - 2\,\mathrm{Cov}[X, Y]$$

Stochastic average gradient (SAG), 2013

- **maintain table** containing gradients g_i of f_i
- pick random $i_k \in [n]$ and

$$g_{i_k}^k := \nabla f_{i_k}(x^k)$$

  set $g_i^k = g_i^{k-1}$ for all $i \neq i_k$ (remain the same)
- Update

$$x^{k+1} = x^k - \alpha_k \frac{1}{n} \sum_{i=1}^n g_i^k.$$

- gradient estimator no longer unbiased
- Isn't it expensive to average these gradients?

$$x^{k+1} = x^k - \alpha_k \Big(\frac{g_{i_k}^k}{n} - \frac{g_{i_k}^{k-1}}{n} + \underbrace{\frac{1}{n} \sum_{i=1}^n g_i^{k-1}}_{\text{old table average}} \Big)$$

SAG variance reduction

Gradient estimator in SAG:

$$x^{k+1} = x^k - \alpha_k \frac{1}{n}\Big(\underbrace{g_{i_k}^k}_{X} \underbrace{- g_{i_k}^{k-1} - \sum_{i=1}^{n} g_i^{k-1}}_{Y} \Big)$$

- Indeed $\mathbb{E}[X] = \nabla f(x^k)$, but $\mathbb{E}[Y] \neq 0 \rightarrow$ is **biased estimator**
- X and Y are correlated as $X - Y \rightarrow 0$:
  - x^k and x_{k-1} both converge to $x^* \Rightarrow \nabla f_i(x_k) - \nabla f_i(x_{k-1}) \rightarrow 0$
  - the last term converges to $\nabla f(x^*) = 0$

Convergence

As always, initialization plays a role: $D^2 := \|x^0 - x^*\|^2$.

$$\text{SAG:} \quad \frac{n}{k}(f(x^0) - f^*) + \frac{L}{k}D^2$$

$$\text{GD:} \quad \frac{L}{k}D^2$$

$$\text{SGD:} \quad \frac{L}{\sqrt{k}}D^2$$

- ◇ Achieves linear convergence in the **strongly** convex setting.
- ◇ proofs are difficult (and computer-aided)

Same gradient oracle cost as SGD, but same converge rate as GD.

Experiments from the original paper

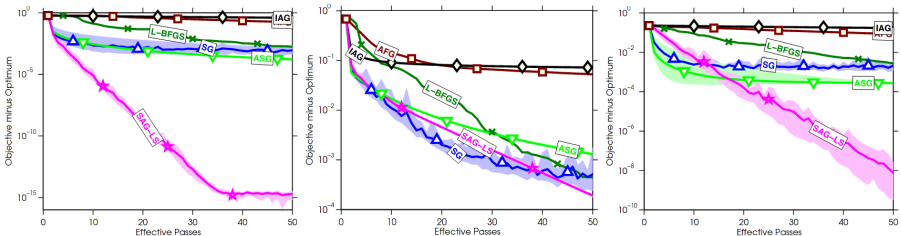
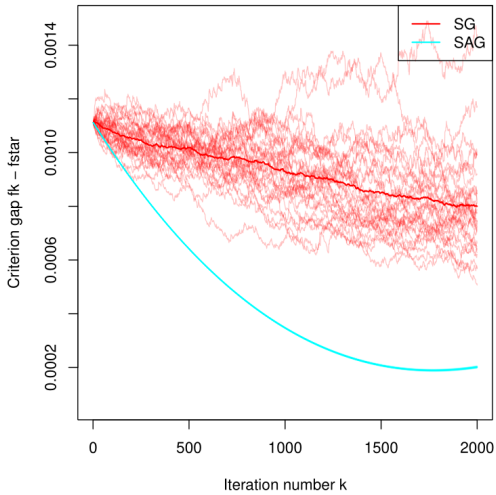


Figure: Solving ℓ_2-regularized logistic regression.

More "naive" implementation



SAG experiments

- does not work well out of the box
- needs a warm up to get good $(g_1^0, g_2^0, \ldots, g_m^0)$
  - achieved by running one full epoch of SGD
- reguires hand tuned stepsize or line search

SAGA, 2014

Very similar to SAG:

- maintain table containing gradients g_i of f_i
- pick random $i_k \in [n]$ and

$$g_{i_k}^k := \nabla f_{i_k}(x^k)$$

  set $g_i^k = g_i^{k-1}$ for all $i \neq i_k$ (remain the same)
- Update

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

- estimator now **unbiased**!

For Comparison

SAGA gradient estimate:

$$g_{i_k}^k - g_{i_k}^{k-1} + \frac{1}{n}\sum_{i=1}^n g_i^k.$$

SAG gradient estimate:

$$\frac{1}{n}g_{i_k}^k - \frac{1}{n}g_{i_k}^{k-1} + \frac{1}{n}\sum_{i=1}^n g_i^k.$$

Experiments from the original paper

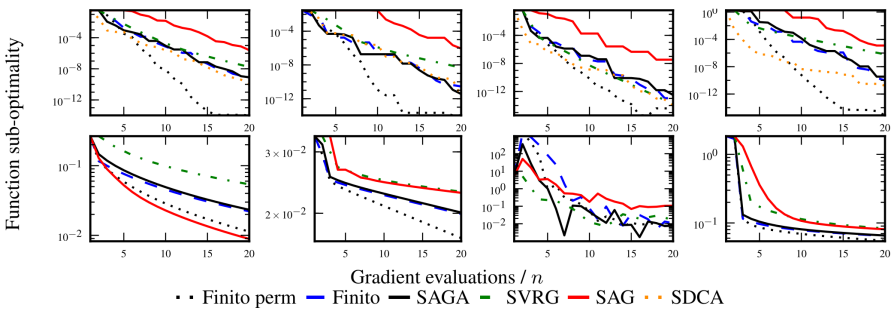
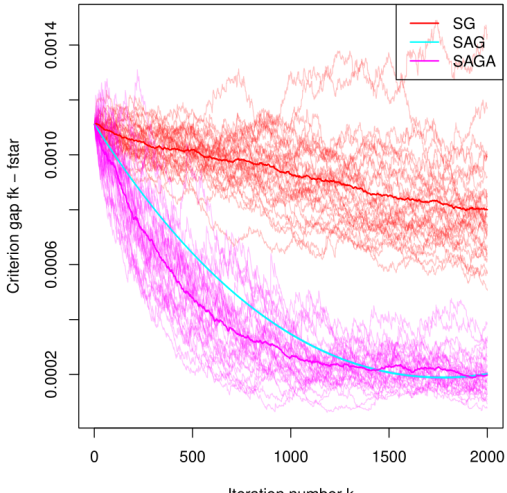


Figure: Solving regularized logistic regression. First row is ℓ_2-regularized; second row is ℓ_1.

More "naive" implementation



Stochastic Variance Reduced Gradient (SVRG), 2013

Algorithm SVRG

for $k = 1, 2, \ldots$ **do**
  Set $x^1 = \tilde{x} = \tilde{x}^k$
  Compute $\tilde{\mu} := \nabla f(\tilde{x})$ //update snapshot
  for $l = 1, 2, \ldots, m$ **do** //m iterations per epoch
    pick i_l uniform at random in $[n]$
    Set $x^{l+1} = x^l - \alpha(\nabla f_{i_l}(x^l) - \nabla f_{i_l}(\tilde{x}) + \tilde{\mu})$
  $\tilde{x}^{k+1} = x^{m+1}$

- Does **not need to store** full table of gradients.
- requires *batch* gradient computation every *epoch*
- per iteration cost is comparable to that of SGD if $m \geq n$
- convergence rates similar to SAGA, but simpler analysis.

SVRG

key idea: by storing old point we can

$$\underbrace{\nabla f_{i_k}(x^k) - \nabla f_{i_k}(x^{\text{old}})}_{\to 0 \text{ if } x \approx x^{\text{old}}} + \underbrace{\nabla f(x^{\text{old}})}_{\to 0 \text{ if } x^{\text{old}}}$$

- is an unbiased estimate of $\nabla f(x^k)$
- converges to 0 (meaning reduced variability) if $x^k \approx x^{\text{old}} \approx x^*$

SVRG: Theorem

Each f_i is convex and L-smooth, and sum is μ-strongly convex.

Theorem

Choose m large enough s.t. $\rho = \frac{1}{\mu\alpha(1-2L\alpha)m} + \frac{2L\alpha}{1-2L\alpha} < 1$, *then*

$$\mathbb{E}[F(x_s^{old}) - F(x^*)] \leq \rho^s [F(x_0^{old}) - F(x^*)]$$

Computational cost:

- per epoch: $(m + n)$
- inner loop is annoying (has to choose m) $\rightarrow$ loopless variant

SVRG: Convergence Proof

Denote $g_s^k = \nabla f_{i_l}(x_s^k) - \nabla f_{i_l}(x_s^{old}) + \nabla F(x_s^{old})$. Conditioning on everything prior to x_s^{k+1} we get

$$\begin{aligned}
\mathbb{E}[\|x_s^{k+1} - x^*\|^2] &= \mathbb{E}[\|x_s^k - \alpha g_s^k - x^*\|^2] \\
&= \|x_s^k - x^*\|^2 - 2\alpha(x_s^k - x^*)^T \mathbb{E}[g_s^t] + \alpha^2 \mathbb{E}[\|g_s^k\|^2] \\
&= \|x_s^k - x^*\|^2 - 2\alpha(x_s^k - x^*)^T \nabla F(x_s^t) + \alpha^2 \mathbb{E}[\|g_s^k\|^2] \\
&\leq \|x_s^k - x^*\|^2 - 2\alpha(F(x_s^k) - F(x^*)) + \alpha^2 \mathbb{E}[\|g_s^k\|^2]
\end{aligned}$$

- ◇ **key step:** control $\mathbb{E}[\|g_s^k\|^2]$

SVRG: convergence Proof

Lemma

$$\mathbb{E}[\|g_s^k\|^2] \leq 4L[F(x_s^k) - F(x^*) + F(x_s^{old} - F(x^*))]$$

Comparison

f is L-smooth and μ-strongly convex. Condition number: $\kappa = L/\mu$.

Strongly convex problems: number gradient calls to compute $\mathbb{E}[f(x_k)] - f^* \leq \epsilon$ is given by

SVRG / SAGA	GD	SGD
$(n+\kappa)\log\frac{1}{\epsilon}$	$n\kappa\log\frac{1}{\epsilon}$	κ^2/ϵ

	SAGA	SAG	SVRG
Low Storage Cost	✗	✗	✓
Simple(-ish) Proof	✓	✗	✓

Figure: Summary of other relevant properties.

Variance reduction + momentum/acceleration

Katyusha

Strongly convex problems: number gradient calls to compute $\mathbb{E}[f(x_k)] - f^* \leq \epsilon$ is given by

Katyusha	SVRG / SAGA	GD	NAG	SGD
$(n + \sqrt{n\kappa}) \log \frac{1}{\epsilon}$	$(n + \kappa) \log \frac{1}{\epsilon}$	$n\kappa \log \frac{1}{\epsilon}$	$n\sqrt{\kappa} \log \frac{1}{\epsilon}$	κ^2/ϵ

- Improvement critical for **ill conditioned** ($\kappa \gg n$) problems.

Katyusha in the non-strongly convex setting

just convex problems: number gradient calls to compute $\mathbb{E}[f(x_k)] - f^* \leq \epsilon$ is given by

lower bound	Katyusha	SVRG	GD	NAG	SGD
$n + \sqrt{\frac{nL}{\epsilon}}$	$n \log \frac{1}{\epsilon} + \sqrt{\frac{nL}{\epsilon}}$	$n + \sqrt{n}\frac{L}{\epsilon}$	$n\frac{L}{\epsilon}$	$n\sqrt{\frac{L}{\epsilon}}$	$\frac{1}{\epsilon^2}$

Almost matches lower bound.

Summary

- Variance reduction recovers the rates of batch (deterministic methods)
- but (more or less) keeps number of gradient calls of SGD
- still require batch gradient computations sometimes
- requires offline setting (multiple passes through data)
- requires knowledge of (multiple) parameters to get good stepsize

Check out this fantastic review paper: https://ieeexplore-ieee-org.uaccess.univie.ac.at/stamp/stamp.jsp?tp=arnumber=9226504