Optimal Mini-Batch and Step Sizes for SAGA

Nidham Gazagnadou[1,a]

joint work with Robert M. Gower[1] & Joseph Salmon[2]

[1]Télécom Paris, Institut Polytechnique de Paris, Paris, France
[2]IMAG, Université de Montpellier, CNRS, Montpellier, France

[a]This work was supported by grants from Région Ile-de-France

Goals of this Work

- **Finite Sum Minimization problem**

$$w^* = \operatorname*{arg\,min}_{w \in \mathbb{R}^d} \left[f(w) := \frac{1}{n} \sum_{i=1}^{n} f_i(w) \right] \qquad (\mathcal{P})$$

Goals of this Work

- **Finite Sum Minimization problem**

$$w^* = \arg\min_{w \in \mathbb{R}^d} \left[f(w) := \frac{1}{n} \sum_{i=1}^{n} f_i(w) \right] \qquad (\mathcal{P})$$

- **Stochastic Gradient Descent (SGD) and practitioners**
  - SGD and variance-reduced variants widely used to solve $(\mathcal{P})$ ...

Goals of this Work

- **Finite Sum Minimization problem**

$$w^* = \arg\min_{w \in \mathbb{R}^d} \left[f(w) := \frac{1}{n} \sum_{i=1}^{n} f_i(w) \right] \qquad (\mathcal{P})$$

- **Stochastic Gradient Descent (SGD) and practitioners**
  - SGD and variance-reduced variants widely used to solve $(\mathcal{P})$ ...
  - ... yet, painful hyper-parameters tuning

Goals of this Work

- **Finite Sum Minimization problem**

$$w^* = \arg\min_{w \in \mathbb{R}^d} \left[f(w) := \frac{1}{n} \sum_{i=1}^{n} f_i(w) \right] \qquad (\mathcal{P})$$

- **Stochastic Gradient Descent (SGD) and practitioners**
  - SGD and variance-reduced variants widely used to solve $(\mathcal{P})$ ...
  - ... yet, painful hyper-parameters tuning
- **This presentation**

$\rightarrow$ Provide theoretical optimal step and mini-batch sizes for SAGA algorithm

Expected Smoothness Constant

Supervised Learning Optimization Problem

- **Empirical Risk Minimization (ERM)**
  Find optimal parameter/model w^* s.t.

$$w^* = \arg\min_{w \in \mathbb{R}^d} \left[f(w) := \frac{1}{n} \sum_{i=1}^{n} f_i(w) \right] \qquad (\mathcal{P})$$

Supervised Learning Optimization Problem

Loss function of the i–th data sample

- **Empirical Risk Minimization (ERM)**
  Find optimal parameter/model w^* s.t.

$$w^* = \arg\min_{w \in \mathbb{R}^d} \left[f(w) := \frac{1}{n} \sum_{i=1}^{n} f_i(w) \right] \qquad (\mathcal{P})$$

Supervised Learning Optimization Problem

Loss function of the i–th data sample

- **Empirical Risk Minimization (ERM)**
  Find optimal parameter/model w^* s.t.

$$w^* = \arg\min_{w \in \mathbb{R}^d} \left[f(w) := \frac{1}{n} \sum_{i=1}^{n} f_i(w) \right] \tag{$\mathcal{P}$}$$

  with
  - f is L–smooth and μ–strongly convex
  - f_i is L_i–smooth, $\forall i \in [n] := \{1, \dots, n\}$

Supervised Learning Optimization Problem

Loss function of the i–th data sample

- **Empirical Risk Minimization (ERM)**
  Find optimal parameter/model w^* s.t.

$$w^* = \arg\min_{w \in \mathbb{R}^d} \left[f(w) := \frac{1}{n} \sum_{i=1}^{n} f_i(w) \right] \qquad (\mathcal{P})$$

  with
  - f is L–smooth and μ–strongly convex
  - f_i is L_i–smooth, $\forall i \in [n] := \{1, \ldots, n\}$

- **Includes problems such as**
  - Ridge regression: $f_i(w) = \frac{1}{2}(a_i^\top w - y_i)^2 + \frac{\lambda}{2} \|w\|_2^2$
  - Regularized logistic regression: $f_i(w) = \log(1 + e^{-y_i a_i^\top w}) + \frac{\lambda}{2} \|w\|_2^2$

  where
  - $a_i \in \mathbb{R}^d$: feature vector (input)
  - $y_i \in \mathbb{R}$ or $\{-1, 1\}$: label (output)
  - $\lambda > 0$: ridge/Tikhonov's regularization parameter

Solving the ERM Problem

Given a precision $\epsilon > 0$

Solving the ERM Problem

Given a precision $\epsilon > 0$

- **Gradient Descent (GD)**
  - Step update: $w^{k+1} = w^k - \frac{1}{L} \nabla f(w^k)$
  - Iteration complexity: $\mathcal{O}\left(\frac{L}{\mu} \log(1/\epsilon)\right)$

Solving the ERM Problem

Given a precision $\epsilon > 0$

- **Gradient Descent (GD)**
  - Step update: $w^{k+1} = w^k - \frac{1}{L}\nabla f(w^k)$
  - Iteration complexity: $\mathcal{O}\left(\frac{L}{\mu}\log(1/\epsilon)\right)$
- **SAGA (or SVRG/SARAH)**
  Let $L_{\max} := \max_{i\in[n]} L_i$
  - Step update: $w^{k+1} = w^k - \frac{1}{3(n\mu + L_{\max})}\left[\nabla f_i(w^k) - \boldsymbol{J}^k_{:i} + \frac{1}{n}\boldsymbol{J}^k \mathbb{1}\right]$
  - Iteration complexity: $\mathcal{O}\left(\left(\frac{L_{\max}}{\mu} + n\right)\log(1/\epsilon)\right)$

Solving the ERM Problem

Given a precision $\epsilon > 0$

- **Gradient Descent (GD)**
  - Step update: $w^{k+1} = w^k - \frac{1}{L}\nabla f(w^k)$
  - Iteration complexity: $\mathcal{O}\left(\frac{L}{\mu}\log(1/\epsilon)\right)$
- **SAGA (or SVRG/SARAH)**
  Let $L_{\max} := \max_{i\in[n]} L_i$
  - Step update: $w^{k+1} = w^k - \frac{1}{3(n\mu + L_{\max})}\left[\nabla f_i(w^k) - \boldsymbol{J}^k_{:i} + \frac{1}{n}\boldsymbol{J}^k\mathbb{1}\right]$
  - Iteration complexity: $\mathcal{O}\left(\left(\frac{L_{\max}}{\mu} + n\right)\log(1/\epsilon)\right)$

Previous gradients stored in the Jacobian estimate

Solving the ERM Problem

Given a precision $\epsilon > 0$

- **Gradient Descent (GD)**
  - Step update: $w^{k+1} = w^k - \frac{1}{L}\nabla f(w^k)$
  - Iteration complexity: $\mathcal{O}\left(\frac{L}{\mu}\log(1/\epsilon)\right)$
- **SAGA (or SVRG/SARAH)**
  Let $L_{\max} := \max_{i\in[n]} L_i$
  - Step update: $w^{k+1} = w^k - \frac{1}{3(n\mu + L_{\max})}\left[\nabla f_i(w^k) - \mathbf{J}^k_{:i} + \frac{1}{n}\mathbf{J}^k \mathbb{1}\right]$
  - Iteration complexity: $\mathcal{O}\left(\left(\frac{L_{\max}}{\mu} + n\right)\log(1/\epsilon)\right)$
- Distance between L and $L_{\max}$

$$L \leq L_{\max} \leq nL$$

Previous gradients stored in the Jacobian estimate

Solving the ERM Problem

Given a precision $\epsilon > 0$

- **Gradient Descent (GD)**
  - Step update: $w^{k+1} = w^k - \frac{1}{L}\nabla f(w^k)$
  - Iteration complexity: $\mathcal{O}\left(\frac{L}{\mu}\log(1/\epsilon)\right)$

- **SAGA (or SVRG/SARAH)**
  Let $L_{\max} := \max_{i\in[n]} L_i$
  - Step update: $w^{k+1} = w^k - \frac{1}{3(n\mu + L_{\max})}\left[\nabla f_i(w^k) - \mathbf{J}^k_{:i} + \frac{1}{n}\mathbf{J}^k\mathbb{1}\right]$
  - Iteration complexity: $\mathcal{O}\left(\left(\frac{L_{\max}}{\mu} + n\right)\log(1/\epsilon)\right)$

- Distance between L and $L_{\max}$

$$L \leq L_{\max} \leq nL$$

Previous gradients stored in the Jacobian estimate

When n is big possibly $L \ll L_{\max}$

Solving the ERM Problem

Given a precision $\epsilon > 0$

- **Gradient Descent (GD)**
  - Step update: $w^{k+1} = w^k - \frac{1}{L}\nabla f(w^k)$
  - Iteration complexity: $\mathcal{O}\left(\frac{L}{\mu}\log(1/\epsilon)\right)$
- **SAGA (or SVRG/SARAH)**
  Let $L_{\max} := \max_{i\in[n]} L_i$
  - Step update: $w^{k+1} = w^k - \frac{1}{3(n\mu + L_{\max})}\left[\nabla f_i(w^k) - \boldsymbol{J}^k_{:i} + \frac{1}{n}\boldsymbol{J}^k \mathbb{1}\right]$
  - Iteration complexity: $\mathcal{O}\left(\left(\frac{L_{\max}}{\mu} + n\right)\log(1/\epsilon)\right)$
- Distance between L and $L_{\max}$

$$L \leq L_{\max} \leq nL$$

Previous gradients stored in the Jacobian estimate

When n is big possibly $L \ll L_{\max}$

→ Can we benefit from mini-batching to find an interpolating smoothness s.t. $L \overset{?}{\leq} \mathcal{L} \overset{?}{\leq} L_{\max}$?

Key Constant: Expected Smoothness

Definition (Subsample/batch function)

Let $B \subseteq [n]$ a mini-batch of size $|B| = b$

$$f_B(w) := \frac{1}{b}\sum_{i\in B} f_i(w)$$

and denote L_B be the smallest constant s.t. f_B is L_B–smooth.

Key Constant: Expected Smoothness

Definition (Subsample/batch function)

Let $B \subseteq [n]$ a mini-batch of size $|B| = b$

$$f_B(w) := \frac{1}{b} \sum_{i \in B} f_i(w)$$

and denote L_B be the smallest constant s.t. f_B is L_B–smooth.

Recovering known smoothness constants

- $B = [n] \implies L_B = L$
- $B = \{i\} \implies L_B = L_i$

Key Constant: Expected Smoothness

Definition (Subsample/batch function)

Let $B \subseteq [n]$ a mini-batch of size $|B| = b$

$$f_B(w) := \frac{1}{b} \sum_{i \in B} f_i(w)$$

and denote L_B be the smallest constant s.t. f_B is L_B–smooth.

Recovering known smoothness constants

- $B = [n] \implies L_B = L$
- $B = \{i\} \implies L_B = L_i$

Assumption (Expected Smoothness)

Let $S \subseteq [n]$ be a random set of b points sampled without replacement. There exist $\mathcal{L}(b) > 0$ s.t.

$$\mathbb{E}\left[\|\nabla f_S(w) - \nabla f_S(w^*)\|_2^2\right] \leq 2\mathcal{L}(b)\,(f(w) - f(w^*))$$

Key Constant: Expected Smoothness (cont'd)

Definition ($b-$sampling without replacement)

S (a random set-valued mapping) is a $b-$sampling without replacement if

$$\mathbb{P}\left[S = B\right] = \frac{1}{\binom{n}{b}} \quad \forall B \subset [n] \; : \; |B| = b$$

Key Constant: Expected Smoothness (cont'd)

Definition ($b-$sampling without replacement)

S (a random set-valued mapping) is a $b-$sampling without replacement if

$$\mathbb{P}\left[S = B\right] = \frac{1}{\binom{n}{b}} \quad \forall B \subset [n] \ : \ |B| = b$$

Lemma (Expected smoothness formula)

For $b-$sampling without replacement,

$$\mathcal{L}(b) := \frac{1}{\binom{n-1}{b-1}} \max_{i=1,\ldots,n} \left\{ \sum_{B \subseteq [n] \ : \ |B| = b \ \wedge \ i \in B} L_B \right\}$$

Consequence of f_B being L_B–smooth and convex for a $B \subseteq [n]$

Key Constant: Expected Smoothness (cont'd)

Definition ($b-$sampling without replacement)

S (a random set-valued mapping) is a $b-$sampling without replacement if

$$\mathbb{P}\left[S = B\right] = \frac{1}{\binom{n}{b}} \quad \forall B \subset [n] \ : \ |B| = b$$

Lemma (Expected smoothness formula)

For $b-$sampling without replacement,

$$\mathcal{L}(b) := \frac{1}{\binom{n-1}{b-1}} \max_{i=1,\ldots,n} \left\{ \sum_{B \subseteq [n] \,:\, |B|=b \,\wedge\, i \in B} L_B \right\}$$

Consequence of f_B being L_B–smooth and convex for a $B \subseteq [n]$

Problem: calculating $\mathcal{L}(b)$ is intractable for large n

Extreme Values of the Expected Smoothness

Let S be a $b-$sampling without replacement

$$\mathcal{L}(b) = \frac{1}{\binom{n-1}{b-1}} \max_{i=1,\ldots,n} \left\{ \sum_{B \subseteq [n] \,:\, |B|=b \,\wedge\, i \in B} L_B \right\}$$

Extreme Values of the Expected Smoothness

Let S be a $b-$sampling without replacement

$$\mathcal{L}(1) = \frac{1}{\binom{n-1}{0}} \max_{i=1,\ldots,n} \left\{ \sum_{B \subseteq [n] \,:\, |B|=1 \,\wedge\, i \in B} L_B \right\}$$

- If $b = 1$
  - Recovered algorithm: SAGA

Extreme Values of the Expected Smoothness

Let S be a $b-$sampling without replacement

$$\mathcal{L}(1) = \frac{1}{\binom{n-1}{0}} \max_{i=1,\ldots,n} \left\{ \sum_{B \in \{\{1\},\ldots,\{n\}\} : i \in B} L \right\}$$

- If $b = 1$
  - Recovered algorithm: SAGA
  - $B \in \{\{1\}, \ldots, \{n\}\}$

Extreme Values of the Expected Smoothness

Let S be a $b-$sampling without replacement

$$\mathcal{L}(1) = \max_{i=1,\ldots,n} L_i$$

- If $b = 1$
  - Recovered algorithm: SAGA
  - $B \in \{\{1\}, \ldots, \{n\}\}$

Extreme Values of the Expected Smoothness

Let S be a $b-$sampling without replacement

$$\mathcal{L}(1) = \max_{i=1,\dots,n} L_i$$

- If $b = 1$
  - Recovered algorithm: SAGA
  - $B \in \{\{1\}, \dots, \{n\}\}$

$$\boxed{\mathcal{L}(1) = L_{\mathbf{max}}}$$

Extreme Values of the Expected Smoothness

Let S be a $b-$sampling without replacement

$$\mathcal{L}(n) = \frac{1}{\binom{n-1}{n-1}} \max_{i=1,\ldots,n} \left\{ \sum_{B \subseteq [n] \,:\, |B|=n \,\wedge\, i \in B} L \right\}$$

- If $b = 1$
    - Recovered algorithm: SAGA
    - $B \in \{\{1\}, \ldots, \{n\}\}$

$$\boxed{\mathcal{L}(1) = L_{\mathbf{max}}}$$

- If $b = n$
    - Recovered algorithm: Gradient Descent

Extreme Values of the Expected Smoothness

Let S be a $b-$sampling without replacement

$$\mathcal{L}(n) = \max_{i=1,\dots,n} \left\{ \sum_{B=[n]} L_B \right\}$$

- If $b = 1$
  - Recovered algorithm: SAGA
  - $B \in \{\{1\}, \dots, \{n\}\}$

$$\boxed{\mathcal{L}(1) = L_{\mathbf{max}}}$$

- If $b = n$
  - Recovered algorithm: Gradient Descent
  - $B = [n]$

Extreme Values of the Expected Smoothness

Let S be a $b-$sampling without replacement

$$\mathcal{L}(n) = \max_{i=1,\dots,n} L$$

- If $b = 1$
  - Recovered algorithm: SAGA
  - $B \in \{\{1\}, \dots, \{n\}\}$

$$\boxed{\mathcal{L}(1) = L_{\max}}$$

- If $b = n$
  - Recovered algorithm: Gradient Descent
  - $B = [n]$

Extreme Values of the Expected Smoothness

Let S be a $b-$sampling without replacement

$$\mathcal{L}(n) = L$$

- If $b = 1$
  - Recovered algorithm: SAGA
  - $B \in \{\{1\}, \dots, \{n\}\}$

$$\boxed{\mathcal{L}(1) = L_{\textbf{max}}}$$

- If $b = n$
  - Recovered algorithm: Gradient Descent
  - $B = [n]$

$$\boxed{\mathcal{L}(n) = L}$$

Extreme Values of the Expected Smoothness

Let S be a $b-$sampling without replacement

$$\mathcal{L}(n) = L$$

- If $b = 1$
  - Recovered algorithm: SAGA
  - $B \in \{\{1\}, \ldots, \{n\}\}$

$$\boxed{\mathcal{L}(1) = L_{\max}}$$

- If $b = n$
  - Recovered algorithm: Gradient Descent
  - $B = [n]$

$$\boxed{\mathcal{L}(n) = L}$$

$\rightarrow \mathcal{L}(b)$ interpolates between $L_{\max}$ and L

Optimal Mini-Batch and Step Sizes for SAGA

Mini-Batch SAGA

- **The algorithm**

  For $k = 0, 1, 2, \ldots$

Mini-Batch SAGA

- **The algorithm**

  For $k = 0, 1, 2, \ldots$

  - Sample a mini-batch $B \subset [n]$ s.t. $|B| = b$

Mini-Batch SAGA

- **The algorithm**

  For $k = 0, 1, 2, \ldots$

  - Sample a mini-batch $B \subset [n]$ s.t. $|B| = b$
  - Compute a gradient estimate

$$g(w^k) = \frac{1}{b}\sum_{i\in B}\nabla f_i(w^k) - \frac{1}{b}\sum_{i\in B}\boldsymbol{J}^k_{:i} + \frac{1}{n}\boldsymbol{J}^k\mathbb{1}$$

Mini-Batch SAGA

- **The algorithm**

  For $k = 0, 1, 2, \ldots$

  - Sample a mini-batch $B \subset [n]$ s.t. $|B| = b$
  - Compute a gradient estimate

$$g(w^k) = \frac{1}{b}\sum_{i \in B} \nabla f_i(w^k) - \frac{1}{b}\sum_{i \in B} \boldsymbol{J}^k_{:i} + \frac{1}{n}\boldsymbol{J}^k \mathbb{1}$$

  - Take a step

$$w^{k+1} = w^k - \gamma g(w^k)$$

Mini-Batch SAGA

- **The algorithm**

  For $k = 0, 1, 2, \ldots$

  - Sample a mini-batch $B \subset [n]$ s.t. $|B| = b$
  - Compute a gradient estimate

    $$g(w^k) = \frac{1}{b}\sum_{i \in B} \nabla f_i(w^k) - \frac{1}{b}\sum_{i \in B} \boldsymbol{J}^k_{:i} + \frac{1}{n}\boldsymbol{J}^k \mathbb{1}$$

  - Take a step

    $$w^{k+1} = w^k - \gamma g(w^k)$$

  - Update the **Jacobian estimate** $\boldsymbol{J}^k$

    $$\boldsymbol{J}^k_{:i} = \nabla f_i(w^k), \quad \forall i \in B$$

Mini-Batch SAGA

- **The algorithm**

  For $k = 0, 1, 2, \ldots$

  - Sample a mini-batch $B \subset [n]$ s.t. $|B| = b$
  - Compute a gradient estimate

$$g(w^k) = \frac{1}{b}\sum_{i \in B} \nabla f_i(w^k) - \frac{1}{b}\sum_{i \in B} \boldsymbol{J}^k_{:i} + \frac{1}{n}\boldsymbol{J}^k \mathbb{1}$$

  - Take a step

$$w^{k+1} = w^k - \gamma g(w^k)$$

  - Update the **Jacobian estimate $\boldsymbol{J}^k$**

$$\boldsymbol{J}^k_{:i} = \nabla f_i(w^k), \quad \forall i \in B$$

- **What is the optimal mini-batch size?**

$\rightarrow$ Find the "best" b value

Mini-Batch SAGA Iteration Complexity

Theorem (Convergence of mini-batch SAGA[1])

Consider the iterates w^k of the mini-batch SAGA algorithm. Let the step size be

$$\gamma(b) = \frac{1}{4} \frac{1}{\max\left\{\mathcal{L}(b), \frac{1}{b}\frac{n-b}{n-1}L_{\max} + \frac{\mu}{4}\frac{n}{b}\right\}}$$

Given an $\epsilon > 0$, if $k \geq K_{iter}(b)$ where

$$k \geq K_{iter}(b) := \left\{\frac{4\mathcal{L}(b)}{\mu}, \frac{n}{b} + \frac{n-b}{n-1}\frac{4L_{\max}}{b\mu}\right\} \log\left(\frac{1}{\epsilon}\right) \implies \mathbb{E}\left[\left\|w^k - w^*\right\|^2\right] \leq \epsilon\, C \ .$$

with $C > 0$ a constant[a].

[a] $C := \left\|w^0 - w^*\right\|^2 + \frac{\gamma}{2L_{\max}} \sum_{i\in[n]} \left\|J^0_{:i} - \nabla f(w^*)\right\|^2$

[1] Gower et al (2018), arXiv:1805.02632, "Stochastic Quasi-Gradient Methods: Variance Reduction via Jacobian Sketching"

Methodology

Given a precision $\epsilon > 0$,

- **Optimal mini-batch size**

$$\text{find} \quad b^* \in \arg\min_{b \in [n]} K_{\text{total}}(b) = b \times K_{\text{iter}}(b)$$

[1] Gower et al (2018), arXiv:1805.02632, "Stochastic Quasi-Gradient Methods: Variance Reduction via Jacobian Sketching"

Methodology

#gradients per iteration

Given a precision $\epsilon > 0$,

- **Optimal mini-batch size**

$$\text{find} \quad b^* \in \arg\min_{b \in [n]} K_{\text{total}}(b) = b \times K_{\text{iter}}(b)$$

[1] Gower et al (2018), arXiv:1805.02632, "Stochastic Quasi-Gradient Methods: Variance Reduction via Jacobian Sketching"

Methodology

Given a precision $\epsilon > 0$,

#gradients per iteration

#iterations k to achieve $\mathbb{E}\left[\left\|w^k - w^*\right\|^2\right] \leq \epsilon C$

- **Optimal mini-batch size**

$$\text{find} \quad b^* \in \arg\min_{b \in [n]} K_{\text{total}}(b) = b \times K_{\text{iter}}(b)$$

[1] Gower et al (2018), arXiv:1805.02632, "Stochastic Quasi-Gradient Methods: Variance Reduction via Jacobian Sketching"

Methodology

#gradients per iteration

#iterations k to achieve $\mathbb{E}\left[\left\|w^k - w^*\right\|^2\right] \leq \epsilon C$

Given a precision $\epsilon > 0$,

- **Optimal mini-batch size**

$$\text{find} \quad b^* \in \underset{b \in [n]}{\arg\min}\, K_{\text{total}}(b) = b \times K_{\text{iter}}(b)$$

- **Total complexity**[1]

$$K_{\text{total}}(b) = \max\left\{\frac{4b\mathcal{L}(b)}{\mu}, n + \frac{n-b}{n-1}\frac{4L_{\max}}{\mu}\right\}\log\left(\frac{1}{\epsilon}\right)$$

[1] Gower et al (2018), arXiv:1805.02632, "Stochastic Quasi-Gradient Methods: Variance Reduction via Jacobian Sketching"

Methodology

#gradients per iteration

#iterations k to achieve $\mathbb{E}\left[\left\|w^k - w^*\right\|^2\right] \leq \epsilon C$

Given a precision $\epsilon > 0$,

- **Optimal mini-batch size**

$$\text{find} \quad b^* \in \arg\min_{b \in [n]} K_{\text{total}}(b) = b \times K_{\text{iter}}(b)$$

- **Total complexity**[1]

$$K_{\text{total}}(b) = \max\left\{\frac{4b\mathcal{L}(b)}{\mu}, n + \frac{n-b}{n-1}\frac{4L_{\max}}{\mu}\right\} \log\left(\frac{1}{\epsilon}\right)$$

- **Importance of expected smoothness**
    - $\mathcal{L}(b)$ embodies the complexity
    - Gives larger step sizes γ

[1] Gower et al (2018), arXiv:1805.02632, "Stochastic Quasi-Gradient Methods: Variance Reduction via Jacobian Sketching"

Methodology

#gradients per iteration

#iterations k to achieve $\mathbb{E}\left[\left\|w^k - w^*\right\|^2\right] \leq \epsilon C$

Given a precision $\epsilon > 0$,

- **Optimal mini-batch size**

$$\text{find} \quad b^* \in \underset{b\in[n]}{\arg\min}\, K_{\text{total}}(b) = b \times K_{\text{iter}}(b)$$

- **Total complexity**[1]

$$K_{\text{total}}(b) = \max\left\{\frac{4b\mathcal{L}(b)}{\mu}, n + \frac{n-b}{n-1}\frac{4L_{\max}}{\mu}\right\}\log\left(\frac{1}{\epsilon}\right)$$

- **Importance of expected smoothness**
  - $\mathcal{L}(b)$ embodies the complexity
  - Gives larger step sizes γ

$\rightarrow$ **Need to estimate $\mathcal{L}(b)$**

[1] Gower et al (2018), arXiv:1805.02632, "Stochastic Quasi-Gradient Methods: Variance Reduction via Jacobian Sketching"

First Proven Upper-Bound

Lemma (Simple bound)

If S is a $b-$sampling without replacement,

$$\mathcal{L}(b) \leq \mathcal{L}_{simple}(b) := \frac{1}{b}\frac{n-b}{n-1}L_{\max} + \frac{n}{b}\frac{b-1}{n-1}\bar{L}$$

where $\bar{L} := \frac{1}{n}\sum_{i=1}^{n} L_i$

First Proven Upper-Bound

Lemma (Simple bound)

If S is a $b-$sampling without replacement,

$$\mathcal{L}(b) \leq \mathcal{L}_{simple}(b) := \frac{1}{b}\frac{n-b}{n-1}L_{\max} + \frac{n}{b}\frac{b-1}{n-1}\bar{L}$$

where $\bar{L} := \frac{1}{n}\sum_{i=1}^{n} L_i$

– $\mathcal{L}_{\text{simple}}(b)$ interpolates between $L_{\max}$ and $\bar{L} \geq L$ ...

First Proven Upper-Bound

Lemma (Simple bound)

If S is a $b-$sampling without replacement,

$$\mathcal{L}(b) \leq \mathcal{L}_{simple}(b) := \frac{1}{b}\frac{n-b}{n-1}L_{\max} + \frac{n}{b}\frac{b-1}{n-1}\bar{L}$$

where $\bar{L} := \dfrac{1}{n}\displaystyle\sum_{i=1}^{n} L_i$

- $\mathcal{L}_{\text{simple}}(b)$ interpolates between $L_{\max}$ and $\bar{L} \geq L$ ...
- ... but $\mathcal{L}(b)$ interpolates between $L_{\max}$ and L

First Proven Upper-Bound

Lemma (Simple bound)

If S is a $b-$sampling without replacement,

$$\mathcal{L}(b) \leq \mathcal{L}_{simple}(b) := \frac{1}{b}\frac{n-b}{n-1}L_{\max} + \frac{n}{b}\frac{b-1}{n-1}\bar{L}$$

where $\bar{L} := \frac{1}{n}\sum_{i=1}^{n} L_i$

- $\mathcal{L}_{\text{simple}}(b)$ interpolates between $L_{\max}$ and $\bar{L} \geq L$ ...
- ... but $\mathcal{L}(b)$ interpolates between $L_{\max}$ and L

Problem: $\bar{L}$ and L can be far from each other

Exploiting Matrix Concentration Inequalities

Lemma (Bernstein bound)

If S is a $b-$sampling without replacement,

$$\mathcal{L}(b) \leq \mathcal{L}_{Bernstein}(b) := \frac{1}{b}\left(\frac{n-b}{n-1} + \frac{4}{3}\log(d)\right) L_{\max} + 2\frac{b-1}{b}\frac{n}{n-1} L$$

Exploiting Matrix Concentration Inequalities

Lemma (Bernstein bound)

If S is a $b-$sampling without replacement,

$$\mathcal{L}(b) \leq \mathcal{L}_{Bernstein}(b) := \frac{1}{b}\left(\frac{n-b}{n-1} + \frac{4}{3}\log(d)\right) L_{\max} + 2\frac{b-1}{b}\frac{n}{n-1} L$$

- **Proof idea**

$$\mathcal{L}(b) = \frac{1}{\binom{n-1}{b-1}} \max_{i=1,\ldots,n} \left\{ \sum_{B \subseteq [n] \,:\, |B|=b \,\wedge\, i \in B} L_B \right\}$$
$$\leq [...] + \max_{i=1,\ldots,n} \mathbb{E}\left[\lambda_{\max}\left(\sum_k \mathbf{M}_k^i\right)\right]$$

where $\left(\mathbf{M}_k^i\right)_k$ is a sequence of random matrices

Exploiting Matrix Concentration Inequalities

Lemma (Bernstein bound)

If S is a $b-$sampling without replacement,

$$\mathcal{L}(b) \leq \mathcal{L}_{Bernstein}(b) := \frac{1}{b}\left(\frac{n-b}{n-1} + \frac{4}{3}\log(d)\right) L_{\max} + 2\frac{b-1}{b}\frac{n}{n-1}L$$

- **Proof idea**

$$\mathcal{L}(b) = \frac{1}{\binom{n-1}{b-1}} \max_{i=1,\ldots,n} \left\{ \sum_{B\subseteq[n]\,:\,|B|=b \,\wedge\, i\in B} L_B \right\}$$
$$\leq [...] + \max_{i=1,\ldots,n} \mathbb{E}\left[\lambda_{\max}\left(\sum_k \mathbf{M}_k^i\right)\right]$$

  where $\left(\mathbf{M}_k^i\right)_k$ is a sequence of random matrices

- **Technical detail**

  Adapt Matrix Bernstein Inequality to sampling without replacement[2]

[2]Gross & Nesme (2010), Tropp (2011, 2015)

Exploiting Matrix Concentration Inequalities

Lemma (Bernstein bound)

If S is a $b-$sampling without replacement,

$$\mathcal{L}(b) \leq \mathcal{L}_{Bernstein}(b) := \frac{1}{b}\left(\frac{n-b}{n-1} + \frac{4}{3}\log(d)\right) L_{\max} + 2\frac{b-1}{b}\frac{n}{n-1}L$$

- **Proof idea**

$$\mathcal{L}(b) = \frac{1}{\binom{n-1}{b-1}} \max_{i=1,\dots,n} \left\{ \sum_{B\subseteq[n]\,:\,|B|=b\,\wedge\, i\in B} L_B \right\}$$
$$\leq [\dots] + \max_{i=1,\dots,n} \mathbb{E}\left[\lambda_{\max}\left(\sum_k \mathbf{M}_k^i\right)\right]$$

where $\left(\mathbf{M}_k^i\right)_k$ is a sequence of random matrices

- **Technical detail**

Adapt Matrix Bernstein Inequality to sampling without replacement[2]

Problem: $\mathcal{L}_{\text{Bernstein}}(b)$ approximation interpolates between $\approx \log(d) L_{\max}$ and $2L$

[2] Gross & Nesme (2010), Tropp (2011, 2015)

The "Practical" Estimate

$$\mathcal{L}_{\text{practical}}(b) := \frac{1}{b}\frac{n-b}{n-1}L_{\max} + \frac{n}{b}\frac{b-1}{n-1}L$$

The "Practical" Estimate

$$\mathcal{L}_{\text{practical}}(b) := \frac{1}{b}\frac{n-b}{n-1}L_{\max} + \frac{n}{b}\frac{b-1}{n-1}L$$

L instead of $\bar{L}$ in the simple bound

The "Practical" Estimate

$$\mathcal{L}_{\text{practical}}(b) := \frac{1}{b}\frac{n-b}{n-1}L_{\max} + \frac{n}{b}\frac{b-1}{n-1} L$$

L instead of $\bar{L}$ in the simple bound

– Nicely interpolation between $L_{\max}$ and L ...

The "Practical" Estimate

$$\mathcal{L}_{\text{practical}}(b) := \frac{1}{b}\frac{n-b}{n-1}L_{\max} + \frac{n}{b}\frac{b-1}{n-1}L$$

L instead of $\bar{L}$ in the simple bound

- Nicely interpolation between $L_{\max}$ and $L \ldots$
- ... Yet, not a proven bound: $\mathcal{L}(b) \overset{?}{\leq} \mathcal{L}_{\text{practical}}(b)$

The "Practical" Estimate

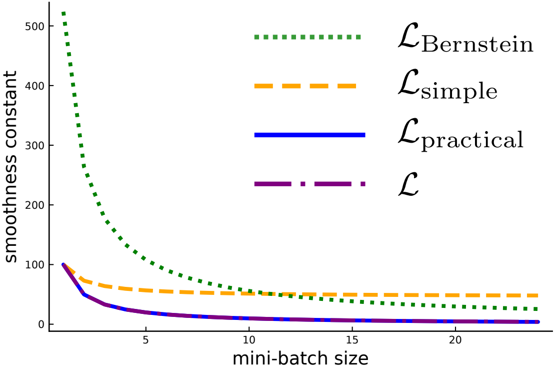
$$\mathcal{L}_{\text{practical}}(b) := \frac{1}{b}\frac{n-b}{n-1}L_{\max} + \frac{n}{b}\frac{b-1}{n-1}L$$

L instead of $\bar{L}$ in the simple bound

– Nicely interpolation between $L_{\max}$ and L ...

– ... Yet, not a proven bound: $\mathcal{L}(b) \overset{?}{\leq} \mathcal{L}_{\text{practical}}(b)$

Upper bounds and $\mathcal{L}(b)$ computed on artificial data ($n = d = 24$)



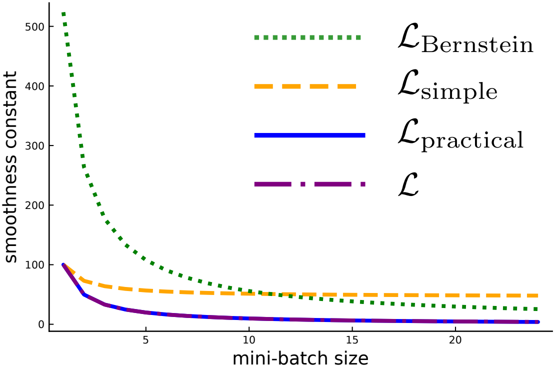
The “Practical” Estimate

$$\mathcal{L}_{\text{practical}}(b) := \frac{1}{b}\frac{n-b}{n-1}L_{\max} + \frac{n}{b}\frac{b-1}{n-1}L$$

L instead of $\bar{L}$ in the simple bound

- Nicely interpolation between $L_{\max}$ and L ...
- ... Yet, not a proven bound: $\mathcal{L}(b) \overset{?}{\leq} \mathcal{L}_{\text{practical}}(b)$

Upper bounds and $\mathcal{L}(b)$ computed on artificial data ($n = d = 24$)



$\rightarrow$ **Numerically** $\mathcal{L}_{\text{practical}}(b) \approx \mathcal{L}(b)$

Optimal Mini-Batch from the "Practical" Estimate

Given a precision $\epsilon > 0$,

- **Total complexity bound**
  Since $\mathcal{L}(b) \leq \mathcal{L}_{\text{practical}}(b)$,

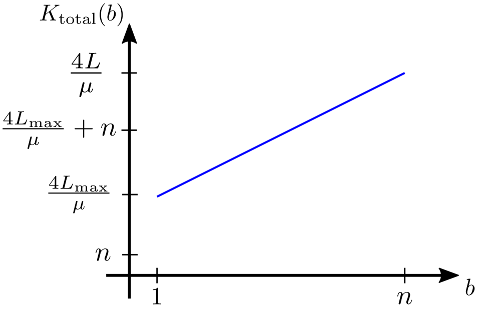
$$K_{\text{total}}(b) \leq \max\left\{ n\frac{b-1}{n-1}\frac{4L}{\mu} + \frac{n-b}{n-1}\frac{4L_{\max}}{\mu},\ n + \frac{n-b}{n-1}\frac{4L_{\max}}{\mu} \right\} \log\left(\frac{1}{\epsilon}\right)$$

Optimal Mini-Batch from the "Practical" Estimate

Given a precision $\epsilon > 0$,

- **Total complexity bound**
  Since $\mathcal{L}(b) \leq \mathcal{L}_{\text{practical}}(b)$,

$$K_{\text{total}}(b) \leq \max\left\{ \underbrace{n\frac{b-1}{n-1}\frac{4L}{\mu} + \frac{n-b}{n-1}\frac{4L_{\max}}{\mu}}_{\text{linearly increasing}}, n + \frac{n-b}{n-1}\frac{4L_{\max}}{\mu} \right\} \log\left(\frac{1}{\epsilon}\right)$$

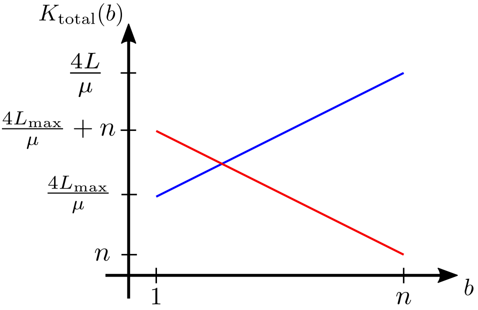


Optimal Mini-Batch from the "Practical" Estimate

Given a precision $\epsilon > 0$,

- **Total complexity bound**
  Since $\mathcal{L}(b) \leq \mathcal{L}_{\text{practical}}(b)$,

$$K_{\text{total}}(b) \leq \max \left\{ \underbrace{n\frac{b-1}{n-1}\frac{4L}{\mu} + \frac{n-b}{n-1}\frac{4L_{\max}}{\mu}}_{\text{linearly increasing}}, \underbrace{n + \frac{n-b}{n-1}\frac{4L_{\max}}{\mu}}_{\text{linearly decreasing}} \right\} \log\left(\frac{1}{\epsilon}\right)$$



Optimal Mini-Batch from the "Practical" Estimate

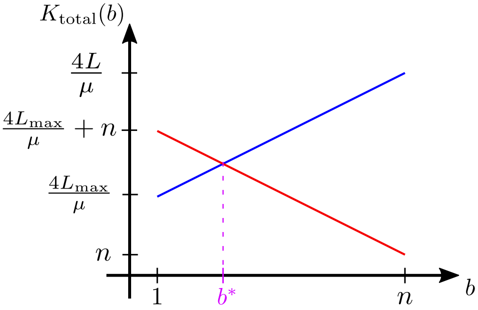
Given a precision $\epsilon > 0$,

- **Total complexity bound**
  Since $\mathcal{L}(b) \leq \mathcal{L}_{\text{practical}}(b)$,

$$K_{\text{total}}(b) \leq \max\left\{ \underbrace{n\frac{b-1}{n-1}\frac{4L}{\mu} + \frac{n-b}{n-1}\frac{4L_{\max}}{\mu}}_{\text{linearly increasing}}, \underbrace{n + \frac{n-b}{n-1}\frac{4L_{\max}}{\mu}}_{\text{linearly decreasing}} \right\} \log\left(\frac{1}{\epsilon}\right)$$

Optimal mini-batch size

$$\Longrightarrow \boxed{b_{\text{practical}} = \left\lfloor 1 + \frac{\mu(n-1)}{4L} \right\rfloor}$$



Mini-Batch SAGA Step Sizes

- **Link between the step size and the expected smoothness**

$$\gamma(b) = \frac{1}{4}\frac{1}{\max\left\{\mathcal{L}(b), \frac{1}{b}\frac{n-b}{n-1}L_{\max} + \frac{\mu}{4}\frac{n}{b}\right\}}$$

Mini-Batch SAGA Step Sizes

- **Link between the step size and the expected smoothness**

$$\gamma(b) = \frac{1}{4}\frac{1}{\max\left\{\mathcal{L}(b), \frac{1}{b}\frac{n-b}{n-1}L_{\max} + \frac{\mu}{4}\frac{n}{b}\right\}}$$

- The smaller $\mathcal{L}(b)$ (the smoother f_B), the larger $\gamma(b)$

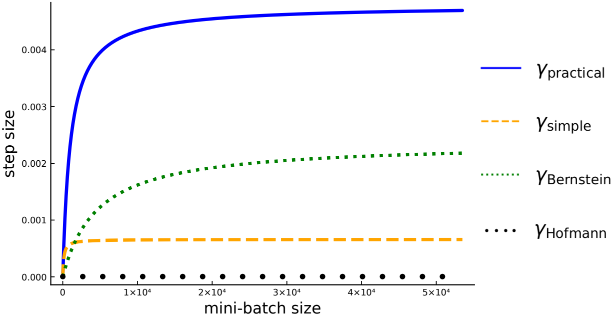
Mini-Batch SAGA Step Sizes

- **Link between the step size and the expected smoothness**

$$\gamma(b) = \frac{1}{4} \frac{1}{\max\left\{\mathcal{L}(b), \frac{1}{b}\frac{n-b}{n-1}L_{\max} + \frac{\mu}{4}\frac{n}{b}\right\}}$$

- The smaller $\mathcal{L}(b)$ (the smoother f_B), the larger $\gamma(b)$
- Plugging $\mathcal{L}_{\text{practical}}(b)$, $\mathcal{L}_{\text{simple}}(b)$ or $\mathcal{L}_{\text{Bernstein}}(b)$ into $\gamma(b)$

Step size increasing with mini-batch size on *slice* data set ($n = 53,500, d = 384$)



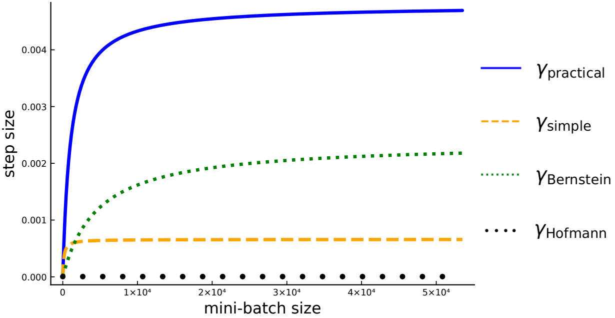
Mini-Batch SAGA Step Sizes

- **Link between the step size and the expected smoothness**

$$\gamma(b) = \frac{1}{4}\frac{1}{\max\left\{\mathcal{L}(b), \frac{1}{b}\frac{n-b}{n-1}L_{\max} + \frac{\mu}{4}\frac{n}{b}\right\}}$$

- The smaller $\mathcal{L}(b)$ (the smoother f_B), the larger $\gamma(b)$
- Plugging $\mathcal{L}_{\text{practical}}(b)$, $\mathcal{L}_{\text{simple}}(b)$ or $\mathcal{L}_{\text{Bernstein}}(b)$ into $\gamma(b)$

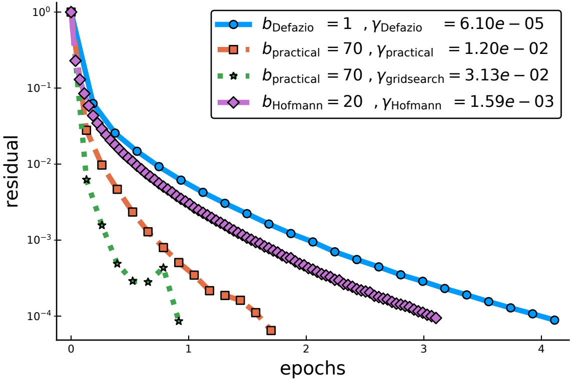
Step size increasing with mini-batch size on *slice* data set ($n = 53,500, d = 384$)



$\rightarrow$ Straightforward and larger step size for large b

Numerical Experiments

Convergence Results on Real Data

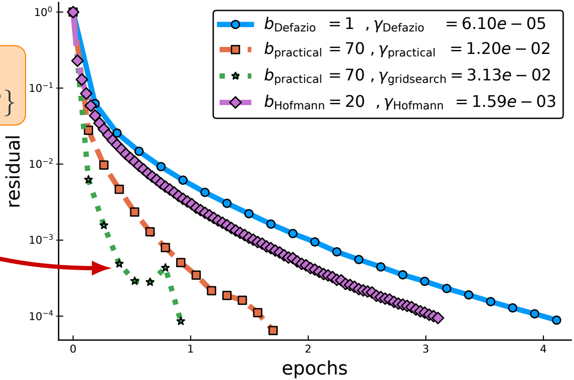


Comparison of SAGA settings for the *slice*[3] data set
$(n = 53,500, d = 384)$

[3]UCI Machine Learning Repository

Convergence Results on Real Data

Grid search over
$\gamma \in \{2^{-33}, 2^{-31} \ldots, 2^{23}, 2^{25}\}$

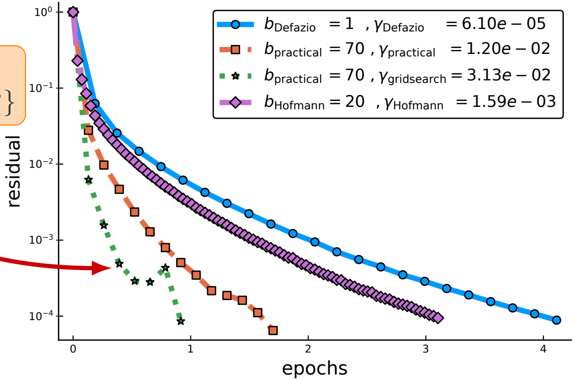


Comparison of SAGA settings for the *slice*[3] data set
$(n = 53,500, d = 384)$

[3]UCI Machine Learning Repository

Convergence Results on Real Data

Grid search over
$\gamma \in \{2^{-33}, 2^{-31} \ldots, 2^{23}, 2^{25}\}$



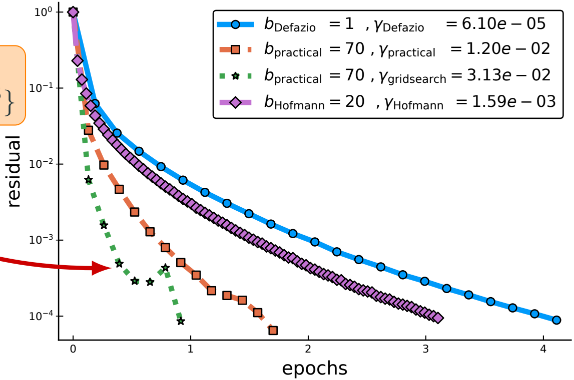
Comparison of SAGA settings for the *slice*[3] data set
$(n = 53,500, d = 384)$

→ **Larger mini-batch and step sizes: faster convergence**

[3] UCI Machine Learning Repository

Convergence Results on Real Data

Grid search over
$\gamma \in \{2^{-33}, 2^{-31}\ldots, 2^{23}, 2^{25}\}$



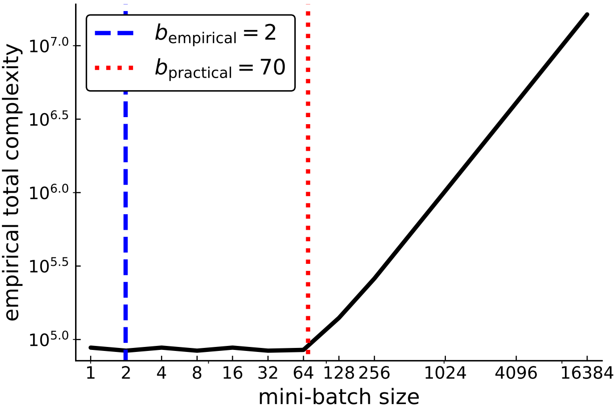
Comparison of SAGA settings for the *slice*[3] data set
$(n = 53,500, d = 384)$

→ Larger mini-batch and step sizes: faster convergence

→ Competing against grid-search!

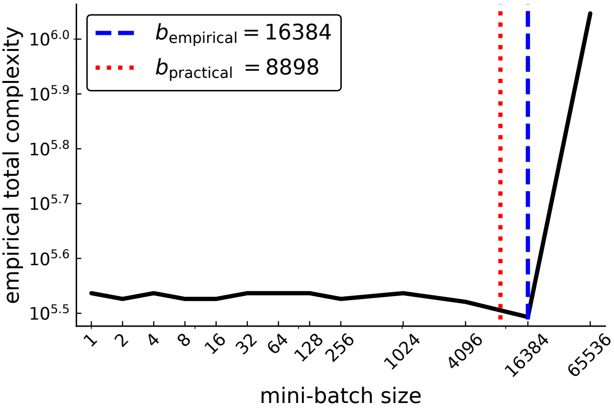
[3]UCI Machine Learning Repository

Optimality of Our Mini-Batch Size



Complexity explosion for the *slice* data set $(n = 53,500, d = 384)$

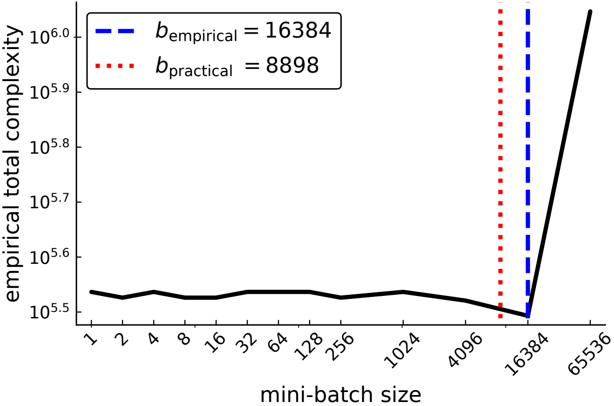
Optimality of Our Mini-Batch Size (cont'd)



Complexity explosion for the *real-sim*[4] data set ($n = 72,309, d = 20,958$)

[4]LIBSVM Data

Optimality of Our Mini-Batch Size (cont'd)



Complexity explosion for the *real-sim*[4] data set ($n = 72,309, d = 20,958$)

$\rightarrow$ Regime change observed & predicting the largest mini-batch size before complexity explosion

[4]LIBSVM Data

Conclusion

Summary of Our Contributions

What was done for SAGA

What has been done since then

[4] Gower et. al (2019), ICML, "SGD: General Analysis and Improved Rates"
[5] LIBSVM and UCI repositories
[6] Sebbouh et. al (2019), arXiv:1908.02725, "Towards closing the gap between the theory and practice of SVRG"

Summary of Our Contributions

What was done for SAGA

- Estimates of the expected smoothness

What has been done since then

[4]Gower et. al (2019), ICML, "SGD: General Analysis and Improved Rates"
[5]LIBSVM and UCI repositories
[6]Sebbouh et. al (2019), arXiv:1908.02725, "Towards closing the gap between the theory and practice of SVRG"

Summary of Our Contributions

Turned out that $\mathcal{L}_{\text{practical}}(b)$ is an actual upper-bound![4]

What was done for SAGA

- Estimates of the expected smoothness

What has been done since then

[4]Gower et. al (2019), ICML, "SGD: General Analysis and Improved Rates"

[5]LIBSVM and UCI repositories

[6]Sebbouh et. al (2019), arXiv:1908.02725, "Towards closing the gap between the theory and practice of SVRG"

Summary of Our Contributions

Turned out that $\mathcal{L}_{\text{practical}}(b)$ is an actual upper-bound![4]

What was done for SAGA

- Estimates of the expected smoothness
- Simple formula for the step size $\gamma(b)$

What has been done since then

[4]Gower et. al (2019), ICML, "SGD: General Analysis and Improved Rates"
[5]LIBSVM and UCI repositories
[6]Sebbouh et. al (2019), arXiv:1908.02725, "Towards closing the gap between the theory and practice of SVRG"

Summary of Our Contributions

Turned out that $\mathcal{L}_{\text{practical}}(b)$ is an actual upper-bound![4]

What was done for SAGA

- Estimates of the expected smoothness
- Simple formula for the step size $\gamma(b)$
- Optimal mini-batch size $b_{\text{practical}}$

What has been done since then

[4]Gower et. al (2019), ICML, "SGD: General Analysis and Improved Rates"
[5]LIBSVM and UCI repositories
[6]Sebbouh et. al (2019), arXiv:1908.02725, "Towards closing the gap between the theory and practice of SVRG"

Summary of Our Contributions

Turned out that $\mathcal{L}_{\text{practical}}(b)$ is an actual upper-bound![4]

What was done for SAGA

- Estimates of the expected smoothness
- Simple formula for the step size $\gamma(b)$
- Optimal mini-batch size $b_{\text{practical}}$
- Convincing numerics verifying the optimality of our parameters on real data sets[5]

Julia code available at
https://github.com/gowerrobert/StochOpt.jl/

What has been done since then

[4] Gower et. al (2019), ICML, "SGD: General Analysis and Improved Rates"
[5] LIBSVM and UCI repositories
[6] Sebbouh et. al (2019), arXiv:1908.02725, "Towards closing the gap between the theory and practice of SVRG"

Summary of Our Contributions

Turned out that $\mathcal{L}_{\text{practical}}(b)$ is an actual upper-bound![4]

What was done for SAGA

- Estimates of the expected smoothness
- Simple formula for the step size $\gamma(b)$
- Optimal mini-batch size $b_{\text{practical}}$
- Convincing numerics verifying the optimality of our parameters on real data sets[5]

Julia code available at
https://github.com/gowerrobert/StochOpt.jl/

What has been done since then

- Extended study to variants of SVRG[6]

---

[4] Gower et. al (2019), ICML, "SGD: General Analysis and Improved Rates"
[5] LIBSVM and UCI repositories
[6] Sebbouh et. al (2019), arXiv:1908.02725, "Towards closing the gap between the theory and practice of SVRG"

References (1/2)

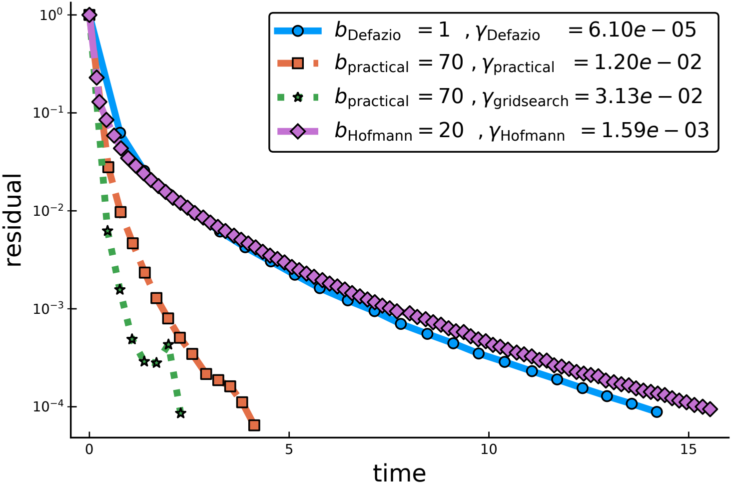
- Defazio, Bach and Lacoste-Julien (2014), NIPS
  “SAGA: A Fast Incremental Gradient Method With Support for Non-Strongly Convex Composite Objectives”
- Gower, Richtárik and Bach (2018), preprint online arXiv:1805.02632
  “Stochastic Quasi-Gradient Methods: Variance Reduction via Jacobian Sketching”
- Gower, Loizou, Qian, Sailanbayev, Shulgin, Richtárik (2019), ICML
  “SGD: General Analysis and Improved Rates”
- Sebbouh, Gazagnadou, Jelassi, Bach, Gower (2019), preprint online arXiv:1908.02725
  “Towards closing the gap between the theory and practice of SVRG”
- Robbins and Monro (1951), Annals of Mathematical Statistics
  “A stochastic approximation method”
- Schmidt, Le Roux and Bach (2017), Mathematical Programming
  “Minimizing finite sums with the stochastic average gradient”
- Johnson and Zhang (2013), NIPS
  “Accelerating Stochastic Gradient Descent using Predictive Variance Reduction”

References (2/2)

- Tropp (2015), Foundations and Trends® in Machine Learning
  "An Introduction to Matrix Concentration Inequalities"
- Tropp (2012), Foundations of Computational Mathematics
  "User-Friendly Tail Bounds for Sums of Random Matrices"
- Tropp (2011), Advances in Adaptive Data Analysis
  "Improved analysis of the subsampled randomized Hadamard transform"
- Bach (2013), COLT
  "Sharp analysis of low-rank kernel matrix approximations"
- Gross and Nesme (2010), preprint online arXiv:1001.2738
  "Note on sampling without replacing from a finite collection of matrices"
- Hoeffding (1963), Journal of the American Statistical Association
  "Probability inequalities for sums of bounded random variables"
- Chang and Lin (2011), ACM TIST
  "LIBSVM : A library for support vector machines"

Questions ?

Time Plot Convergence Results on Real Data



Comparison of SAGA settings for the *slice*[5] data set
$(n = 53,500, d = 384)$

[5] UCI Machine Learning Repository

JacSketch[7] Lyapunov Function

Definition (Stochastic Lyapunov function)

$$\Psi^k := \left\| w^k - w^* \right\|_2^2 + \frac{\gamma}{2bL_{\max}} \left\| \mathbf{J}^k - \nabla \mathbf{F}(w^*) \right\|_{\mathrm{F}}^2$$

- $\left\| \cdot \right\|_{\mathrm{F}}$: Frobenius norm
- $\nabla \mathbf{F}(w) = [\nabla f_1(w), \ldots, \nabla f_n(w)] \in \mathbb{R}^{d \times n}$: Jacobian matrix
- $\left\{ w^k, \mathbf{J}^k \right\}_{k \geq 0}$ are the points and Jacobian estimate

If $\epsilon > 0$ denotes the desired precision, Theorem 3.6 ensures that, for a step size $\gamma = \min \left\{ \frac{1}{4\mathcal{L}}, \frac{1}{\frac{1}{b}\frac{n-b}{n-1} L_{\max} + \frac{\mu}{4}\frac{n}{b}} \right\}$,

$$\mathbb{E}\left[\Psi^k\right] \leq \epsilon \Psi^0 \ .$$

[7] Gower et al (2018), arXiv:1805.02632, “Stochastic Quasi-Gradient Methods: Variance Reduction via Jacobian Sketching”

Proof of the "Practical" Bound (1/2)

Lemma (Practical bound)

If S is a $b-$sampling without replacement,

$$\mathcal{L}(b) \leq \mathcal{L}_{practical}(b) := \frac{1}{b}\frac{n-b}{n-1}L_{\max} + \frac{n}{b}\frac{b-1}{n-1}L$$

Proof: See Proposition 3.8 in Gower et. al (2019), "SGD: General Analysis and Improved Rates",
Let S be a b–sampling without replacement with $b \in [n]$, and $x, z \in \mathbb{R}^d$. Let us denote

$$p_B := \mathbb{P}\left[S = B\right] = \frac{1}{\binom{n}{b}} \quad , \quad p_i := \mathbb{P}\left[i \in S\right] = \frac{b}{n} \quad , \quad P_{ij} := \mathbb{P}\left[i, j \in S\right] = \begin{cases} \frac{b(b-1)}{n(n-1)} & \text{if } i \neq j \\ p_i = \frac{n}{b} & \text{else} \end{cases} ,$$

for all $B \subseteq [n]$ and all $i, j \in [n]$.

$$\|\nabla f_S(w) - \nabla f_S(z)\|_2^2 = \frac{1}{n^2}\left\|\sum_{i \in S}\frac{1}{p_i}\left(\nabla f_i(w) - \nabla f_i(z)\right)\right\|_2^2$$

$$= \sum_{i,j \in S}\left\langle \frac{1}{np_i}\left(\nabla f_i(w) - \nabla f_i(z)\right), \frac{1}{np_j}\left(\nabla f_j(w) - \nabla f_j(z)\right)\right\rangle$$

Then, we take expectation over all possible mini-batches $(B \subseteq [n] : |B| = b)$.

$$\mathbb{E}\left[\|\nabla f_S(w) - \nabla f_S(z)\|_2^2\right] = \sum_B p_B \sum_{i,j \in B}\left\langle \frac{1}{np_i}\left(\nabla f_i(w) - \nabla f_i(z)\right), \frac{1}{np_j}\left(\nabla f_j(w) - \nabla f_j(z)\right)\right\rangle$$

$$\overset{\text{double counting}}{=} \sum_{i,j=1}^{n}\sum_{B \,:\, i,j \in B} p_B \left\langle \frac{1}{np_i}\left(\nabla f_i(w) - \nabla f_i(z)\right), \frac{1}{np_j}\left(\nabla f_j(w) - \nabla f_j(z)\right)\right\rangle$$

$$\overset{\sum_{B \,:\, i,j \in B} p_B = P_{ij}}{=} \sum_{i,j=1}^{n}\left\langle \frac{1}{np_i}\left(\nabla f_i(w) - \nabla f_i(z)\right), \frac{1}{np_j}\left(\nabla f_j(w) - \nabla f_j(z)\right)\right\rangle P_{ij}$$

Proof of the "Practical" Bound (2/2)

Now consider the two disjoint cases where $i \neq j$ and $i = j$.

$$\mathbb{E}\left[\|\nabla f_S(w) - \nabla f_S(z)\|_2^2\right] = \sum_{i,j=1}^{n} \frac{P_{ij}}{p_i p_j} \left\langle \frac{1}{n}(\nabla f_i(w) - \nabla f_i(z)), \frac{1}{n}(\nabla f_j(w) - \nabla f_j(z)) \right\rangle$$

$$= \sum_{i,j\,:\,i\neq j} \frac{n(b-1)}{b(n-1)} \left\langle \frac{1}{n}(\nabla f_i(w) - \nabla f_i(z)), \frac{1}{n}(\nabla f_j(w) - \nabla f_j(z)) \right\rangle + \sum_{i=1}^{n} \frac{1}{n^2}\frac{1}{p_i} \|\nabla f_i(w) - \nabla f_i(z)\|_2^2$$

$$= \sum_{i,j=1}^{n} \frac{n(b-1)}{b(n-1)} \left\langle \frac{1}{n}(\nabla f_i(w) - \nabla f_i(z)), \frac{1}{n}(\nabla f_j(w) - \nabla f_j(z)) \right\rangle$$

$$+ \sum_{i=1}^{n} \frac{1}{n^2}\frac{1}{p_i} \left(1 - \frac{n(b-1)}{b(n-1)} p_i\right) \|\nabla f_i(w) - \nabla f_i(z)\|_2^2$$

$$\overset{\substack{f_i \text{ smooth} \\ \& \text{ convex}}}{\leq} \frac{n(b-1)}{b(n-1)} \|\nabla f(w) - \nabla f(z)\|_2^2 + 2\sum_{i=1}^{n} \frac{L_i}{n^2 p_i} \left(1 - \frac{n(b-1)}{b(n-1)} p_i\right) (f_i(w) - f_i(z) - \langle \nabla f_i(z), w - z\rangle)$$

$$\overset{\substack{f \text{ smooth} \\ \& \text{ convex}}}{\leq} 2\left(\frac{n(b-1)}{b(n-1)} L + \max_{i=1,\ldots,n} \frac{L_i}{n p_i}\left(1 - \frac{n(b-1)}{b(n-1)} p_i\right)\right)(f(w) - f(z) - \langle \nabla f(z), w - z\rangle)$$

$$= 2\left(\frac{n(b-1)}{b(n-1)} L + \max_{i=1,\ldots,n} \frac{L_i}{b}\left(1 - \frac{b-1}{n-1}\right)\right)(f(w) - f(z) - \langle \nabla f(z), w - z\rangle)$$

$$= 2\underbrace{\left(\frac{1}{b}\frac{n-b}{n-1} L_{\max} + \frac{n}{b}\frac{b-1}{n-1} L\right)}_{\mathcal{L}_{\text{practical}(b)}}(f(w) - f(z) - \langle \nabla f(z), w - z\rangle)$$

$\square$

From Sampling Without to With Replacement

Lemma (Domination of the trace of the mgf of a sample without replacement)

Consider two finite sequences, of same length, $\{\mathbf{X}_k\}$ and $\{\mathbf{M}_k\}$ of Hermitian random matrices of same size sampled respectively with and without replacement from a finite set $\mathcal{X}$. Let $\theta \in \mathbb{R}$, then

$$\mathbb{E}\,\mathrm{tr}\exp\left(\theta \sum\nolimits_k \mathbf{M}_k\right) \le \mathbb{E}\,\mathrm{tr}\exp\left(\theta \sum\nolimits_k \mathbf{X}_k\right) \quad .$$

See Gross and Nesme (2010), arXiv:1001.2738,
"Note on sampling without replacing from a finite collection of matrices"

Proof Sketch of the Bernstein Bound (1/2)

(i) Write $\mathcal{L}$ as an **expectation**

$$\begin{aligned}
\mathcal{L} &= \max_{i=1,\ldots,n} \mathbb{E}\left[L_{S^i \cup \{i\}}\right] \\
&= \max_{i=1,\ldots,n} U\mathbb{E}\left[\lambda_{\max}\left(\frac{1}{b}\sum_{j\in S^n\cup\{i\}} a_j a_j^\top\right)\right] \\
&\le \frac{1}{b}\frac{n-b}{n-1} L_{\max} + \frac{n}{b}\frac{b-1}{n-1} L \\
&\quad + \max_{i=1,\ldots,n} U\mathbb{E}\left[\lambda_{\max}\left(\underbrace{\frac{1}{b}\sum_{j\in S^i} a_j a_j^\top - \frac{1}{b}\frac{b-1}{n-1}\sum_{j\in[n]\setminus\{i\}} a_j a_j^\top}_{\mathbf{N}=\sum_k \mathbf{M}_k}\right)\right]
\end{aligned}$$

Proof Sketch of the Bernstein Bound (1/2)

(i) Write $\mathcal{L}$ as an **expectation**

$$
\begin{aligned}
\mathcal{L} &= \max_{i=1,\ldots,n} \mathbb{E}\left[L_{S^i \cup \{i\}}\right] \\
&= \max_{i=1,\ldots,n} U\mathbb{E}\left[\lambda_{\max}\left(\frac{1}{b}\sum_{j \in S^n \cup \{i\}} a_j a_j^\top\right)\right] \\
&\leq \frac{1}{b}\frac{n-b}{n-1} L_{\max} + \frac{n}{b}\frac{b-1}{n-1} L \\
&\quad + \max_{i=1,\ldots,n} U\mathbb{E}\left[\lambda_{\max}\left(\underbrace{\frac{1}{b}\sum_{j \in S^i} a_j a_j^\top - \frac{1}{b}\frac{b-1}{n-1}\sum_{j \in [n]\setminus\{i\}} a_j a_j^\top}_{\mathbf{N}=\sum_k \mathbf{M}_k}\right)\right]
\end{aligned}
$$

Practical approximation

Proof Sketch of the Bernstein Bound (2/2)

(ii) Write $\mathbf{N}$ as a sum of random matrices and apply

Theorem (Matrix Bernstein Inequality Without Replacement)

Let $\mathcal{X}$ be a finite set of Hermitian matrices with dimension d s.t.

$$\lambda_{\max}(\mathbf{X}) \leq L \quad \forall \mathbf{X} \in \mathcal{X} \ .$$

Sample $\{\mathbf{X}_k\}$ and $\{\mathbf{M}_k\}$ uniformly at random from $\mathcal{X}$ resp. with and without replacement s.t.

$$\mathbb{E}\,\mathbf{X}_k = 0 \quad \forall k \ .$$

Let $\mathbf{Y} := \sum_k \mathbf{X}_k$ and $\mathbf{N} := \sum_k \mathbf{M}_k$. Then

$$\mathbb{E}\,\lambda_{\max}(\mathbf{N}) \leq \sqrt{2v(\mathbf{Y})\log d} + \frac{1}{3} L \log d \ .$$

where $\quad v(\mathbf{Y}) := \left\|\mathbb{E}\,\mathbf{Y}^2\right\| = \left\|\sum_k \mathbb{E}\,\mathbf{X}_k^2\right\| = \lambda_{\max}\left(\sum_k \mathbb{E}\,\mathbf{X}_k^2\right).$

(Tropp, 2011, 2012, 2015; Gross and Nesme, 2010; Bach 2013)

Stochastic Reformulation of the ERM

- **Sampling vector**
  Let $v \in \mathbb{R}^n$, with distribution $\mathcal{D}$ s.t.

$$\mathbb{E}_{\mathcal{D}}[v] = \mathbb{1}$$

  where $\mathbb{1}$ is the all-ones vector

Stochastic Reformulation of the ERM

- **Sampling vector**
  Let $v \in \mathbb{R}^n$, with distribution $\mathcal{D}$ s.t.
  $$\mathbb{E}_{\mathcal{D}}[v] = \mathbb{1}$$
  where $\mathbb{1}$ is the all-ones vector
- **Unbiased subsampled function**
  $$f_v(w) := \frac{1}{n}\sum_{i=1}^{n} f_i(w) v_i = \frac{1}{n}\langle F(w), v\rangle$$
  where $F(w) := (f_1(w), \ldots, f_n(w))^\top \in \mathbb{R}^n$

Stochastic Reformulation of the ERM

- **Sampling vector**
  Let $v \in \mathbb{R}^n$, with distribution $\mathcal{D}$ s.t.
  $$\mathbb{E}_{\mathcal{D}}[v] = \mathbb{1}$$
  where $\mathbb{1}$ is the all-ones vector
- **Unbiased subsampled function**
  $$f_v(w) := \frac{1}{n}\sum_{i=1}^{n} f_i(w) v_i = \frac{1}{n}\langle F(w), v\rangle$$
  where $F(w) := (f_1(w), \ldots, f_n(w))^\top \in \mathbb{R}^n$
  $$\implies \mathbb{E}_{\mathcal{D}}[f_v(w)] = \frac{1}{n}\langle F(w), \mathbb{E}_{\mathcal{D}}[v]\rangle = f(w)$$

Stochastic Reformulation of the ERM

- **Sampling vector**
  Let $v \in \mathbb{R}^n$, with distribution $\mathcal{D}$ s.t.

$$\mathbb{E}_{\mathcal{D}}[v] = \mathbb{1}$$

  where $\mathbb{1}$ is the all-ones vector
- **Unbiased subsampled function**

$$f_v(w) := \frac{1}{n}\sum_{i=1}^{n} f_i(w) v_i = \frac{1}{n}\langle F(w), v\rangle$$

  where $F(w) := (f_1(w), \ldots, f_n(w))^\top \in \mathbb{R}^n$

$$\implies \mathbb{E}_{\mathcal{D}}[f_v(w)] = \frac{1}{n}\langle F(w), \mathbb{E}_{\mathcal{D}}[v]\rangle = f(w)$$

- **ERM reformulation**

$$\text{solving ERM} \iff \text{find } w^* \in \arg\min_{w \in \mathbb{R}^d} \frac{1}{n}\sum_{i=1}^{n} f_i(w) = \mathbb{E}_{\mathcal{D}}[f_v(w)]$$

Arbitrary Sampling

Examples of sampling vector

Let $\nabla \mathbf{F}(w) := [\nabla f_1(w), \ldots, \nabla f_n(w)] \in \mathbb{R}^{d \times n}$ be the Jacobian

Arbitrary Sampling

Examples of sampling vector

Let $\nabla \mathbf{F}(w) := [\nabla f_1(w), \ldots, \nabla f_n(w)] \in \mathbb{R}^{d \times n}$ be the Jacobian

- Let $\boldsymbol{v} = \mathbb{1}$ (deterministic)

$$\nabla f_v(w) = \frac{1}{n} \nabla \mathbf{F}(w) \mathbb{1} = \nabla f(w) \tag{GD}$$

Arbitrary Sampling

Examples of sampling vector

Let $\nabla \mathbf{F}(w) := [\nabla f_1(w), \ldots, \nabla f_n(w)] \in \mathbb{R}^{d \times n}$ be the Jacobian

- Let $\boldsymbol{v} = \mathbb{1}$ (deterministic)

$$\nabla f_v(w) = \frac{1}{n} \nabla \mathbf{F}(w) \mathbb{1} = \nabla f(w) \tag{GD}$$

- Let $\boldsymbol{v} = \boldsymbol{n e_i}$ (where e_i is the i-th vector of basis) with probability $1/n$ for all $i \in [n]$

$$\nabla f_v(w) = \frac{1}{n} \nabla \mathbf{F}(w) n e_i = \nabla f_i(w) \tag{SGD}$$

Arbitrary Sampling

Examples of sampling vector

Let $\nabla \mathbf{F}(w) := [\nabla f_1(w), \ldots, \nabla f_n(w)] \in \mathbb{R}^{d\times n}$ be the Jacobian

- Let $\boldsymbol{v} = \mathbb{1}$ (deterministic)

$$\nabla f_v(w) = \frac{1}{n}\nabla \mathbf{F}(w)\mathbb{1} = \nabla f(w) \tag{GD}$$

- Let $\boldsymbol{v} = \boldsymbol{n e_i}$ (where e_i is the i-th vector of basis) with probability $1/n$ for all $i \in [n]$

$$\nabla f_v(w) = \frac{1}{n}\nabla \mathbf{F}(w) n e_i = \nabla f_i(w) \tag{SGD}$$

- Let $\boldsymbol{v} = (\boldsymbol{n}/\boldsymbol{b})\sum_{\boldsymbol{i}\in \boldsymbol{B}} \boldsymbol{e_i}$ with B drawn from $[n]$ uniformly s.t. $|B| = b$

$$\nabla f_v(w) = \frac{1}{n}\nabla \mathbf{F}(w)\frac{n}{b}\sum_{i\in B} e_i = \frac{1}{b}\sum_{i\in B}\nabla f_i(w)$$

(b–SGD without replacement)

Examples of sampling vector

Let $\nabla \mathbf{F}(w) := [\nabla f_1(w), \ldots, \nabla f_n(w)] \in \mathbb{R}^{d\times n}$ be the Jacobian

- Let $\boldsymbol{v} = \mathbb{1}$ (deterministic)

$$\nabla f_v(w) = \frac{1}{n}\nabla \mathbf{F}(w)\mathbb{1} = \nabla f(w) \qquad \text{(GD)}$$

- Let $\boldsymbol{v} = \boldsymbol{n}\boldsymbol{e}_{\boldsymbol{i}}$ (where e_i is the i-th vector of basis) with probability $1/n$ for all $i \in [n]$

$$\nabla f_v(w) = \frac{1}{n}\nabla \mathbf{F}(w) n e_i = \nabla f_i(w) \qquad \text{(SGD)}$$

- Let $\boldsymbol{v} = (\boldsymbol{n}/\boldsymbol{b})\sum_{\boldsymbol{i}\in \boldsymbol{B}} \boldsymbol{e}_{\boldsymbol{i}}$ with B drawn from $[n]$ uniformly s.t. $|B| = b$

$$\nabla f_v(w) = \frac{1}{n}\nabla \mathbf{F}(w)\frac{n}{b}\sum_{i\in B} e_i = \frac{1}{b}\sum_{i\in B}\nabla f_i(w)$$

(b–SGD without replacement)

→ Arbitrary sampling covers many common methods

Unbiased Gradient Estimate

- **Unbiased estimates**

Unbiased Gradient Estimate

- **Unbiased estimates**
  - Function: $\mathbb{E}_{\mathcal{D}}\left[f_v(w)\right] = \frac{1}{n}\langle F(w), \mathbb{E}_{\mathcal{D}}\left[v\right]\rangle = f(w)$

Unbiased Gradient Estimate

- **Unbiased estimates**
  - Function: $\mathbb{E}_{\mathcal{D}}\left[f_v(w)\right] = \frac{1}{n}\langle F(w), \mathbb{E}_{\mathcal{D}}\left[v\right]\rangle = f(w)$
  - Gradient: $\mathbb{E}_{\mathcal{D}}\left[\nabla f_v(w)\right] = \frac{1}{n}\nabla \mathbf{F}(w)\mathbb{E}_{\mathcal{D}}\left[v\right] = \nabla f(w)$

    where $\nabla \mathbf{F}(w) := [\nabla f_1(w), \ldots, \nabla f_n(w)] \in \mathbb{R}^{d\times n}$

Unbiased Gradient Estimate

- **Unbiased estimates**
  - Function: $\mathbb{E}_{\mathcal{D}}\left[f_v(w)\right] = \frac{1}{n}\langle F(w), \mathbb{E}_{\mathcal{D}}\left[v\right]\rangle = f(w)$
  - Gradient: $\mathbb{E}_{\mathcal{D}}\left[\nabla f_v(w)\right] = \frac{1}{n}\nabla \mathbf{F}(w)\mathbb{E}_{\mathcal{D}}\left[v\right] = \nabla f(w)$

    where $\nabla \mathbf{F}(w) := [\nabla f_1(w), \ldots, \nabla f_n(w)] \in \mathbb{R}^{d\times n}$

$\rightarrow$ **Motivates using:** $\quad w^{k+1} = w^k - \gamma_k \nabla f_v(w^k)$

where γ_k is a step size sequence

Reducing the Variance With Control Variates

- **Controlled stochastic reformulation of the ERM**

$$\text{find } w^* \in \arg\min_{w \in \mathbb{R}^d} \frac{1}{n}\sum_{i=1}^{n} f_i(w) = \mathbb{E}_{\mathcal{D}}\left[f_v(w) \underbrace{- z_v(w) + \mathbb{E}_{\mathcal{D}}\left[z_v(w)\right]}_{\text{unbiased correction term}}\right]$$

with $z_v(\cdot)$ a random function

Reducing the Variance With Control Variates

- **Controlled stochastic reformulation of the ERM**

$$\text{find } w^* \in \underset{w\in\mathbb{R}^d}{\arg\min} \frac{1}{n}\sum_{i=1}^{n} f_i(w) = \mathbb{E}_{\mathcal{D}}\Big[f_v(w) \underbrace{- z_v(w) + \mathbb{E}_{\mathcal{D}}\left[z_v(w)\right]}_{\text{unbiased correction term}}\Big]$$

with $z_v(\cdot)$ a random function

- **Stochastic variance-reduced gradient estimator**

$$g_v(w) := \nabla f_v(w) - \nabla z_v(w) + \mathbb{E}_{\mathcal{D}}\left[\nabla z_v(w)\right]$$

Iteration: $w^{k+1} = w^k - \gamma g_v(w^k)$

Reducing the Variance With Control Variates

- **Controlled stochastic reformulation of the ERM**

$$\text{find } w^* \in \underset{w\in\mathbb{R}^d}{\arg\min} \frac{1}{n}\sum_{i=1}^{n} f_i(w) = \mathbb{E}_{\mathcal{D}}\big[f_v(w) \underbrace{- z_v(w) + \mathbb{E}_{\mathcal{D}}\left[z_v(w)\right]}_{\text{unbiased correction term}}\big]$$

with $z_v(\cdot)$ a random function

- **Stochastic variance-reduced gradient estimator**

$$g_v(w) := \nabla f_v(w) - \nabla z_v(w) + \mathbb{E}_{\mathcal{D}}\left[\nabla z_v(w)\right]$$

Iteration: $w^{k+1} = w^k - \gamma g_v(w^k)$

$\rightarrow$ **Recovers stochastic variance-reduced methods**

Brief recap

Brief recap

- **Iterative methods**

$$w^{k+1} = w^k - \gamma g_v(w^k)$$

Brief recap

- **Iterative methods**

$$w^{k+1} = w^k - \gamma g_v(w^k)$$

Such that

Brief recap

- **Iterative methods**

$$w^{k+1} = w^k - \gamma g_v(w^k)$$

Such that

  - Unbiased gradient: $\mathbb{E}\left[g_v(w^k)\right] = \nabla f(w^k)$

Brief recap

- **Iterative methods**

$$w^{k+1} = w^k - \gamma g_v(w^k)$$

Such that

- Unbiased gradient: $\mathbb{E}\left[g_v(w^k)\right] = \nabla f(w^k)$
- Decreasing variance: $\mathbb{E}\left[\left\|g_v(w^k) - \nabla f(w^k)\right\|_2^2\right] \xrightarrow[w^k \to w^*]{} 0$

Brief recap

- **Iterative methods**

$$w^{k+1} = w^k - \gamma g_v(w^k)$$

  Such that

  - Unbiased gradient: $\mathbb{E}\left[g_v(w^k)\right] = \nabla f(w^k)$
  - Decreasing variance: $\mathbb{E}\left[\left\|g_v(w^k) - \nabla f(w^k)\right\|_2^2\right] \xrightarrow[w^k \to w^*]{} 0$

- **Advantages**
  - **No decreasing step sizes $(\gamma_k)_k$ needed**

Brief recap

- **Iterative methods**

$$w^{k+1} = w^k - \gamma g_v(w^k)$$

Such that

  - Unbiased gradient: $\mathbb{E}\left[g_v(w^k)\right] = \nabla f(w^k)$
  - Decreasing variance: $\mathbb{E}\left[\left\|g_v(w^k) - \nabla f(w^k)\right\|_2^2\right] \xrightarrow[w^k \to w^*]{} 0$

- **Advantages**
  - **No decreasing step sizes $(\gamma_k)_k$ needed**
  - $\boldsymbol{f_v(w)}$ **determines the smoothness** of the controlled stochastic reformulation

Assumption (Expected Smoothness)

$$\mathbb{E}\left[\|\nabla f_v(w) - \nabla f_v(w^*)\|_2^2\right] \leq 2\mathcal{L}\left(f_v(w) - \nabla f_v(w^*)\right)$$

Brief recap

- **Iterative methods**

$$w^{k+1} = w^k - \gamma g_v(w^k)$$

Such that

- Unbiased gradient: $\mathbb{E}\left[g_v(w^k)\right] = \nabla f(w^k)$
- Decreasing variance: $\mathbb{E}\left[\left\|g_v(w^k) - \nabla f(w^k)\right\|_2^2\right] \xrightarrow[w^k \to w^*]{} 0$

- **Advantages**
  - **No decreasing step sizes $(\gamma_k)_k$ needed**
  - $\boldsymbol{f_v(w)}$ **determines the smoothness** of the controlled stochastic reformulation

Assumption (Expected Smoothness)

$$\mathbb{E}\left[\|\nabla f_v(w) - \nabla f_v(w^*)\|_2^2\right] \leq 2\mathcal{L}\left(f_v(w) - \nabla f_v(w^*)\right)$$

- No "bounded gradient" assumption such as $\mathbb{E}\left[\left\|\nabla f_v(w^k)\right\|_2^2\right] \leq cst$

Example: Recovering SAGA Algorithm

- **SAGA**[6]

$$\begin{cases} f_v(w) = \frac{1}{n}\langle F(w), v\rangle & \Longrightarrow \nabla f_v(w) = \frac{1}{n}\nabla \mathbf{F}(w)v \\ z_v(w) = \frac{1}{n}\langle \underbrace{\boldsymbol{J}^\top w}_{\text{linear estimation of } F(w)}, v\rangle & \Longrightarrow \nabla z_v(w) = \frac{1}{n}\boldsymbol{J}v \end{cases}$$

with $\boldsymbol{J}$ an **estimate of the Jacobian** in $R^{d\times n}$

[6] Defazio, Bach and Lacoste-Julien (2014), NIPS, "SAGA: A Fast Incremental Gradient Method With Support for Non-Strongly Convex Composite Objectives"

Example: Recovering SAGA Algorithm

- **SAGA**[6]

$$\begin{cases} f_v(w) = \frac{1}{n}\langle F(w), v\rangle & \Longrightarrow \nabla f_v(w) = \frac{1}{n}\nabla \mathbf{F}(w)v \\ z_v(w) = \frac{1}{n}\langle \underbrace{\boldsymbol{J}^\top w}_{\text{linear estimation of } F(w)}, v\rangle & \Longrightarrow \nabla z_v(w) = \frac{1}{n}\boldsymbol{J}v \end{cases}$$

with $\boldsymbol{J}$ an **estimate of the Jacobian** in $R^{d\times n}$

- If $\boldsymbol{v} = \boldsymbol{ne_i}$, where e_i is the i-th vector of basis

$$\begin{aligned} g_v(w) &:= \nabla f_v(w) - \nabla z_v(w) + \mathbb{E}_{\mathcal{D}}\left[\nabla z_v(w)\right] \\ &= \frac{1}{n}\nabla \mathbf{F}(w)ne_i - \frac{1}{n}\boldsymbol{J}ne_i + \mathbb{E}\left[\frac{1}{n}\boldsymbol{J}v\right] \\ &= \nabla f_i(w) - \boldsymbol{J}_{:i} + \frac{1}{n}\boldsymbol{J}\mathbb{1} \end{aligned}$$

where $\mathbb{1}$ is the all-ones vector and $\boldsymbol{J}_{:i}$ the $i-$th column of $\boldsymbol{J}$

[6]Defazio, Bach and Lacoste-Julien (2014), NIPS, "SAGA: A Fast Incremental Gradient Method With Support for Non-Strongly Convex Composite Objectives"

Example: Recovering SAGA Algorithm

- **SAGA**[6]

$$\begin{cases} f_v(w) = \frac{1}{n}\langle F(w), v\rangle & \implies \nabla f_v(w) = \frac{1}{n}\nabla \mathbf{F}(w)v \\ z_v(w) = \frac{1}{n}\langle \underbrace{\boldsymbol{J}^\top w}_{\text{linear estimation of } F(w)}, v\rangle & \implies \nabla z_v(w) = \frac{1}{n}\boldsymbol{J}v \end{cases}$$

  with $\boldsymbol{J}$ an **estimate of the Jacobian** in $R^{d\times n}$

- If $\boldsymbol{v = ne_i}$, where e_i is the i-th vector of basis

$$\begin{aligned} g_v(w) &:= \nabla f_v(w) - \nabla z_v(w) + \mathbb{E}_{\mathcal{D}}\left[\nabla z_v(w)\right] \\ &= \frac{1}{n}\nabla \mathbf{F}(w)ne_i - \frac{1}{n}\boldsymbol{J}ne_i + \mathbb{E}\left[\frac{1}{n}\boldsymbol{J}v\right] \\ &= \nabla f_i(w) - \boldsymbol{J}_{:i} + \frac{1}{n}\boldsymbol{J}\mathbb{1} \end{aligned}$$

  where $\mathbb{1}$ is the all-ones vector and $\boldsymbol{J}_{:i}$ the $i-$th column of $\boldsymbol{J}$

- **Variance term**

  Convergence analysis: $\mathbb{E}_{\mathcal{D}}\left[\|g_v(w) - \nabla f(w)\|_2^2\right]$ low for $\boldsymbol{J} \approx \nabla \mathbf{F}(w)$

[6] Defazio, Bach and Lacoste-Julien (2014), NIPS, "SAGA: A Fast Incremental Gradient Method With Support for Non-Strongly Convex Composite Objectives"