Distributed stochastic nonconvex optimization: Optimal regimes and tradeoffs

Usman A. Khan
Electrical and Computer Engineering, Tufts University

ZJU-CSE Summer School 2021
Zhejiang University, China
August 09, 2021

Acknowledgments

Reza D.
(2011-15)

C. Xi
(2012-17)

S. Safavi
(2013-17)

F. Saadatniaki
(2014-19)

R. Xin
(2016-)

M. I. Qureshi
(2018-)

A. Swar
(2020-)

H. Raja
(2021-)

Research Overview

Learning from Data

- Data is everywhere and holds a significant potential
  - Image classification, Medical diagnosis, Credit card fraud, ...

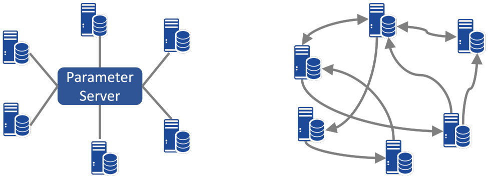


Figure 1: Centralized and distributed learning architectures

- Collecting all data at a central location may not be practical
  - Large, private, datasets with communication constraints
- Distributed methods rely on local processing and communication

A simple case study . . .

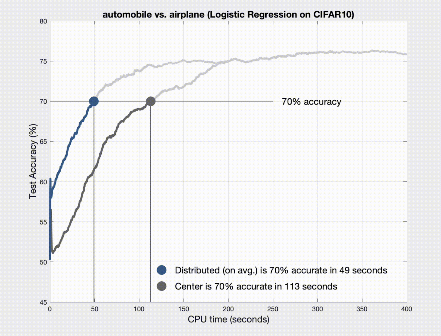


Figure 2: Test accuracy of a model trained with 10,000 32×32 pixel images

- When do distributed methods outperform their centralized analogs?
- How do we formally quantify such a comparison?

Some Preliminaries

Example: Recognizing Traffic Signs

- Identify STOP vs. YIELD sign

Figure 3: Binary classification: (Left) Training phase (Right) Testing phase

- Input data: images $\{\boldsymbol{\theta}_j\}$ and their labels $\{y_j\}$
- Model: A classifier $\mathbf{x}$ that predicts a label $\widehat{y}_j$ for each image $\boldsymbol{\theta}_j$
  - Changing $\mathbf{x}$ changes the predicted label $\widehat{y}_j(\mathbf{x}; \boldsymbol{\theta}_j)$

- Pick a classifier $\mathbf{x}^*$ that minimizes *some* loss over all images

$$\mathbf{x}^* = \underset{\mathbf{x}\in\mathbb{R}^p}{\text{argmin}} \sum_j \ell(y_j,\ \widehat{y}_j(\mathbf{x}; \boldsymbol{\theta}_j))$$

Minimizing Functions

$$\min_{\mathbf{x}} f(\mathbf{x}), \qquad f := \sum_j \ell(y_j,\ \widehat{y}_j(\mathbf{x};\boldsymbol{\theta}_j)) : \mathbb{R}^p \to \mathbb{R}$$

- Different predictors $\widehat{y}$ and losses ℓ lead to different cost functions f

- **Quadratic**: Signal estimation, linear regression, LQR
- **(Strongly) convex**: Logistic regression, classification
- **Nonconvex**: Neural networks, reinforcement learning, blind sensing

- *This talk*
- First-order (gradient-based) methods over various function classes
  - Search for a point $\mathbf{x}^* \in \mathbb{R}^p$ such that $\nabla f(\mathbf{x}^*) = \mathbf{0}_p$
  - When the training data is distributed over a network of nodes (machines, devices, robots)

Basic Definitions

- $f : \mathbb{R}^p \to \mathbb{R}$ is L-smooth and $f(\mathbf{x}) \geq f^* \geq -\infty, \forall \mathbf{x}$
  - Not necessarily convex, bounded above by a quadratic
  - Assumed throughout

- $f : \mathbb{R}^p \to \mathbb{R}$ is convex (lies above all of its tangents)

- f is μ-strongly-convex (convex and bounded below by a quadratic)
  - For SC functions, we have $\kappa := {}^{L}/_{\mu} \geq 1$

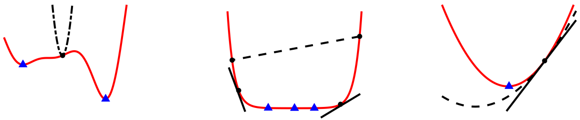


Figure 4: Nonconvex: $\sin(ax)(x + bx^2)$. Convexity. Strong Convexity.

Smooth function classes

- Minimizing smooth (differentiable) functions $f : \mathbb{R}^p \to \mathbb{R}$
  - Search for a *stationary point* $\mathbf{x}^* \in \mathbb{R}^p$, i.e., $\nabla f(\mathbf{x}^*) = \mathbf{0}_p$

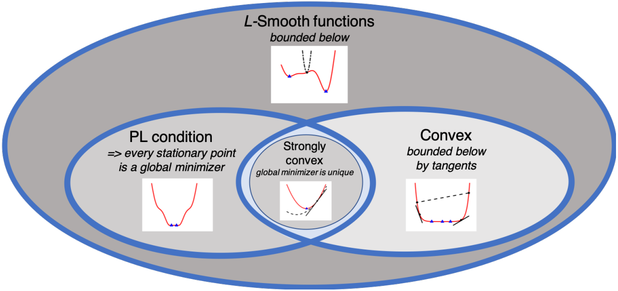


Figure 5: Function classes restricted to L-smooth functions

- Nonconvex: $\mathbf{x}^*$ may be a minimum, a maximum, or a saddle point
- Convex (and PL) functions: $f(\mathbf{x}^*)$ is the unique global minimum
- Strongly convex functions: $\mathbf{x}^*$ is the unique global minimizer

First-order methods (Gradient Descent)

$$\min_{\mathbf{x} \in \mathbb{R}^p} f(\mathbf{x})$$

- Search for a **stationary point** $\mathbf{x}^*$, i.e., $\nabla f(\mathbf{x}^*) = \mathbf{0}_p$
- Intuition: Take a step in the direction opposite to the gradient
  - At $\star$, $\nabla f(\mathbf{x}^*) = \mathbf{0}_p$

Figure 6: Minimizing strongly convex functions: $\mathbb{R} \to \mathbb{R}$ and $\mathbb{R}^2 \to \mathbb{R}$

- **Gradient Descent**: $\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha \cdot \nabla f(\mathbf{x}_k)$

Function classes: Performance metrics and Rates

- Gradient Descent: $\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha \cdot \nabla f(\mathbf{x}_k)$

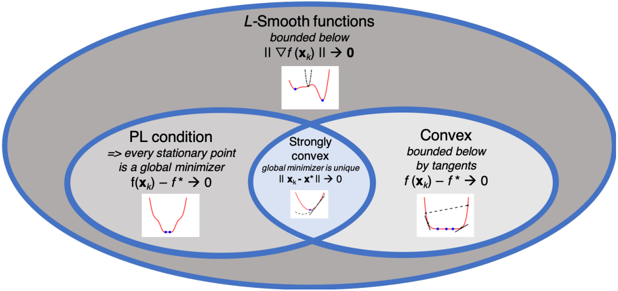


Figure 7: Function classes restricted to L-smooth functions

- Convergence rates of GD (non-stochastic and not accelerated):
  - Nonconvex: $||\nabla f(\mathbf{x}_k)|| \to 0$ at $\mathcal{O}(1/\sqrt{k})$
  - Convex: $f(\mathbf{x}_k) - f(\mathbf{x}^*) \to 0$ at $\mathcal{O}(1/k)$
  - SC (and PL): $f(\mathbf{x}_k) - f(\mathbf{x}^*) \to 0$ and $\|\mathbf{x}_k - \mathbf{x}^*\| \to 0$ exponentially (linearly on the log-scale)

How to extend GD when the data is distributed?

- Let's consider a simple example: Linear Regression
- Implement **local GD** at each node i: $\mathbf{x}_{k+1}^{i} = \mathbf{x}_{k}^{i} - \alpha \cdot \nabla f_i(\mathbf{x}_{k}^{i})$

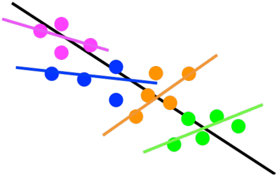


Figure 8: Linear regression: Locally optimal solutions

- Local GD does not lead to agreement on the optimal solution
- Requirements for a distributed algorithm
  - Agreement: Each node agrees on the same solution
  - Optimality: The agreed upon solution is the optimal

Distributed optimization

Smooth and strongly convex problems with full gradients

Distributed Optimization

$$\min_{\mathbf{x}\in\mathbb{R}^p} F(\mathbf{x}), \qquad F(\mathbf{x}) := \sum_{i=1}^{n} f_i(\mathbf{x})$$

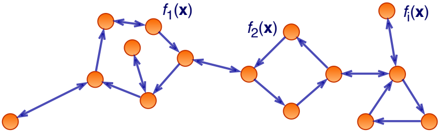


Figure 9: A peer-to-peer or edge computing architecture

Assumptions

- Each f_i is private to node i
- Each f_i is L_i-smooth and μ_i-strongly-convex (*assumed for now!*)
- The nodes communicate over a network (a connected graph)

- F has a unique global minimizer $\mathbf{x}^*$ such that $\nabla F(\mathbf{x}^*) = \mathbf{0}_p$

Distributed Gradient Descent (DGD)

$$\mathbf{x}_{k+1}^i = \sum_{r=1}^{n} w_{ir} \cdot \mathbf{x}_k^r - \alpha \cdot \nabla f_i(\mathbf{x}_k^i)$$

- Mix and Descend [Nedić et al. '09]
  - The weight matrix $W = \{w_{ij}\}_{\geq 0}$ sums to 1 on rows and columns
  - DGD converges linearly (on a log-scale) up to a steady-state error
  - Exact convergence with a decaying step-size but at a sublinear rate

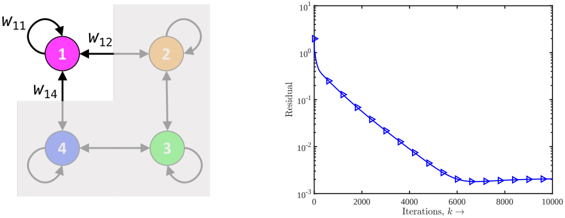


Figure 10: (Left) An undirected graph. (Right) DGD performance.

Recap

- GD and Distributed GD

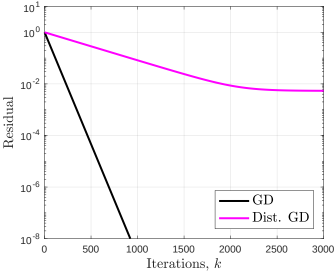


Figure 11: Performance for smooth and strongly convex problems

- How do we remove the steady-state error in DGD?

Distributed Gradient Descent with Gradient Tracking

GT-DGD: Intuition

- Problem: $\min_{\mathbf{x}} \sum_i f_i(\mathbf{x})$, i.e., search for $\mathbf{x}^*$ such that $\sum_i \nabla f_i(\mathbf{x}^*) = \mathbf{0}_p$

- DGD does not reach $\mathbf{x}^*$ because $\mathbf{x}^*$ is not its fixed point

$$\begin{aligned} \mathbf{x}_{k+1}^i &= \sum_{r=1}^n w_{ir} \cdot \mathbf{x}_k^r - \alpha \cdot \nabla f_i(\mathbf{x}_k^i) \\ \mathbf{x}^* &\neq \quad 1 \cdot \mathbf{x}^* - \alpha \cdot \nabla f_i(\mathbf{x}^*) \end{aligned}$$

- This is because $\nabla f_i(\mathbf{x}^*) \neq 0$ but only the sum gradient is
- We call this the local-vs.-global dissimilarity bias ($\eta \cong \|\nabla f_i - \nabla F\|$)

- Fix: Replace $\nabla f_i(\mathbf{x}_k^i)$ with $\mathbf{y}_k^i$ that **tracks** the global gradient ∇F

$$\mathbf{x}_{k+1}^i = \sum_{r=1}^n w_{ir} \cdot \mathbf{x}_k^r - \alpha \cdot \mathbf{y}_k^i$$

- Linear convergence in distributed optimization (SSC)
  - Undirected graphs: [Xu et al. '15], [Lorenzo et al. '15]
  - Directed graphs: [Xi-Khan '15], [Xi-Xin-Khan '16,'17], [Xin-Khan '18]

AB Algorithm

- Problem: $\min_{\mathbf{x}} \sum_i f_i(\mathbf{x})$
- DGD: $\mathbf{x}_{k+1}^i = \sum_{r=1}^n w_{ir} \cdot \mathbf{x}_k^r - \alpha \cdot \nabla f_i(\mathbf{x}_k^i)$

Algorithm 1 [Xin-Khan '18]: at each node i

Data: $\mathbf{x}_0^i \in \mathbb{R}^p$; $\alpha > 0$; $\{a_{ir}\}_{r=1}^n$; $\{b_{ir}\}_{r=1}^n$; $\mathbf{y}_0^i = \nabla f_i(\mathbf{x}_0^i)$

for $k = 0, 1, \dots,$ **do**

$$\mathbf{x}_{k+1}^i = \sum_{r=1}^n a_{ir} \cdot \mathbf{x}_k^r - \alpha \cdot \mathbf{y}_k^r$$

$$\mathbf{y}_{k+1}^i = \sum_{r=1}^n b_{ir} \cdot \mathbf{y}_k^r + \nabla f_i(\mathbf{x}_{k+1}^i) - \nabla f_i(\mathbf{x}_k^i)$$

end

- AB converges linearly to $\mathbf{x}^*$ with the help of **Gradient Tracking**
  - Over both directed and undirected graphs
- We can further add heavy-ball or Nesterov momentum

AB: Results (Smooth and Strongly convex)

- Linear convergence of AB over both directed and undirected graphs
  - [Xin-Khan '18]: For a range of step-sizes $\alpha \in (0, \bar{\alpha}]$
  - [Xin-Khan '18]: For non-identical step-sizes α_i's at the nodes
  - [Pu et al. '18]: Over mean-connected graphs
  - [Saadatniaki-Xin-Khan '18]: Over time-varying random graphs
  - Asynchronous, delays, nonconvex analysis (but without explicit rates)
- Condition number dependence
  - GD κ, AB undirected $\kappa^{5/4}$, AB directed κ^2
- AB with heavy-ball momentum
  - [Xin-Khan '18]: Linear convergence for a range of alg. parameters
  - *Acceleration is not proved analytically and remains an open problem*
- AB with Nesterov momentum
  - [Qu et al. '18]: Undirected graphs $\kappa^{5/7}$
  - [Xin-Jakovetić-Khan '19]: Convergence and acceleration are shown numerically over directed graphs
  - *Directed graphs: Convergence and acceleration are both open*

Performance comparison

- GD, HB, DGD, AB, ABm

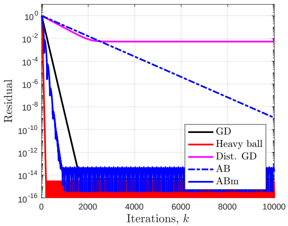


Figure 12: Performance for smooth and strongly convex problems, $\kappa = 100$

- Addition of gradient tracking recovers linear convergence (proved)
- Acceleration can be shown numerically but it is not proved (yet!)

- What happens when the gradients are imperfect?

Distributed Stochastic Optimization

- Stochastic gradients with noise variance ν^2

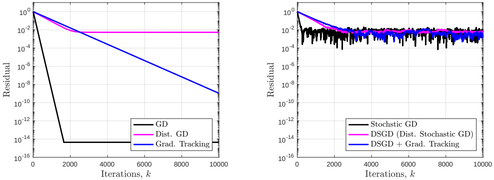


Figure 13: Full gradients ($\nu^2 = 0$) vs. stochastic gradients

- **DSGD**: Residual decays **linearly** to an error ball [Yuan et al. '19]

$$\limsup_{k\to\infty} \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}[\|\mathbf{x}_k^i - \mathbf{x}^*\|_2^2] = \mathcal{O}\Big(\frac{\alpha}{n\mu}\nu^2 + \frac{\alpha^2\kappa^2}{1-\lambda}\nu^2 + \frac{\alpha^2\kappa^2}{(1-\lambda)^2}\eta\Big),$$

where η quantifies the local-vs.-global dissimilarity bias

- **Gradient tracking eliminates η but the variance remains**

Distributed Stochastic Optimization

Nonconvex problems

Distributed Stochastic Optimization: Measurement Model

$$\min_{\mathbf{x}} F(\mathbf{x}), \qquad F(\mathbf{x}) := \sum_{i=1}^{n} f_i(\mathbf{x}), \quad f_i : \mathbb{R}^p \to \mathbb{R}$$

- *Online/Streaming*: Given some $\mathbf{x} \in \mathbb{R}^p$, each node i makes a noisy measurement of the local gradient $\nabla f_i(\mathbf{x})$
- *Offline/Batch*: Each node i possesses a local dataset with m_i data points and their corresponding labels, i.e., $\nabla f_i(\mathbf{x}) = \sum_{j=1}^{m_i} \nabla f_{i,j}(\mathbf{x})$

Figure 14: (Left) Online streaming data (Right) Offline batch data

Distributed Stochastic Optimization: Communication Model

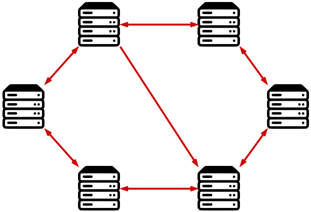


Figure 15: Data Center

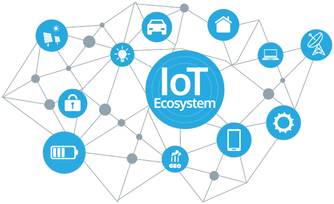


Figure 16: Internet of Things

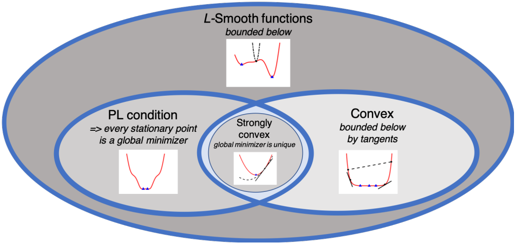
- Controllable topology
- $\#$ nodes $\ll$ $\#$ local samples
- Big-data regime

- Ad hoc topology
- $\#$ local samples is small
- IoT regime

Distributed Stochastic Optimization

- Gradient tracking eliminates η (the local-vs.-global dissimilarity bias) but the variance ν^2 remains

- Can we quantify the improvement due to gradient tracking?
- Can we eliminate the steady-state error due to the variance?
- What can we say about different function classes?



Batch problems: The GT+VR framework

GT+VR framework

- Each node i possesses a local batch of m_i data samples
  - The local cost f_i is the sum over all data samples $\sum_{j=1}^{m_i} f_{i,j}$

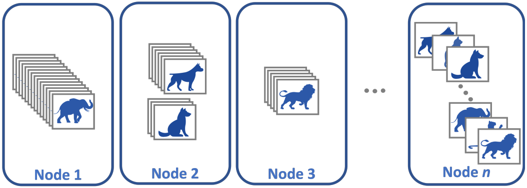


Figure 17: Arbitrary data distribution over the network

- Local Gradient computation $\sum_{j=1}^{m_i} \nabla f_{i,j}$ is prohibitively expensive
- Traditionally: $\mathbf{x}_{k+1}^i = \sum_r w_{ir} \cdot \mathbf{x}_k^r - \alpha \cdot \nabla f_{i,\tau}(\mathbf{x}_k^i)$
  - Performance is impacted due to sampling and local vs. global bias

GT+VR framework

- The GT+VR framework: From $\nabla f_{i,\tau}$ to $\nabla F = \sum_{i=1}^{n} \sum_{j=1}^{m_i} \nabla f_{i,j}$
  - Local variance reduction: **Sample** then **Estimate**

$$\nabla f_{i,\tau} \to \nabla f_i = \sum_{j=1}^{m_i} \nabla f_{i,j}$$

  - Global gradient tracking: **Fuse** the estimates over the network

$$\nabla f_i \to \nabla F = \sum_{i=1}^{n} \nabla f_i$$

- Popular VR methods: SAG, SAGA, SVRG, SPIDER, SARAH
- Our work[1]: GT-SAGA, GT-SVRG, GT-SARAH

1. R. Xin, S. Kar, and U. A. Khan, "Gradient tracking and variance reduction for decentralized optimization and machine learning," IEEE Signal Processing Magazine, 37(3), pp. 102-113, May 2020.

GT-SAGA

- GT-SAGA: Requires $\mathcal{O}(m_i p)$ storage at each node

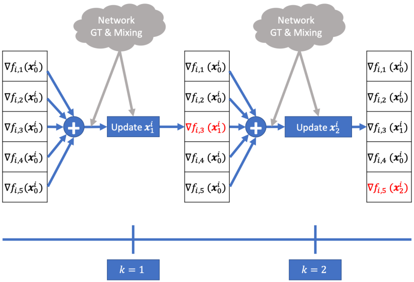
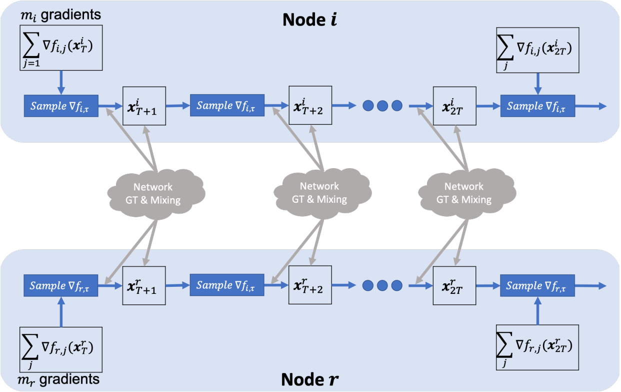


Figure 18: GT-SAGA at node i

- [Xin-Kar-Khan: May '20, Xin-Khan-Kar: Nov. '20]
  - Strongly convex problems: Linear convergence, improved rates
  - Linear speedup and network-independent convergence for both nonconvex and nonconvex with PL

GT-SARAH

- GT-SARAH (StochAstic Recursive grAdient algoritHm)
  - No storage but additional network synchrony when $m_i \neq m_r$



GT-SAGA vs. GT-SARAH

- A space vs. time tradeoff: Storage vs. Synchronization

- GT-SAGA: For ad hoc problems with heterogeneous data
- GT-SARAH: For very large-scale problem in controlled settings

- We can show[1,2] these tradeoffs theoretically!!!

1. R. Xin, U. A. Khan, and S. Kar, "A fast randomized incremental gradient method for non-convex decentralized stochastic optimization," Oct. 2020, arxiv: 2011.03853.
2. R. Xin, U. A. Khan, and S. Kar, "A near-optimal stochastic gradient method for decentralized non-convex finite-sum optimization," Aug. 2020, arxiv: 2008.07428.

GT-SARAH: Smooth and nonconvex

- GT plus SARAH based VR
  - Assume $m_i = m, \forall i$, for simplicity

Theorem (Almost sure and mean-squared results, Xin-Khan-Kar '20)

At each node i, GT-SARAH's iterate $\mathbf{x}_k^i$ follows

$$\mathbb{P}\left(\lim_{k\to\infty} \|\nabla F(\mathbf{x}_k^i)\| = 0\right) = 1 \qquad \textit{and} \qquad \lim_{k\to\infty} \mathbb{E}\left[\left\|\nabla F(\mathbf{x}_k^i)\right\|^2\right] = 0.$$

GT-SARAH: Smooth and nonconvex

$$\min_{\mathbf{x}} \sum_{i=1}^{n} \sum_{j=1}^{m} f_{i,j}(\mathbf{x})$$

- Total of $N = nm$ data points divided equally among n nodes
- How many gradient computations are required to reach an ϵ-accurate solution?

Theorem (Gradient computation complexity, Xin-Khan-Kar '20)

Under a certain constant step-size α, GT-SARAH, with $\mathcal{O}(m)$ inner loop iterations, reaches an ϵ-optimal stationary point of the global cost F in

$$\mathcal{H} := \mathcal{O}\left(\max\left\{N^{1/2}, \tfrac{n}{(1-\lambda)^2}, \tfrac{(n+m)^{1/3} n^{2/3}}{1-\lambda}\right\}\left(c \cdot L + \tfrac{1}{n}\textstyle\sum_{i=1}^{n} \|\nabla f_i(\overline{\mathbf{x}}_0)\|^2\right)\frac{1}{\epsilon}\right)$$

gradient computations across all nodes, where $c := F\left(\overline{\mathbf{x}}_0\right) - F^*$.

GT-SARAH: Smooth and nonconvex

$$\min_{\mathbf{x}} \sum_{i=1}^{n} \sum_{j=1}^{m} f_{i,j}(\mathbf{x})$$

- Total of $N = nm$ data points divided equally among n nodes
- How many gradient computations are required to reach an ϵ-accurate solution?
- In a certain big-data regime $n \leq \mathcal{O}(m(1-\lambda)^6)$: $\mathcal{H} = \mathcal{O}(N^{1/2}\epsilon^{-1})$
  - Independent of the network topology
  - Linear speedup compared to centralized SARAH

GT-SARAH: Smooth and nonconvex

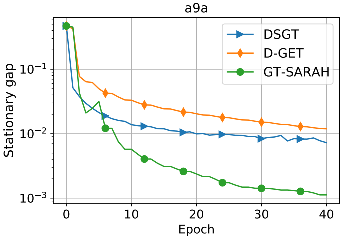
- Minimize a sum of $N := nm$ smooth nonconvex functions
- The rate $O(N^{1/2}\epsilon^{-1})$ in the big-data regime matches the centralized algorithmic lower bound for this problem class [SPIDER: Fang et al. '18]

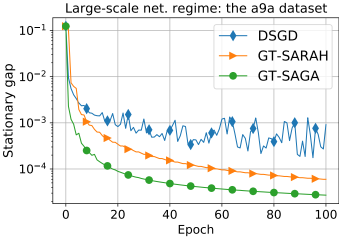
- Independent of the variance of local gradient estimators
- Independent of the local vs. global dissimilarity bias
- Independent of the network

- Linear speedup
  GT-SARAH is n times faster than the centralized SARAH

Experiments: Nonconvex binary classification

- Performance Comparison



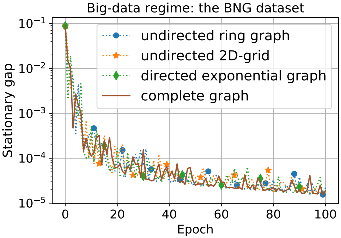


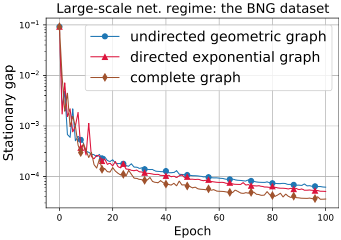
- Big-data regime
- 10×10 grid graph

- IoT regime
- Nearest neighbor graph

Experiments: Nonconvex binary classification

- Effect of network topology in GT-SAGA





- Big-data regime
- IoT regime

Online Stochastic Nonconvex Problems

- What happens for streaming data where VR is not applicable?

- GT-DSGD[1]: Vanilla distributed SGD + GT
- Decaying stepsizes can be used to kill the variance

- GT-HSGD[2]: A novel way for variance reduction
  $\beta \cdot (\texttt{Local stoch. gradient}) + (1-\beta) \cdot (\texttt{inner loop of SARAH})$
- Outperforms existing methods with a $\beta \in (0,1)$

1. R. Xin, U. A. Khan, and S. Kar, "An improved convergence analysis for decentralized online stochastic non-convex optimization," IEEE Transactions on Signal Processing, 69, pp. 1842-1858, Mar. 2021.

2. R. Xin, U. A. Khan, and S. Kar, "A hybrid variance-reduced method for decentralized stochastic non-convex optimization," in 38th International Conference on Machine Learning, Jul. 2021, accepted for publication.

Distributed optimization: Demo

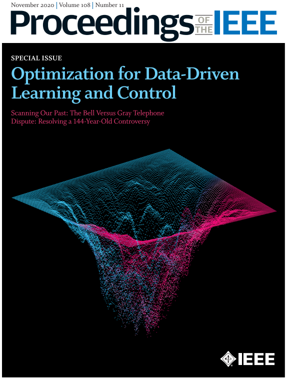
- Full gradient, distributed linear regression, $n = 100$ nodes
  - Each node possesses one data point
  - Collaborate to learn the slope and intercept

Conclusions

- Gradient tracking for distributed optimization
  - GT eliminates the local vs. global dissimilarity bias
  - Linear convergence for smooth and strongly convex problems
  - Acceleration is possible but analysis is hard!
- GT+VR: Gradient tracking for distributed batch optimization
  - GT-SAGA: State-of-the-art in the IoT regime
  - GT-SARAH: State-of-the-art in the big-data regime
- Gradient tracking for distributed online stochastic optimization
  - Shown best known rates for strongly convex and nonconvex problems in applicable regimes
  - Decaying step-sizes eliminate the variance due to the stochastic grad
  - Hybrid VR techniques
- Network-independent convergence behavior
- Outperforms the centralized analogs in applicable regimes

Optimization for Data-driven Learning and Control

- *There is a lot more being done and a lot more to do!*
- P-IEEE Special Issue, vol. 108, no. 11
  U. A. Khan, *Lead Editor*
  with Guest Editors: W. U. Bajwa, A. Nedić, M. G. Rabbat, A. H. Sayed



GT-SARAH: Analysis

GT-SARAH: Analysis

- Use the L-smoothness of F to establish the following lemma

$$F(\mathbf{y}) \leq F(\mathbf{x}) + \langle \nabla F(\mathbf{x}), \mathbf{y} - \mathbf{x} \rangle + \frac{L}{2} \|\mathbf{y} - \mathbf{x}\|^2 \qquad \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^p$$

Lemma (Descent inequality)

If the step-size follows that $0 < \alpha \leq \frac{1}{2L}$, *then we have*

$$\mathbb{E}\left[F\left(\overline{\mathbf{x}}^{T+1,K}\right)\right] \leq F(\overline{\mathbf{x}}^{0,1}) - \frac{\alpha}{2} \sum_{k,t}^{K,T} \mathbb{E}\left[\left\|\nabla F(\overline{\mathbf{x}}^{t,k})\right\|^2\right]$$

$$- \alpha \left(\frac{1}{4} \sum_{k,t}^{K,T} \mathbb{E}\left[\left\|\overline{\mathbf{v}}^{t,k}\right\|^2\right] - \sum_{k,t}^{K,T} \mathbb{E}\left[\left\|\overline{\mathbf{v}}^{t,k} - \overline{\nabla \mathbf{f}}(\mathbf{x}^{t,k})\right\|^2\right] - L^2 \sum_{k,t}^{K,T} \mathbb{E}\left[\frac{\left\|\mathbf{x}^{t,k} - \mathbf{1} \otimes \overline{\mathbf{x}}^{t,k}\right\|^2}{n}\right] \right)$$

- The object in red has two errors that we need to bound
  - Gradient estimation error: $\mathbb{E}[\|\overline{\mathbf{v}}^{t,k} - \overline{\nabla \mathbf{f}}(\mathbf{x}^{t,k})\|^2]$
  - Agreement error: $\mathbb{E}[\|\mathbf{x}^{t,k} - \mathbf{1} \otimes \overline{\mathbf{x}}^{t,k}\|^2]$

GT-SARAH: Analysis

Lemma (Gradient estimation error)

We have $\forall k \geq 1$,

$$\sum_{t=0}^{T} \mathbb{E}\left[\|\overline{\mathbf{v}}^{t,k} - \overline{\nabla \mathbf{f}}(\mathbf{x}^{t,k})\|^2\right] \leq \frac{3\alpha^2 TL^2}{n} \sum_{t=0}^{T-1} \mathbb{E}\left[\|\overline{\mathbf{v}}^{t,k}\|^2\right] + \frac{6TL^2}{n} \sum_{t=0}^{T} \mathbb{E}\left[\frac{\|\mathbf{x}^{t,k} - \mathbf{1} \otimes \bar{\mathbf{x}}^{t,k}\|^2}{n}\right].$$

Lemma (Agreement error)

If the step-size follows $0 < \alpha \leq \frac{(1-\lambda^2)^2}{8\sqrt{42}L}$, *then*

$$\sum_{k=1}^{K} \sum_{t=0}^{T} \mathbb{E}\left[\frac{\|\mathbf{x}^{t,k} - \mathbf{1} \otimes \bar{\mathbf{x}}^{t,k}\|^2}{n}\right] \leq \frac{64\alpha^2}{(1-\lambda^2)^3} \frac{\|\nabla \mathbf{f}(\mathbf{x}^{0,1})\|^2}{n} + \frac{1536\alpha^4 L^2}{(1-\lambda^2)^4} \sum_{k=1}^{K} \sum_{t=0}^{T} \mathbb{E}\left[\|\overline{\mathbf{v}}^{t,k}\|^2\right].$$

- *Agreement error is coupled with the gradient estimation error*
- *Derive an LTI system that describes their evolution*
- *Analyze the LTI dynamics to obtain the agreement error lemma*

- Use the two lemmas back in the descent inequality

GT-SARAH: Analysis

Lemma (Refined descent inequality)

For $0 < \alpha \leq \overline{\alpha} := \min\left\{\frac{(1-\lambda^2)^2}{4\sqrt{42}}, \frac{\sqrt{n}}{\sqrt{6T}}, \left(\frac{2n}{3n+12T}\right)^{\frac{1}{4}}\frac{1-\lambda^2}{6}\right\}\frac{1}{2L}$, *we have*

$$\frac{1}{n}\sum_{i,k,t}^{n,K,T}\mathbb{E}\left[\|\nabla F(\mathbf{x}_i^{t,k})\|^2\right] \leq \frac{4(F(\overline{\mathbf{x}}^{0,1}) - F^*)}{\alpha} + \left(\frac{3}{2} + \frac{6T}{n}\right)\frac{256\alpha^2L^2}{(1-\lambda^2)^3}\frac{\left\|\nabla \mathbf{f}(\mathbf{x}^{0,1})\right\|^2}{n}.$$

- Taking $K \to \infty$ on both sides leads to $\sum_{k,t}^{\infty,T}\mathbb{E}[\|\nabla F(\mathbf{x}_i^{t,k})\|] < \infty$
  - Mean-squared and a.s. results follow

- Divide both sides by $K \cdot T$ and solve for K when the R.H.S $\leq \epsilon$
  - Gradient computation complexity follows by nothing that GT-SARAH computes $n(m + 2T)$ gradients per iteration across all nodes
  - Choose α as the maximum and $T = \mathcal{O}(m)$ to obtain the optimal rate