ProxSARAH: An Efficient Algorithmic Framework For Stochastic Composite Nonconvex Optimization

Nhan H. Pham
nhanph@live.unc.edu, nhanph.github.io

Department of Statistics and Operations Research
The University of North Carolina at Chapel Hill (UNC)
North Carolina

INFORMS Annual Meeting 2019, Seattle, WA

[Oct 21, 2019]

Joint work with

Quoc Tran-Dinh (UNC), Lam Nguyen, and Dzung Phan (IBM).

THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Outline

Overview and References

COMPOSITE **NONCONVEX** OPTIMIZATION

$$\min_{x\in\mathbb{R}^d} \left\{ F(x) := \underbrace{\mathbb{E}\left[f(x,\xi)\right]}_{f(x)} \ + \ \psi(x) \right\}$$

- $f(x)$ is nonconvex and smooth.
- $\psi(x)$ is convex and possibly nonsmooth to handle regularizers, penalty, or constraints.

Majority of this talk is based on the following manuscript:

- N. H. Pham, L. M. Nguyen, D. T. Phan, and T.D. ProxSARAH: An Efficient Algorithmic Framework for Stochastic Composite Nonconvex Optimization. Preprint: https://arxiv.org/pdf/1902.05679.pdf, 2019.

Problems of Interest

Composite (Expectation) Nonconvex Optimization

$$\min_{x\in\mathbb{R}^d} \left\{ F(x) := f(x) + \psi(x) \equiv \mathbb{E}[f(x,\xi)] \ + \ \psi(x) \right\}, \qquad \text{(NCVX)}$$

where

- $f(x) := \mathbb{E}\left[f(x,\xi)\right] : \mathbb{R}^d \to \mathbb{R}\cup\{+\infty\}$: smooth and nonconvex expected function.
- $\psi : \mathbb{R}^d \to \mathbb{R}\cup\{+\infty\}$ is convex and possibly nonsmooth.
- ψ can be proximally friendly.

  Note: "proximally friendly" is not necessary for theoretical results, but for practice.

Composite finite-sum minimization problem

If $f_i(x) := f(x,\xi_i)$ $(i = 1,\cdots,n)$, then (NCVX) reduces to:

$$\min_{x\in\mathbb{R}^d} \left\{ F(x) := f(x) + \psi(x) \equiv \frac{1}{n}\sum_{i=1}^{n} f_i(x) \ + \ \psi(x) \right\}. \qquad \text{(ERM)}$$

Also arising from a sample averaging approximation (SAA) approach.

Motivation

Applications

- Problem (NCVX) and (ERM) cover many applications in **different domains**, including machine learning, statistics, and finance.
  - Empirical risk minimization
  - Neural network training (many talks have mentioned).
  - Many more ...

Theoretical aspect

- **Modern variance reduction** methods mostly focus on non-composite forms.
- **Gap** between the upper bound complexity in current research and lower bound worst-case complexity for (ERM).
- There exists no lower bound complexity for (NCVX), motivating to improve upper bound complexity (?)

Proximal Tractability: Review

Proximal operator

- For a given **convex** function ψ, we define:

$$\mathrm{prox}_{\psi}(x) := \arg\min_{y} \left\{ \psi(y) + \tfrac{1}{2}\|y - x\|^2 \right\}$$

  the **proximal operator** of ψ.
- If $\mathrm{prox}_{\psi}(x)$ is efficient to evaluate, e.g. by:
  - a closed form or
  - a low-order polynomial-time algorithm,

  then we say that ψ is tractably proximal or proximally friendly.

Common examples

- ψ is some common norms: ℓ_1, ℓ_2, ℓ_∞, and nuclear norm.
- ψ is separable functions: group sparsity.
- ψ is the indicator function of a simple set such as box, cone, or simplex, i.e.:

$$\psi(x) = \begin{cases} 0 & \text{if } x \in \mathcal{X}, \\ +\infty & \text{otherwise.} \end{cases}$$

First-order Stationary Points

Optimality condition and first-order stationary points

- Given $F = f + \psi$, the **gradient mapping** of F is defined by

$$G_\eta(x) := \frac{1}{\eta}\left(x - \mathrm{prox}_{\eta\psi}\left(x - \eta\nabla f(x)\right)\right), \quad \eta > 0.$$

- Optimality condition:

$$\mathbb{E}\left[\|G_\eta(x^\star)\|^2\right] = 0. \tag{1}$$

  Any $x^\star$ satisfies (1) is called a **first-order stationary point** of (NCVX).

Approximate first-order stationary points

- Finding an ε-approximate stationary point x_T to $x^\star$ in (1) after at most T iterations within a given accuracy $\varepsilon > 0$, i.e.

$$\mathbb{E}\left[\|G_\eta(x_T)\|^2\right] \leq \varepsilon^2.$$

- How fast does $\mathbb{E}\left[\|G_\eta(x_T)\|^2\right]$ converge to 0?
  - **Iteration-complexity:** Total number of iterations.
  - **First-order oracle complexity:** Total number of stochastic first-order (SFO) evaluations.
  - **Proximal operations:** Total number of $\mathrm{prox}_{\eta\psi}$ operations.

Structural Assumptions on the Models

Fundamental assumptions

- **Boundedness from below:** $F^{\star} := \inf_{x \in \mathbb{R}^p} F(x) > -\infty$.
- L**-average smoothness:** For all $x, \hat{x} \in \mathrm{dom} f$:

Expectation: $$\mathbb{E}_{\xi}\left[\|\nabla_x f(x,\xi) - \nabla_x f(\hat{x},\xi)\|^2\right] \leq L^2 \|x - \hat{x}\|^2.$$

Finite-sum: $$\frac{1}{n}\sum_{i=1}^{n} \|\nabla f_i(x) - \nabla f_i(\hat{x})\|^2 \leq L^2 \|x - \hat{x}\|^2.$$

- **Bounded variance:** For all $x \in \mathrm{dom} f$:

$$\mathbb{E}_{\xi}\left[\|\nabla_x f(x,\xi) - \nabla f(x)\|^2\right] \leq \sigma^2.$$

Our Goals and Main Contributions

Our goals

- Develop **new proximal SARAH**[1] variants to solve both (NCVX) and (ERM).
  - Achieve the optimal complexity bounds or the best-known complexity bounds.
  - Less parameters tuning.

Main theoretical contributions

- New proximal variance reduction stochastic gradient algorithms to solve both (NCVX) and (ERM)
- Obtaining best-known complexity in both expectation and finite-sum cases
  - Optimal complexity bound for (ERM).
- Adaptive step-size variants that outperform the constant step-sizes schemes.

[1]SARAH (stochastic recursive gradient estimator) was introduced by Nguyen et al in an ICML paper, 2017.

Classical Proximal SGD and Other Single-loop Variants

Classical proximal SGD

Starting from x_0, SGD generates $\{x_t\}$ by updating:

$$x_{t+1} = \mathrm{prox}_{\eta_t \psi}\left(x_t - \eta_t u_t\right),$$

where

- $u_t := \nabla_x f(x_t; \xi_t)$ for (NCVX) or $u_t := \nabla_x f_{i_t}(x_t)$ for (ERM).
- u_t is an unbiased estimator of $\nabla f(x_t)$, i.e. $\mathbb{E}[u_t] = \nabla f(x_t)$.

- Using mini-batches, intermediate steps, averaging, momentum, etc.
- **Key point:** How to choose step-size η_t? (also called **learning rate**).

Other single-loop SGD-type schemes

- **SAGA**, **AdaGrad**, **ADAM**, etc.

Double-loop Algorithms: Variance reduction

Notable variants

- **SVRG** [2]: Both **double-loop** and **loopless** variants. The most popular one.
- **SARAH** [4]: Some notable variants such as SPIDER, SpiderBoost, etc.
- Not yet ready for DL?: **Empirical performance** is worse than standard SGD and ADAM in general.
- **May need to tune many parameters**.

Algorithm 1 (General double-loop algorithms)

1: Initialize $\tilde{x}_0$ and learning rate $\eta_t > 0$.
2: **OuterLoop: For** $s := 1, 2, \cdots, S$ **do**
3: Generate a gradient snapshot $v_0^{(s)}$ at $x_0^{(s)} := \tilde{x}_{s-1}$.
4: **InnerLoop: For** $t := 1, \cdots, m$ **do**
5: Compute stochastic gradient estimator $v_t^{(s)}$.
6: Update $x_{t+1}^{(s)} := \mathrm{prox}_{\eta_t \psi}(x_t^{(s)} - \eta_t v_t^{(s)})$.
7: **EndFor**
8: Choose $\tilde{x}_s$ from $\{x_0^{(s)}, \cdots, x_{m+1}^{(s)}\}$.
9: **EndFor**

Iteration Complexity and Oracle Complexity: A Summary

Iteration complexity and oracle complexity

- **Iteration complexity:** Total number of iterations to achieve an ε-stationary point.
- **First-order oracle complexity:** Total number of stochastic gradient evaluations and proximal operations.

Complexity summary (non-exhaustive)

Algorithms	**Finite-sum**	**Expectation**	**Step-size**	Composite	**Adaptive step-size**
GD	$\mathcal{O}\left(\frac{n}{\varepsilon^2}\right)$	NA	$\mathcal{O}\left(L^{-1}\right)$	Yes	Yes
SGD	NA	$\mathcal{O}\left(\sigma^2\varepsilon^{-4}\right)$	$\mathcal{O}\left(L^{-1}\right)$	Yes	Yes
SVRG	$\mathcal{O}\left(n + n^{2/3}\varepsilon^{-2}\right)$	NA	$\mathcal{O}\left((nL)^{-1}\right) \to \mathcal{O}\left(L^{-1}\right)$	Yes	No
SPIDER	$\mathcal{O}\left(n + n^{1/2}\varepsilon^{-2}\right)$	$\mathcal{O}\left(\sigma^2\varepsilon^{-2} + \sigma\varepsilon^{-3}\right)$	$\mathcal{O}\left(L^{-1}\varepsilon\right)$	No	Yes
SpiderBoost	$\mathcal{O}\left(n + n^{1/2}\varepsilon^{-2}\right)$	$\mathcal{O}\left(\sigma^2\varepsilon^{-2} + \sigma\varepsilon^{-3}\right)$	$\mathcal{O}\left(L^{-1}\right)$	Yes	No
ProxSARAH	$\mathcal{O}\left(n + n^{1/2}\varepsilon^{-2}\right)$	$\mathcal{O}\left(\sigma^2\varepsilon^{-2} + \sigma\varepsilon^{-3}\right)$	$\mathcal{O}\left(L^{-1}m^{-1/2}\right) \to \mathcal{O}\left(L^{-1}\right)$	Yes	Yes

Table: Comparison of results on SFO (stochastic first-order oracle) complexity for nonsmooth non-convex optimization (both non-composite and composite cases).

Common Stochastic Gradient Estimators

Common stochastic gradient estimators

- **SGD estimators**: unbiased and fixed variance

$$u_t := \nabla f(x_t, \xi_t) \ \ \text{(singe sample)} \ \ \text{or} \ \ u_t := \frac{1}{b_t} \sum_{\xi_t \in \mathcal{B}_t} \nabla f(x_t, \xi_t) \ \ \text{(batch)}.$$

- **SAGA:** Only for finite-sum problems, unbiased, and variance reduced:

$$v_t := \nabla f_{i_t}(z_{t+1}^{i_t}) - \nabla f(z_t^{i_t}) + \frac{1}{n} \sum_{i=1}^{n} \nabla f(z_t^i),$$

where $z_{t+1}^{i_t} = x_t$ if $i_t = i$, and $z_{t+1}^i = z_t^i$ if $i \neq i_t$.

- **SVRG:** unbiased and variance reduced estimator

$$v_t := \widetilde{u}_t + \nabla f(x_t, \xi_t) - \nabla f(\widetilde{x}, \xi_t),$$

where $\widetilde{x}$ is a snapshot point, and $\widetilde{u}_t$ is an unbiased estimator of ∇f at $\widetilde{x}$.

- **SARAH:** biased and variance reduced estimator

$$v_t := v_{t-1} + \nabla f(x_t, \xi_t) - \nabla f(x_{t-1}, \xi_t).$$

Main Idea and Main Steps

Related works

- **SPIDER**, **SpiderBoost**, and **some other variants**: Update a plain proximal step $x_{t+1}^{(s)} := \mathrm{prox}_{\eta\psi}\left(x_t^{(s)} - \eta v_t^{(s)}\right)$ using SARAH estimator:

$$v_t^{(s)} := v_{t-1}^{(s)} + \left(\nabla f(x_t^{(s)}, \xi_t) - \nabla f(x_{t-1}^{(s)}, \xi_t)\right). \tag{SARAH}$$

- Require **batch** and constant/adaptive step-size to obtain best-known complexity.
- **SPIDER** performs poorly due to small step-size
- **SpiderBoost** performs well in practice if well tuning parameters.

Our scheme

- **ProxSARAH: one proximal step** and **one averaging step**:

$$\begin{cases} \widehat{x}_{t+1}^{(s)} & := \mathrm{prox}_{\eta_t\psi}\left(x_t^{(s)} - \eta_t v_t^{(s)}\right), \\ x_{t+1}^{(s)} & := (1-\gamma_t)x_t^{(s)} + \gamma_t \widehat{x}_{t+1}^{(s)}. \end{cases} \tag{ProxSARAH}$$

- Additional damped step-size $\gamma_t \to$ **more flexibility**.

Proximal SARAH algorithm (ProxSARAH)

Algorithm 2 (ProxSARAH: A simplified version)

1: Choose an initial $\tilde{x}_0$, fix a parameter $\eta > 0$.
2: **OuterLoop: For** $s := 1, 2, \cdots, S$ **do**
3: Generate a snapshot $v_0^{(s)}$ as a stochastic estimator of $\nabla f(x_0^{(s)})$.
4: Update $\widehat{x}_1^{(s)} := \mathrm{prox}_{\eta\psi}(x_0^{(s)} - \eta v_0^{(s)})$ and $x_1^{(s)} := (1-\gamma_0)x_0^{(s)} + \gamma_0\widehat{x}_1^{(0)}$.
5: **InnerLoop: For** $t := 1, \cdots, m$ **do**
6: Evaluate SARAH estimator $v_t^{(s)}$
7: Update $\widehat{x}_{t+1}^{(s)} := \mathrm{prox}_{\eta\psi}(x_t^{(s)} - \eta v_t^{(s)})$ and $x_{t+1}^{(s)} := (1-\gamma_t)x_t^{(s)} + \gamma_t\widehat{x}_{t+1}^{(s)}$
8: **EndFor**
9: Set $\widetilde{x}_s := x_{m+1}^{(s)}$
10: **EndFor**

Remarks

- The **outer loop** in **ProxSARAH** is mandatory to guarantee convergence.
- Both step-sizes η and γ can be fixed or adaptively updated.
- Work with both single sample and mini-batch.
- The **main step** can be written as $x_{t+1} := x_t - \gamma_t \eta G_\eta(x_t)$.

Convergence Guarantee: Summary

Convergence in the finite-sum case (ERM)

Let the step-sizes γ, η be fixed or updated adaptively. If we choose snapshot batch size $b := n$ and epoch length $m := n$, then to guarantee $\mathbb{E}\left[\|G_\eta(\widetilde{x}_T)\|^2\right] \leq \varepsilon^2$, the followings hold

- The number of outer iterations S does not exceed

$$S := \mathcal{O}\left(\frac{L}{\sqrt{n}\varepsilon^2}\left[F(\widetilde{x}_0) - F^\star\right]\right).$$

- The number of stochastic gradient evaluations $\mathcal{T}_{\mathrm{grad}}$ does not exceed

$$\mathcal{T}_{\mathrm{grad}} := \mathcal{O}\left(\frac{L\sqrt{n}}{\varepsilon^2}\left[F(\widetilde{x}_0) - F^\star\right]\right),$$

- The number of $\mathrm{prox}_{\eta\psi}$ operations does not exceed

$$\mathcal{T}_{\mathrm{prox}} := \mathcal{O}\left(\frac{L\sqrt{n}}{\varepsilon^2}\left[F(\widetilde{x}_0) - F^\star\right]\right).$$

Convergence Guarantee: Summary (cont.)

Convergence in the expectation case (NCVX)

Let the step-sizes γ, η be fixed or updated adaptively. If we choose snapshot batch size $b := \mathcal{O}\left(\frac{\sigma^2}{\epsilon^2}\right)$ and epoch length $m := \mathcal{O}\left(\frac{\sigma^2}{\epsilon^2}\right)$, then to guarantee $\mathbb{E}\left[\|G_\eta(\widetilde{x}_T)\|^2\right] \leq \varepsilon^2$, the followings hold

- The number of outer iterations S is at most

$$S := \mathcal{O}\left(\frac{L[F(\widetilde{x}_0) - F^\star]}{\sigma\varepsilon}\right).$$

- The number of individual stochastic gradient evaluations $\nabla f(\cdot, \xi_t)$ does not exceed

$$\mathcal{T}_{\text{grad}} := \mathcal{O}\left(\frac{L\sigma}{\varepsilon^3}\left[F(\widetilde{x}_0) - F^\star\right]\right),$$

- The number of $\mathrm{prox}_{\eta\psi}$ operations does not exceed

$$\mathcal{T}_{\text{prox}} := \mathcal{O}\left(\frac{\sigma L[F(\widetilde{x}_0) - F^\star]}{\varepsilon^2}\right).$$

Optimal Complexity for the Finite-sum Case

Lower bound complexity for the finite-sum problem

Fang et al.[2] and Zhou et al.[3] showed that under standard assumptions, the lower bound complexity of SGD on $\mathcal{T}_{\mathrm{grad}}$ is

$$\Omega\left(\frac{L\left[F(x^0) - F^\star\right]\sqrt{n}}{\varepsilon^2}\right).$$

A few remarks

For the finite-sum case:

- If $n = \mathcal{O}\left(\varepsilon^{-4}\right)$, then $\mathcal{T}_{\mathrm{grad}} = \mathcal{O}\left(n^{1/2}\varepsilon^{-2}\right)$.
- If $n = \Omega\left(\varepsilon^{-4}\right)$, then $\mathcal{T}_{\mathrm{grad}} = \mathcal{O}\left(n + n^{1/2}\varepsilon^{-2}\right)$ due to the full gradient snapshots.

For the expectation case:

- If $\sigma \leq \frac{32L[F(\widetilde{x}_0)-F^\star]}{\varepsilon^2}$, then $\mathcal{T}_{\mathrm{grad}} = \mathcal{O}\left(\sigma\varepsilon^{-3}\right)$.
- Otherwise, $\mathcal{T}_{\mathrm{grad}} = \mathcal{O}\left(\sigma\varepsilon^{-3} + \sigma^2\varepsilon^{-2}\right)$ due to the snapshot $v_0^{(s)}$

[2] C. Fang, C. J. Li, Z. Lin, and T. Zhang. SPIDER: Near-optimal non-convex optimization via stochastic path integrated differential estimator. arXiv preprint arXiv:1807.01695, 2018.

[3] D. Zhou and Q. Gu. Lower bounds for smooth nonconvex finite-sum optimization. arXiv preprint arXiv:1901.11224, 2019.

Three Numerical Examples

Nonconvex optimization models

- **Simple example:** Nonnegative principal component analysis (NN-PCA)
- **Binary classification:** Sparse binary classification with nonconvex losses
- **DL relations:** Sparse feedforward neural network training

Our numerical examples are still very preliminary. Our code can be found at:

https://github.com/unc-optimization/StochasticProximalMethods.

Comparison criteria

- The norm of gradient mapping $\|G_\eta(x_t^{(s)})\|$ with $(\eta = 0.5)$
- Training loss values.
- Training accuracy and test accuracy.

Datasets

- Standard datasets from LIBSVM datasets.
- From small datasets to relatively large datasets.

Candidates and Configurations

ProxSARAH vs. others

- We implement 8 different variants of our ProxSARAH algorithm:
  - ProxSARAH-v1: $\gamma := \frac{\sqrt{2}}{L\sqrt{3m}}$, single sample (i.e., $\hat{b} = 1$), and $m := n$.
  - ProxSARAH-v2: $\gamma := 0.95$, mini-batch size $\hat{b} := \Theta\left(\sqrt{n}\right)$ and $m := \lfloor\sqrt{n}\rfloor$.
  - ProxSARAH-v3: $\gamma := 0.99$, mini-batch size $\hat{b} := \Theta\left(\sqrt{n}\right)$ and $m := \lfloor\sqrt{n}\rfloor$.
  - ProxSARAH-v4: $\gamma := 0.95$, mini-batch size $\hat{b} := \Theta\left(n^{1/3}\right)$ and $m := \lfloor n^{\frac{1}{3}}\rfloor$.
  - ProxSARAH-v5: $\gamma := 0.99$, mini-batch size $\hat{b} := \Theta\left(n^{1/3}\right)$ and $m := \lfloor n^{\frac{1}{3}}\rfloor$.
  - ProxSARAH-A-v1: $\hat{b} = 1$, and adaptive step-sizes.
  - ProxSARAH-A-v2: $\gamma_m := 0.99$ and mini-batch size $\hat{b} := \lfloor\sqrt{n}\rfloor$ and $m := \lfloor\sqrt{n}\rfloor$.
  - ProxSARAH-A-v3: $\gamma_m := 0.99$ and mini-batch size $\hat{b} := \lfloor n^{\frac{1}{3}}\rfloor$ and $m := \lfloor n^{\frac{1}{3}}\rfloor$.
- We also implement others algorithms: ProxSVRG, ProxSpiderBoost, ProxSGD, and ProxGD for comparison.

Nonnegative PCA (NN-PCA) Example

Problem formulation: A simple constrained nonconvex problem

$$F^\star := \min_{x\in\mathbb{R}^d}\left\{F(x) := -\frac{1}{2n}\sum_{i=1}^n x^\top(z_iz_i^\top)x \quad \text{s.t.}\ \ \|x\|\leq 1,\ x\geq 0\right\}.$$

Here, $f(x) = F(x)$ and ψ is the indicator function of $\{x : \|x\| \leq 1,\ x \geq 0\}$.

Datasets [4]

Some datasets	n	d
`mnist`	60,000	784
`rcv1-binary`	20,242	47,236
`real-sim`	72,309	20,958
`url_combined`	2,396,130	3,231,961
`news20.binary`	19,996	1,355,191
`avazu-app`	14,596,137	999,990

Table: Datasets used in the NN-PCA numerical example.

[4] available online at https://www.csie.ntu.edu.tw/∼cjlin/libsvm/

NN-PCA Example: Convergence Behavior on Small Datasets

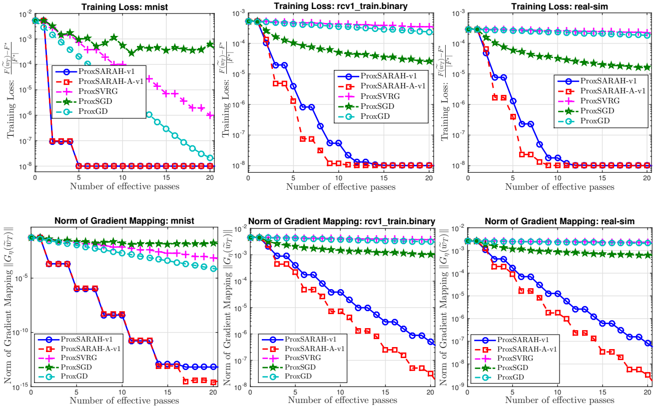

Figure: NN-PCA example with single sample on small and medium datasets.

Observation

- **ProxSARAH** variants outperform others.
- **Adaptive ProxSARAH variant** is better than fixed step-size variant.

NN-PCA Example: Convergence Behavior on Larger Datasets

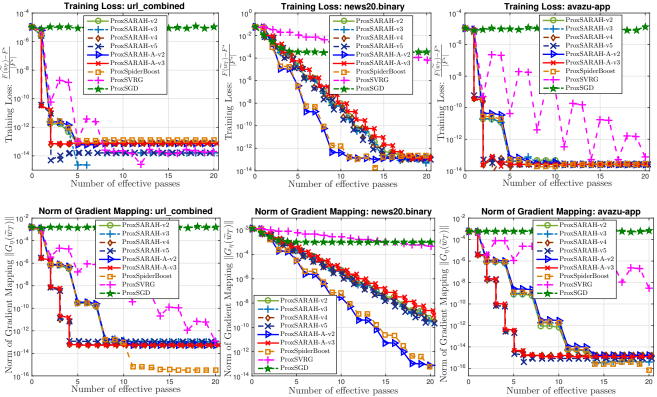

Figure: NN-PCA example with minibatch on large datasets.

A few remarks

- **ProxSpiderBoost** works well in some datasets.
- **ProxSARAH** variants still perform well in all cases.

Sparse Binary Classification with Nonconvex Losses

Problem formulation

$$\min_{x \in \mathbb{R}^d} \left\{ F(x) := \frac{1}{n} \sum_{i=1}^{n} \ell(a_i^\top x, b_i) + \lambda \|x\|_1 \right\}.$$

Some nonconvex and smooth losses

1. **Normalized sigmoid loss:**

$$\ell_1(s, \tau) := 1 - \tanh(\omega \tau s) \text{ for a given } \omega > 0.$$

2. **Nonconvex loss in 2-layer neural networks:**

$$\ell_2(s, \tau) := \left(1 - \frac{1}{1 + \exp(-\tau s)}\right)^2.$$

3. **Logistic difference loss:**

$$\ell_3(s, \tau) := \ln(1 + \exp(-\tau s)) - \ln(1 + \exp(-\tau s - \omega)) \text{ for some } \omega > 0.$$

Sparse Binary Classification with Nonconvex Losses - Single sample

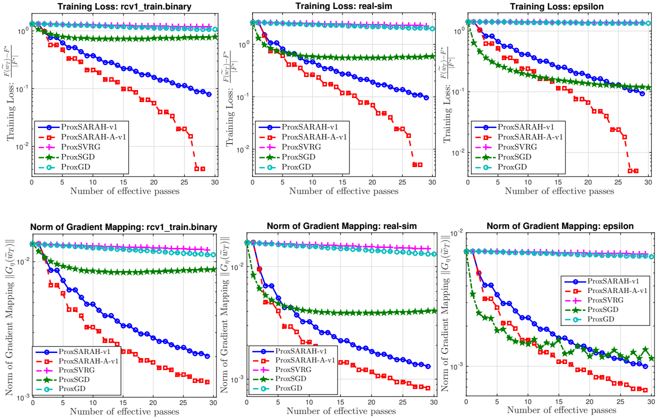

Figure: Example with single sample on small and medium datasets using loss ℓ_2.

Observation

- **ProxSARAH** variants still work best.
- **Adaptive ProxSARAH variant** outperforms fixed step-size variant.

Sparse Binary Classification with Nonconvex Losses - mini-batch

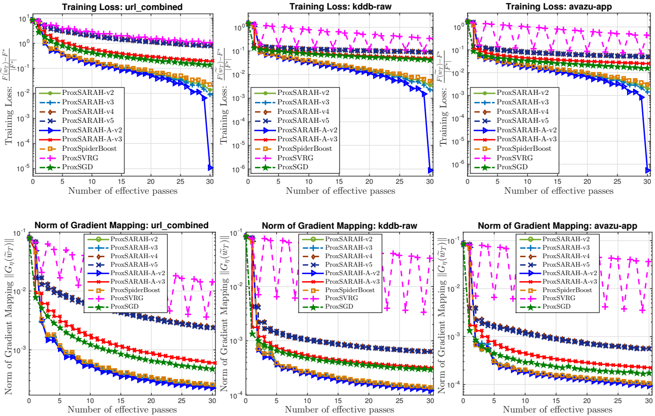

Figure: Example with mini-batch on large datasets using loss ℓ_2.

Observation

- **ProxSARAHs** and **ProxSpiderBoost** are comparable but are better than the rest.

Sparse Feedforward Neural Network Training

Problem formulation

$$\min_{x\in\mathbb{R}^d}\left\{F(x) := \frac{1}{n}\sum_{i=1}^{n}\ell\Big(h(x,a_i),b_i\Big) + \psi(x)\right\},$$

Here, we add an ℓ_1-norm regularizer to sparsify the weights.

Datasets and Network Architectures

Datasets	n	d
`mnist` [5]	60,000	10,000
`fashion_mnist` [6]	60,000	10,000

Table: Datasets used in neural network example.

1. **Test 1:** Fully-connected network $784 \times 100 \times 10$, ReLU activation, and soft-max cross-entropy loss.
2. **Test 2:** Fully-connected network $784 \times 800 \times 10$, ReLU activation, and soft-max cross-entropy loss.

[5] available at http://yann.lecun.com/exdb/mnist

[6] available at https://github.com/zalandoresearch/fashion-mnist

Sparse Feedforward Neural Network Training

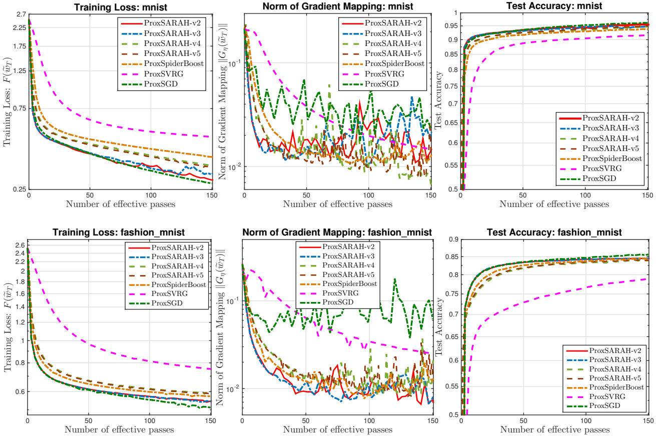

Figure: Fully-connected neural network $784 \times 100 \times 10$ on two datasets.

Observation

- **ProxSARAH** tends to compete with an adaptive **SGD** in this particular test.
- Norms of gradient mappings does not reflect test accuracy and training loss.

Sparse Feedforward Neural Network Training

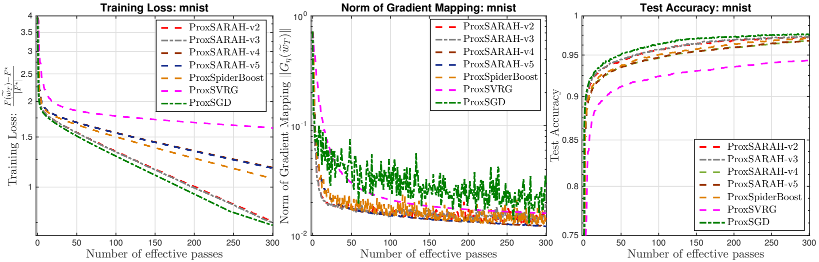

Figure: Fully-connected network $784 \times 800 \times 10$ on `mnist`.

Observation

- **Variance reduction methods** can achieve lower **norms of gradient mapping**.
- **ProxSARAH variants** perform better than other variance reduction methods.
- **ProxSGD** has good performance in terms of **training loss** and **test accuracy**.

Motivation

Motivation

Observation

- Both **SVRG** and **SARAH** are variance reduction methods, but have two loops, making them challenging to tune parameters.
- **SGD** often has good progress at early stage but oscillates at the end.
- **Variance reduction methods** are better at later stage.

Questions

- Can we combine both schemes to obtain a **trade-off**?
- Can we design single loop algorithms with better complexity than **SGD**?

$\Rightarrow$ **A hybrid stochastic optimization approach**

Key idea

Key idea

- Combining **SARAH estimator** and an **unbiased one** such as **SGD**:

$$v_t := \beta_t v_t^{\text{sarah}} + (1 - \beta_t) u_t^{\text{unbiased}},$$

  where $\beta_t \in [0, 1]$ is a given parameter that trades off between **bias** and **variance**.
- Apply ProxSARAH framework to solve (NCVX) and (ERM).

More details

- T.D., N. H. Pham, D. T. Phan, and L. M. Nguyen. A Hybrid Stochastic Optimization Framework for Stochastic Composite Nonconvex Optimization. Preprint: https://arxiv.org/pdf/1907.03793.pdf, 2019.

Summary and future research

Summary

- Seeking **first-order stationary points** of composite nonconvex optimization.
- New SARAH-based algorithms with flexible choices of parameters.
- Theoretical novelty
  - Convergence analysis in both single sample or mini-batch, finite-sum, or expectation cases.
  - **Optimal** or **best-known** convergence rates and complexity bounds in all cases.
  - A **new adaptive step-size scheme** which is updated in an increasing fashion.
- A new hybrid approach for stochastic optimization methods.

Possible future directions

- The hybrid idea can be extended to other stochastic estimators.
- Second-order stationary points (local minima, saddle-points).
- Applications to other problems and algorithmic variants.

Thank you!

Check out nhanph.github.io for more information.

References I

[1] S. Ghadimi and G. Lan.
Stochastic first-and zeroth-order methods for nonconvex stochastic programming.
SIAM J. Optim., 23(4):2341–2368, 2013.

[2] R. Johnson and T. Zhang.
Accelerating stochastic gradient descent using predictive variance reduction.
In *Advances in Neural Information Processing Systems (NIPS)*, pages 315–323, 2013.

[3] A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro.
Robust stochastic approximation approach to stochastic programming.
SIAM J. Optim., 19(4):1574–1609, 2009.

[4] L. M. Nguyen, J. Liu, K. Scheinberg, and M. Takac.
SARAH: A novel method for machine learning problems using stochastic recursive gradient.
In *The 34th International Conference on Machine Learning (ICML)*, 2017.

[5] H. Robbins and S. Monro.
A stochastic approximation method.
The Annals of Mathematical Statistics, 22(3):400–407, 1951.