Finite-Sum Optimization: Adaptivity to Smoothness and Loopless Variance Reduction

Bastien Batardière, Joon Kwon

February 5, 2024

AgroParisTech
Talents d'une planète soutenable

INRAE

Context

- Large scale optimization

  Number of samples $n >> 1$

  Number of parameters $d >> 1$

- Examples
  - scRNA seq data where $n \approx d \approx 20000$
  - Any log-likelihood with lots of samples.

Problem

- Goal:

$$\arg\min_{\theta \in \mathbb{R}^d} f(\theta)$$

where f has a finite-sum structure:

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

- Example of loss functions:
  1. $f_i(\theta) = -\log P_\theta(X_i = x_i)$
  2. Variational ELBOs can be designed this way:

$$J(\theta, \phi | Y) = \frac{1}{n} \sum_{i=1}^{n} J_i(\theta, \phi_i | Y_i)$$

with Y the data, ϕ the variational parameters, θ the model parameters.

Outline

Assumptions

1. Each f_i is convex and differentiable
2. Each f_i is $L-$ smooth:

$$\|\nabla f_i(x) - \nabla f_i(y)\| \leqslant L\|x - y\|, \qquad x, y \in \mathbb{R}^d, \quad 1 \leqslant i \leqslant n.$$

Examples:

- Linear regression $f_i(\theta) = \|x_i^\top \theta - y_i\|^2, (x_i, y_i)_{1 \leqslant i \leqslant n}$
- Logistic regression
  $f_i(\theta) = \log(\mathbb{P}_\theta(X_i = x_i))y_i + \log(1 - \mathbb{P}_\theta(X_i = x_i))(1 - y_i)$

Goal: reach an $\epsilon > 0$ solution: find θ such that

$$f(\theta) - f(\theta^\star) \leqslant \epsilon$$

Coefficient L is not always known.

Stochastic Gradient Descent [Robbins, 1951]

- Stochastic GD (SGD): replace $\nabla f(\theta_t)$ of classic Gradient Descent with an unbiased low cost gradient:

$$i_t \sim \text{unif}(1 \cdots n)$$
$$\theta_{t+1} = \theta_t - \eta_{\mathbf{t}} \nabla f_{\mathbf{i_t}}(\theta_t)$$

- Unbiased gradient estimator:

$$\mathbb{E}\left[\nabla f_{i_t}(\theta_t)\right] = \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(\theta_t) = \nabla f(\theta_t)$$

- If $\eta_t = \frac{1}{\sqrt{t}}$, complexity in $O(\frac{1}{\epsilon^2})$ iteration

Stochastic Averaging GrAdient (SAGA) [Defazio et al., 2014]

- Store the previously computed gradient in a matrix $\phi^{(t)} \in \mathbb{R}^{n \times d}$ to estimate the gradient of the whole dataset. We aim to have

$$\frac{1}{n}\sum_{i=1}^{n}\phi_i^{(t)} \approx \nabla f(\theta_t)$$

- Given $\phi^{(1)} \in \mathbb{R}^{n \times d}, \theta_1$, perform

$$i_t \sim \text{unif}(1 \cdots n) \tag{1}$$

$$g_t = \nabla f_{\mathbf{i_t}}(\theta_t) - \phi_{i_t}^{(t)} + \frac{1}{n}\sum_{i=1}^{n}\phi_i^{(t)} \tag{2}$$

$$\theta_{t+1} = \theta_t - \eta g_t \tag{3}$$

$$\text{Update } \phi_i^{(t+1)} = \begin{cases} \phi_i^{(t)} & \text{if } i \neq i_t \\ \nabla f_{\mathbf{i_t}}(\theta_t) & \text{if } i = i_t \end{cases} \tag{4}$$

Goal: reduce the variance of the gradient.

Update of the matrix ϕ, when $n = 3, d = 1$

$$\begin{pmatrix} \phi_0 & \phi_0 & \phi_0 \end{pmatrix}$$

Update of the matrix ϕ, when $n = 3, d = 1$

$$\begin{pmatrix} \phi_0 & \phi_0 & \phi_0 \end{pmatrix}$$
$$\mathrm{t} = 1,\ \mathrm{i(t)} = 2 \implies \nabla f_2(\theta_1) \text{ is computed}$$
$$\Downarrow$$

Update of the matrix ϕ, when $n = 3, d = 1$

$$\begin{pmatrix} \phi_0 & \phi_0 & \phi_0 \end{pmatrix}$$

$$\mathrm{t} = 1,\ \mathrm{i}(\mathrm{t}) = 2 \implies \nabla f_2(\theta_1) \text{ is computed}$$

$$\Downarrow$$

$$\begin{pmatrix} \phi_0 & \nabla f_2(\theta_1) & \phi_0 \end{pmatrix}$$

Update of the matrix ϕ, when $n = 3, d = 1$

$$\begin{pmatrix} \phi_0 & \phi_0 & \phi_0 \end{pmatrix}$$

$$t = 1,\ i(t) = 2 \implies \nabla f_2(\theta_1) \text{ is computed}$$

$$\Downarrow$$

$$\begin{pmatrix} \phi_0 & \nabla f_2(\theta_1) & \phi_0 \end{pmatrix}$$

$$t = 2,\ i(t) = 3 \implies \nabla f_3(\theta_2) \text{ is computed}$$

$$\Downarrow$$

Update of the matrix ϕ, when $n = 3, d = 1$

$$\begin{pmatrix} \phi_0 & \phi_0 & \phi_0 \end{pmatrix}$$

$$\mathrm{t} = 1,\ \mathrm{i}(\mathrm{t}) = 2 \implies \nabla f_2(\theta_1) \text{ is computed}$$

$$\Downarrow$$

$$\begin{pmatrix} \phi_0 & \nabla f_2(\theta_1) & \phi_0 \end{pmatrix}$$

$$\mathrm{t} = 2,\ \mathrm{i}(\mathrm{t}) = 3 \implies \nabla f_3(\theta_2) \text{ is computed}$$

$$\Downarrow$$

$$\begin{pmatrix} \phi_0 & \nabla f_2(\theta_1) & \nabla f_3(\theta_2) \end{pmatrix}$$

Update of the matrix ϕ, when $n = 3, d = 1$

$$\begin{pmatrix} \phi_0 & \phi_0 & \phi_0 \end{pmatrix}$$

$$t = 1,\ i(t) = 2 \implies \nabla f_2(\theta_1) \text{ is computed}$$

$$\Downarrow$$

$$\begin{pmatrix} \phi_0 & \nabla f_2(\theta_1) & \phi_0 \end{pmatrix}$$

$$t = 2,\ i(t) = 3 \implies \nabla f_3(\theta_2) \text{ is computed}$$

$$\Downarrow$$

$$\begin{pmatrix} \phi_0 & \nabla f_2(\theta_1) & \nabla f_3(\theta_2) \end{pmatrix}$$

$$t = 3,\ i(t) = 3 \implies \nabla f_3(\theta_3)$$

$$\Downarrow$$

Update of the matrix ϕ, when $n = 3, d = 1$

$$\begin{pmatrix} \phi_0 & \phi_0 & \phi_0 \end{pmatrix}$$

$$\mathrm{t} = 1,\ \mathrm{i(t)} = 2 \implies \nabla f_2(\theta_1) \text{ is computed}$$

$$\Downarrow$$

$$\begin{pmatrix} \phi_0 & \nabla f_2(\theta_1) & \phi_0 \end{pmatrix}$$

$$\mathrm{t} = 2,\ \mathrm{i(t)} = 3 \implies \nabla f_3(\theta_2) \text{ is computed}$$

$$\Downarrow$$

$$\begin{pmatrix} \phi_0 & \nabla f_2(\theta_1) & \nabla f_3(\theta_2) \end{pmatrix}$$

$$\mathrm{t} = 3,\ \mathrm{i(t)} = 3 \implies \nabla f_3(\theta_3)$$

$$\Downarrow$$

$$\begin{pmatrix} \phi_0 & \nabla f_2(\theta_1) & \nabla f_3(\theta_3) \end{pmatrix}$$

Adagrad [Duchi et al., 2011]

- Given θ_1, $G_0 = 0_{\mathbb{R}^d}$, perform

$$
\begin{aligned}
i_t &\sim \text{unif}(1 \cdots n) \\
g_t &= \nabla f_{i_t}(\theta_t) \\
G_t &= G_{t-1} + g_t \odot g_t \\
\theta_{t+1} &= \theta_t - \frac{\eta}{\sqrt{G_t}} \odot g_t
\end{aligned}
$$

Goal: adapting the learning rate according to the data.

AdaSAGA

Goal: use both variance reduction from SAGA and adaptation from Adagrad.

- Given $\phi^{(1)} \in \mathbb{R}^{n \times d}, \theta_1, G_0 = 0_{\mathbb{R}^d}$, perform

$$i_t \sim \text{unif}(1 \cdots n)$$

$$g_t = \nabla f_{i_t}(\theta_t) - \phi_{i_t}^{(t)} + \frac{1}{n} \sum_{i=1}^{n} \phi_i^{(t)}$$

$$G_t = G_{t-1} + g_t \odot g_t$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{G_t}} \odot g_t$$

$$\text{Update } \phi_i^{(t+1)} = \begin{cases} \phi_i^{(t)} & \text{if } i \neq i_t \\ \nabla f_{i_t}(\theta_t) & \text{if } i = i_t \end{cases}$$

Convergence results

Theorem

Let $T \geqslant 1, \eta > 0, x^{(1)} \in \mathbb{R}^d$ *and* $\left(x^{(t)}\right)_{1 \leqslant t \leqslant T}$ *be a sequence of iterates defined by AdaSAGA. Then, under convexity and L-smoothness, it holds that:*

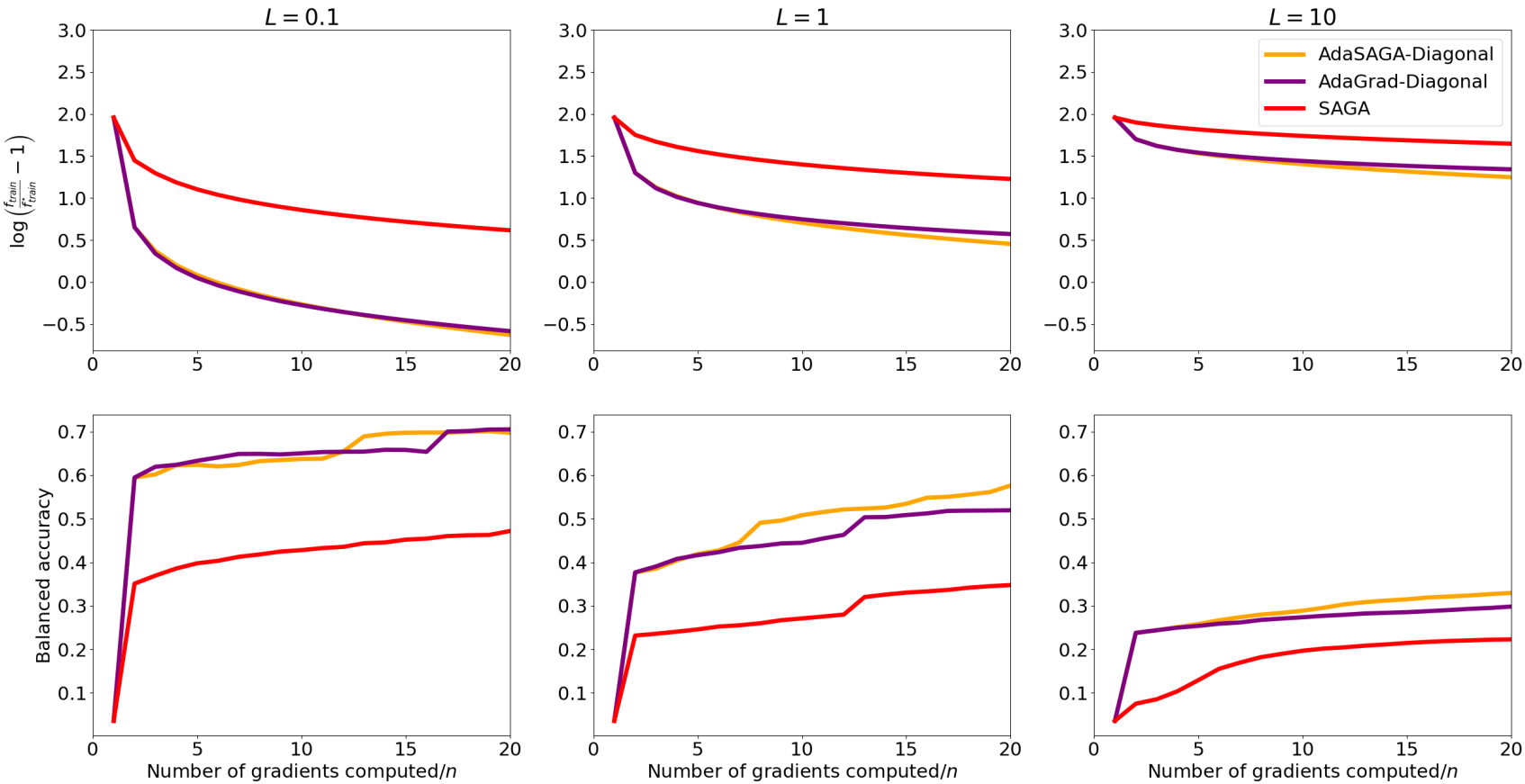
$$\mathbb{E}\left[f(\overline{x}_T)\right] - f\left(x^\star\right) \leqslant \frac{\alpha\left(\eta + \frac{D^2}{2\eta}\right)\sqrt{4Ln\Delta_1} + 8L\alpha^2\left(\eta + \frac{D^2}{2\eta}\right)^2}{T},$$

where $\overline{x}_T = \frac{1}{T}\sum_{t=1}^{T} x^{(t)}$, $\Delta_1 = f\left(x^{(1)}\right) - f\left(x^\star\right)$ *and* $\alpha = 1$ *(resp.* $\alpha = \sqrt{d}$*) for the Norm variant (resp. the Diagonal variant).*

Complexity summary

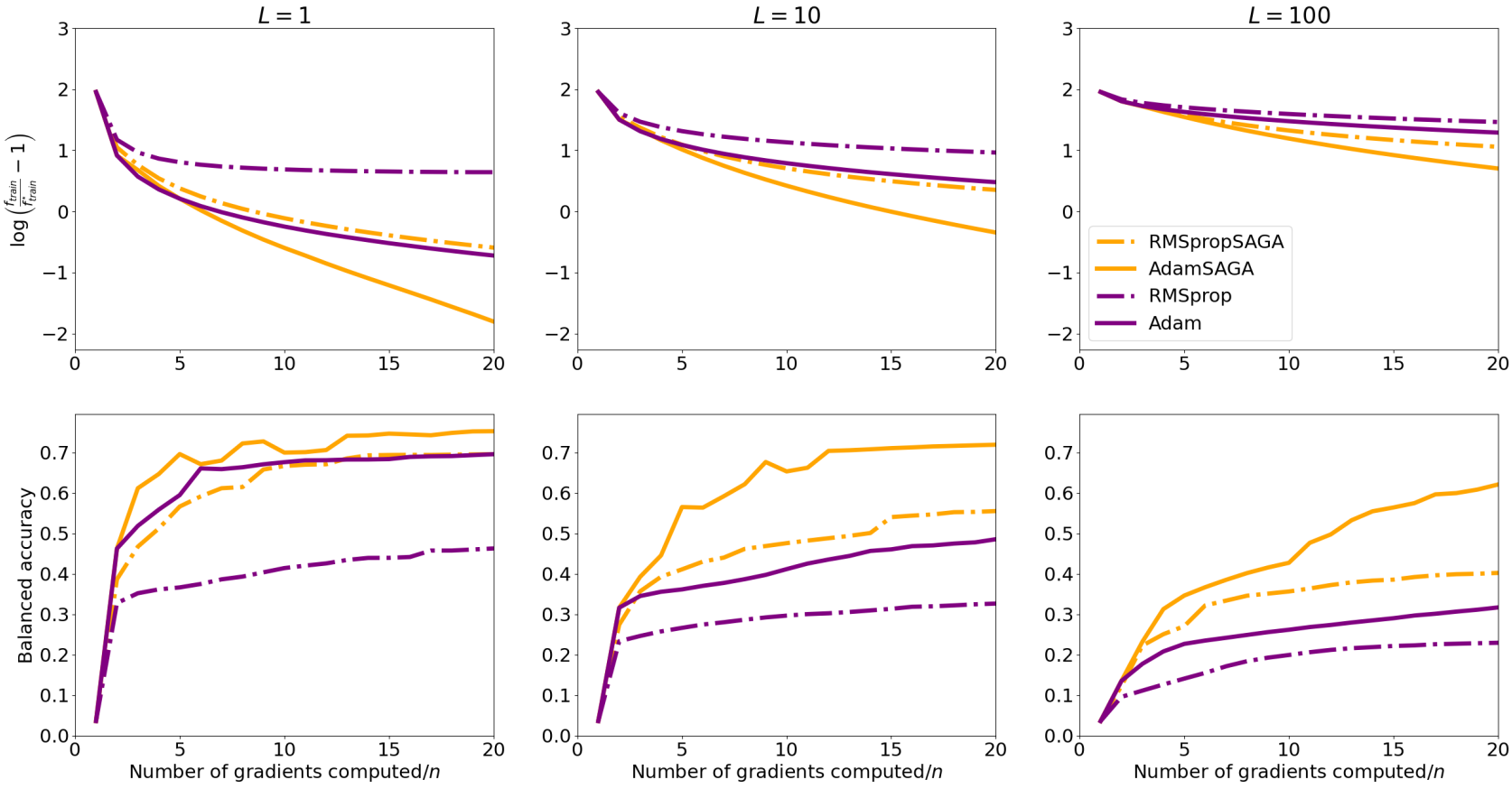
	Gradient complexity to reach an ϵ-solution
GD	$O(\frac{nL}{\epsilon})$
SGD	$O(\frac{\sigma}{\epsilon^2})$
SAGA	$O(\frac{n+L}{\epsilon})$
Adagrad	$O(\frac{\sigma}{\epsilon^2})$
AdaSAGA	$O(\frac{L+\sqrt{Ln}}{\epsilon})$

Numerical experiments



Logistic regression on scMark(20000×8400). Comparison between AdaGrad, SAGA and AdaSAGA.

Numerical experiments



Comparison between Adam, RMSprop, AdamSAGA and RMSPropSAGA on SCMark dataset ($n = 20000, d = 8400$).

Conclusions

- Faster convergence than AdaGrad
- Adaptivity to smoothness
- More effective when replacing AdaGrad with Adam

Perspectives

- Proof when combine with Adam
- Combining with acceleration methods (e.g. Katyusha, Varag, Vrada)
- Non-convex objectives

Thank you for your attention


Allen-Zhu, Z. (2018).
Katyusha: The first direct acceleration of stochastic gradient methods.


Defazio, A., Bach, F., and Lacoste-Julien, S. (2014).
Saga: A fast incremental gradient method with support for non-strongly convex composite objectives.


Dubois-Taine, B., Vaswani, S., Babanezhad, R., Schmidt, M., and Lacoste-Julien, S. (2021).
Svrg meets adagrad: Painless variance reduction.


Duchi, J., Hazan, E., and Singer, Y. (2011).
Adaptive subgradient methods for online learning and stochastic optimization.
J. Mach. Learn. Res., 12(null):2121–2159.


Kingma, D. P. and Ba, J. (2014).
Adam: A method for stochastic optimization.
ICML.


Lan, G., Li, Z., and Zhou, Y. (2019).
A unified variance-reduced accelerated gradient method for