A One-Sample Decentralized Proximal Algorithm for Non-Convex Stochastic Composite Optimization

Tesi Xiao[1,*], Xuxing Chen[2,*], Krishnakumar Balasubramanian[1], Saeed Ghadimi[3]

[1]Department of Statistics, [2]Department of Mathematics, University of California, Davis

[3]Department of Management Sciences, University of Waterloo

[*] Equal Contribution

Introduction

We consider the following decentralized composite optimization problem:

$$\min_{x\in\mathbb{R}^d} \quad \Phi(x) := F(x) + \underbrace{\Psi(x)}_{\text{cvx, shared}}, \quad F(x) := \frac{1}{n}\sum_{i=1}^{n} \underbrace{F_i(x)}_{\text{ncvx, known to agent } i},$$

where each function $F_i(x)$ is a smooth function only known to the agent i; $\Psi(x)$ is non-smooth, convex, and shared across all agents.

Our goal is to design *efficient stochastic one-sample* algorithms for solving the above problem, given access to *noisy evaluations* of ∇F_i's and F_i's on agent i.

Algorithm 1: `Prox-DASA(-GT)`

Input: $x_i^0 = z_i^0 = u_i^0 = \mathbf{0}, \gamma, \{\alpha_k\}_{\geq 0}, m$

for $k = 0, 1, \ldots, K$ **do**

`# Local Update`

for $i = 1, 2, \ldots, n$ *(in parallel)* **do**

$\tilde{x}_i^{k+1} = (1-\alpha_k)x_i^k + \alpha_k \,\mathbf{prox}_{\Psi}^{\gamma}\left(x_i^k - \gamma z_i^k\right)$

`# Compute stochastic gradient` $v_i^{k+1} = \nabla G_i(x_i^k, \xi_i^{k+1})$

$$\tilde{u}_i^{k+1} = \begin{cases} v_i^{k+1} & \texttt{Prox-DASA} \\ \boxed{u_i^k + v_i^{k+1} - v_i^k} \to \text{Gradient tracking} & \texttt{Prox-DASA-GT} \end{cases}$$

$\tilde{z}_i^{k+1} = (1-\alpha_k)z_i^k + \alpha_k \tilde{u}_i^{k+1}$

end

`# Communication`

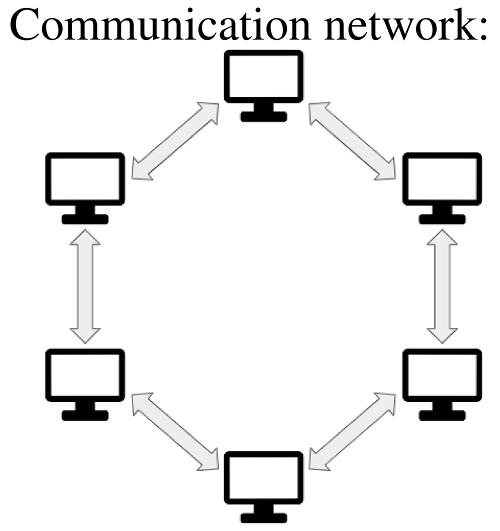
$[x_1^{k+1}, \ldots, x_n^{k+1}] = [\tilde{x}_1^{k+1}, \ldots, \tilde{x}_n^{k+1}]\mathbf{W}^m$

$[z_1^{k+1}, \ldots, z_n^{k+1}] = [\tilde{z}_1^{k+1}, \ldots, \tilde{z}_n^{k+1}]\mathbf{W}^m$

$\boxed{[u_1^{k+1}, \ldots, u_n^{k+1}] = [\tilde{u}_1^{k+1}, \ldots, \tilde{u}_n^{k+1}]\mathbf{W}^m} \to$ Communication of GT variable

end

Communication network:



(credit to Vecteezy.com)

Weight matrix $\mathbf{W}$:

$$\begin{pmatrix} 1/3 & 1/3 & 0 & 0 & 0 & 1/3 \\ 1/3 & 1/3 & 1/3 & 0 & 0 & 0 \\ 0 & 1/3 & 1/3 & 1/3 & 0 & 0 \\ 0 & 0 & 1/3 & 1/3 & 1/3 & 0 \\ 0 & 0 & 0 & 1/3 & 1/3 & 1/3 \\ 1/3 & 0 & 0 & 0 & 1/3 & 1/3 \end{pmatrix}$$

$\rho = 2/3$

Remarks

- **Accelerated Consensus**: $\mathbf{W}^m$ can be replaced with a Chebyshev-type polynomial of $\mathbf{W}$ to improve the dependency on ρ: $\frac{1}{1-\rho} \to \frac{1}{\sqrt{1-\rho}}$.
- **Why one-sample algorithm?** To reduce per-iteration cost and memory usage, improve generalization, and bridge the gap between theory and practice.

Theoretical Results

Measure of Non-stationarity

- Gradient Mapping (GM): $\mathcal{G}(x, z, \gamma) = \frac{1}{\gamma}\left(x - \mathbf{prox}_{\Psi}^{\gamma}(x - \gamma z)\right)$.
- Random vectors $\mathbf{X} = [x_1, \ldots, x_n]$ generated by an algorithm is an ϵ-stationary point in terms of GM, if we have

$$\text{(stationarity violation)} \qquad \mathbb{E}\left[\left\|\mathcal{G}(\bar{x}, \nabla F(\bar{x}), \gamma)\right\|^2\right] \leq \epsilon,$$

$$\text{(consensus error)} \qquad \mathbb{E}\left[\frac{L_{\nabla F}^2}{n}\left\|\mathbf{X} - \bar{\mathbf{X}}\right\|^2\right] \leq \epsilon.$$

Assumptions

- (Network topology) $\mathbf{W} = (w_{ij}) \in \mathbb{R}^{n\times n}$ is symmetric and doubly stochastic, i.e., $w_{ij} \geq 0, \mathbf{W}^\top = \mathbf{W}, \mathbf{W}\mathbf{1} = \mathbf{1}, \mathbf{1}^\top\mathbf{W} = \mathbf{1}^\top$. and its eigenvalues satisfy $1 = \lambda_1 > \lambda_2 \geq \cdots \geq \lambda_n$ and $\rho := \max\{|\lambda_2|, |\lambda_n|\} < 1$.
- (Smoothness) All functions $\{F_i\}_{1\leq i\leq n}$ have Lipschitz continuous gradients.
- The function $\Psi : \mathbb{R}^d \to \mathbb{R}\cup\{+\infty\}$ is a closed proper convex function.
- (Stochastic oracles) All stochastic gradients are unbiased with bounded variance. Different stochastic gradients are independent.
- (Bounded heterogeneity for `Prox-DASA`) There exists a constant $\nu \geq 0$ such that for all $1 \leq i \leq n, x \in \mathbb{R}^d$, $\|\nabla F_i(x) - \nabla F(x)\| \leq \nu$.

Main Results

Suppose the total number of iterations $K \geq K_0$, $\alpha_k \asymp \sqrt{\frac{n}{K}}, \gamma \asymp \frac{1}{L_{\nabla F}}$.
Let C_0 be an initialization-dependent constant and $R \sim \mathrm{Unif}\{1, 2, \ldots, K\}$.

(Prox-DASA) For Algorithm 1 we have

$$\overbrace{\underbrace{\boxed{\mathbb{E}\left[\left\|\mathcal{G}(\bar{x}^R, \nabla F(\bar{x}^R), \gamma)\right\|^2\right]}}_{\textit{stationarity violation}}}^{\mathbb{E}\left[\left\|\bar{z}^R - \nabla F(\bar{x}^R))\right\|^2\right] \text{ (VR property)} \;\lesssim} \lesssim \underbrace{\boxed{\frac{L_{\nabla F}C_0 + \sigma^2}{\sqrt{nK}}}}_{\textit{centralized convergence}} + \overbrace{\underbrace{\boxed{\frac{n(\sigma^2 + L_{\nabla F}^2\nu^2)\rho^{2m}}{K}}}_{\textit{consensus error}}}^{\mathbb{E}\left[\frac{L_{\nabla F}^2}{n}\left\|\mathbf{X}_R - \bar{\mathbf{X}}_R\right\|^2 + \frac{1}{n}\left\|\mathbf{Z}_R - \bar{\mathbf{Z}}_R\right\|^2\right] \;\lesssim}.$$

(Prox-DASA-GT) Similar results hold for Algorithm 1 with **GT** when $\nu = \infty$.

Complexity. For any $\epsilon > 0$, the sample complexity per agent for finding ϵ-stationary points are $\mathcal{O}(\max\{n^{-1}\epsilon^{-2}, K_T\})$ where K_T is the transient time.

- $m = 1$ yields a topology-independent communication complexity $\mathcal{O}(n^{-1}\epsilon^{-2})$ with transient time K_T depending on ρ.
- $m \asymp \lceil\frac{1}{1-\rho}\rceil$ (or $m \asymp \lceil\frac{1}{\sqrt{1-\rho}}\rceil$ for accelerated consensus algorithms) results in a topology-independent transient time.

Comparisons

Algorithm	Batch Size	Sample Complexity	Communication Complexity	Linear Speedup?	Remark
`ProxGT-SA`	$\mathcal{O}(\epsilon^{-1})$	$\mathcal{O}(n^{-1}\epsilon^{-2})$	$\mathcal{O}(\log(n)\epsilon^{-1})$	✓	
`ProxGT-SR-O`	$\mathcal{O}(\epsilon^{-1})$	$\mathcal{O}(n^{-1}\epsilon^{-1.5})$	$\mathcal{O}(\log(n)\epsilon^{-1})$	✓	SVRG: (i) double-loop; (ii) mean-squared smoothness
`DEEPSTORM`	$\mathcal{O}(\epsilon^{-0.5})$ then $\mathcal{O}(1)^*$	$\mathcal{O}(n^{-1}\epsilon^{-1.5})$	$\mathcal{O}(n^{-1}\epsilon^{-1.5})$	✓	STORM: (i) two time-scale; (ii) mean-squared smoothness; (iii) two gradient evaluations per iter.
	$\mathcal{O}(1)$	$\mathcal{O}(\epsilon^{-1.5}\lvert\log\epsilon\rvert^{-1.5})$	$\mathcal{O}(\epsilon^{-1.5}\lvert\log\epsilon\rvert^{-1.5})$	✗	
Prox-DASA	$\mathcal{O}(1)$	$\mathcal{O}(n^{-1}\epsilon^{-2})$	$\mathcal{O}(n^{-1}\epsilon^{-2})$	✓	bounded heterogeneity
Prox-DASA-GT	$\mathcal{O}(1)$	$\mathcal{O}(n^{-1}\epsilon^{-2})$	$\mathcal{O}(n^{-1}\epsilon^{-2})$	✓	

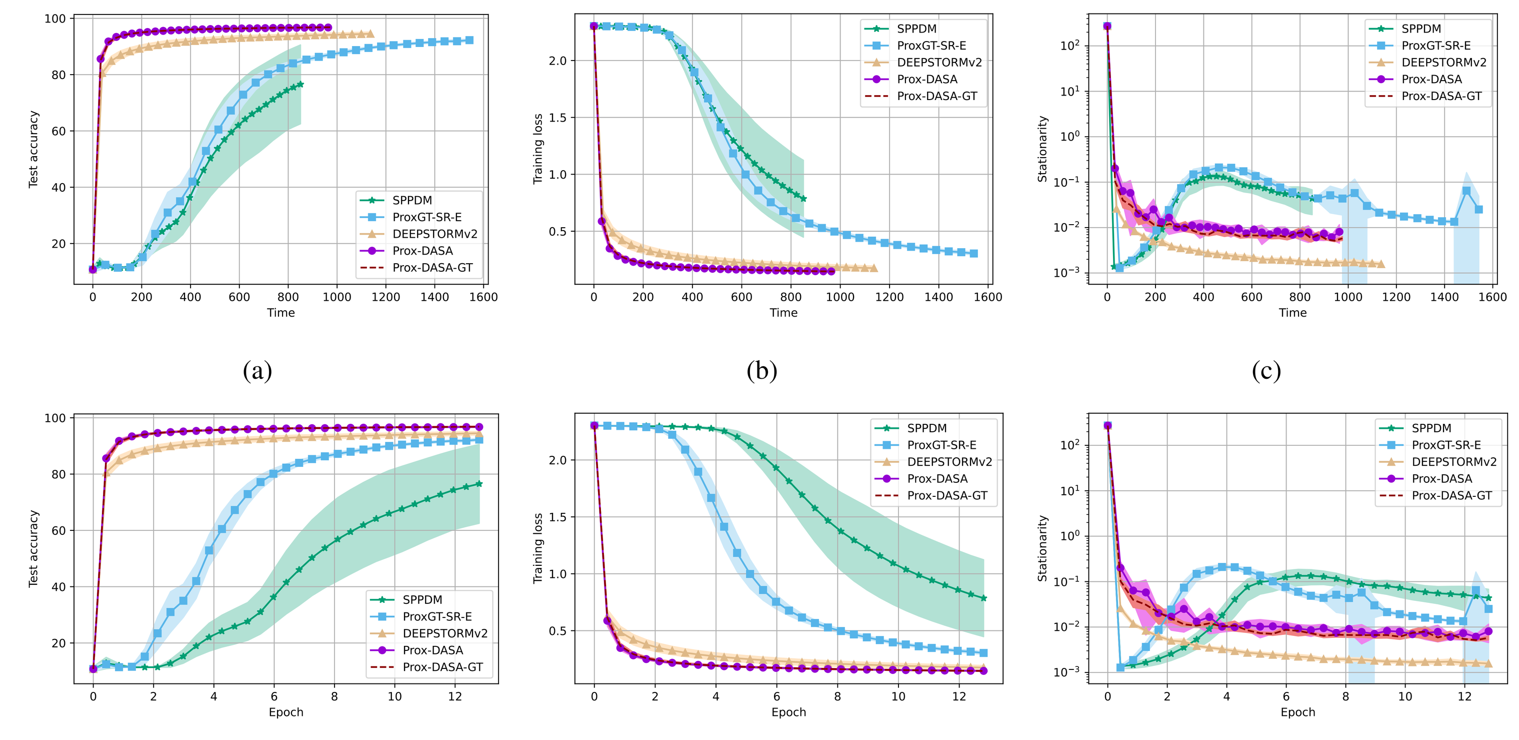
* It requires $\mathcal{O}(\epsilon^{-0.5})$ batch size in the first iteration and then $\mathcal{O}(1)$ for the rest.

Experimental Results

Decentralized traning sparse neural networks for classification tasks on MNIST:

$$\min_{\theta\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^{n}\frac{1}{|\mathcal{D}_i|}\sum_{(x,y)\in\mathcal{D}_i}\ell_i(f(x;\theta), y) + \lambda\|\theta\|_1,$$

Faster and more stable training using small batch sizes!



(a) (b) (c)

(d) (e) (f)

Contributions and Takeaway

Existing works have several drawbacks: **increasing batch sizes**, **algorithmic complexities**, and **theoretical weakness**.

Our algorithms achieve **linear speedup** with $\mathcal{O}(1)$ **batch size** under **mild assumptions** without using complicated variance reduction (VR) techniques.

Moving-Average update is all you need!