Decentralized Learning in the Non-convex World: Recent Results

ZJU-CSE Summer School 2021

w/ Tsung-hui Chang (CUHKSZ), Mingyi Hong, Xinwei Zhang, Songtao Lu (UMN)

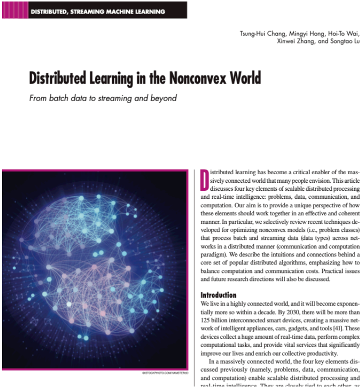
Hoi-To Wai (The Chinese University of Hong Kong)

August 9, 2021

ACK: CUHK Direct Grant, NSF, National Key R&D Program of China

This Talk

- The materials are based on (Chang et al., 2020):



DISTRIBUTED, STREAMING MACHINE LEARNING

Tsung-Hui Chang, Mingyi Hong, Hoi-To Wai,
Xinwei Zhang, and Songtao Lu

Distributed Learning in the Nonconvex World

From batch data to streaming and beyond

Distributed learning has become a critical enabler of the massively connected world that many people envision. This article discusses four key elements of scalable distributed processing and real-time intelligence: problems, data, communication, and computation. Our aim is to provide a unique perspective of how these elements should work together in an effective and coherent manner. In particular, we selectively review recent techniques developed for optimizing nonconvex models (i.e., problem classes) that process batch and streaming data (data types) across networks in a distributed manner (communication and computation paradigm). We describe the intuitions and connections behind a core set of popular distributed algorithms, emphasizing how to balance computation and communication costs. Practical issues and future research directions will also be discussed.

Introduction

We live in a highly connected world, and it will become exponentially more so within a decade. By 2030, there will be more than 125 billion interconnected smart devices, creating a massive network of intelligent appliances, cars, gadgets, and tools [41]. These devices collect a huge amount of real-time data, perform complex computational tasks, and provide vital services that significantly improve our lives and enrich our collective productivity.

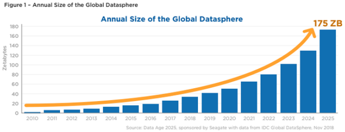
In a massively connected world, the four key elements discussed previously (namely, problems, data, communication, and computation) enable scalable distributed processing and

T.-H. Chang, M. Hong, H.-T. Wai, X. Zhang, S. Lu, "Distributed learning in the nonconvex world: From batch data to streaming and beyond", IEEE SPM, May, 2020.

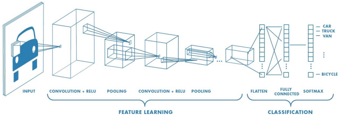
Motivation

We are living in a highly connected world ...

- By 2030, there can be $\geq$ **125 billion** connected smart devices.
- Huge amount of data generated in **real time** and also **locally**.

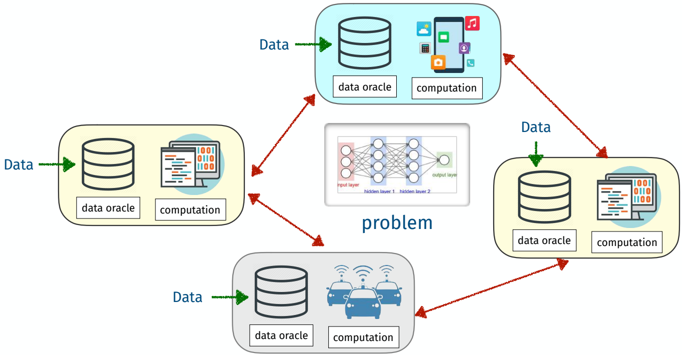


How do we generate 'intelligence' (e.g., training a machine learning model) from these data efficiently?

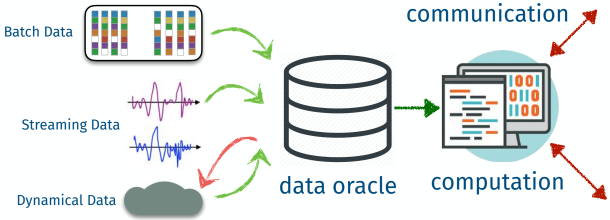


Distributed Learning

- A promising solution is to use *distributed learning* to enable **scalable** and **real-time** intelligence.
- Devices **work together** to solve a common problem —

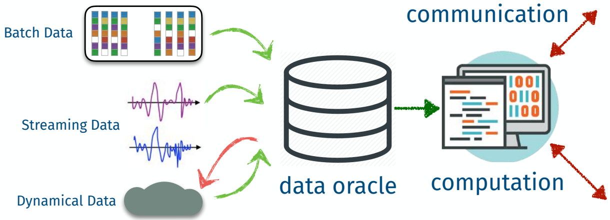


Distributed Learning: Key Aspects



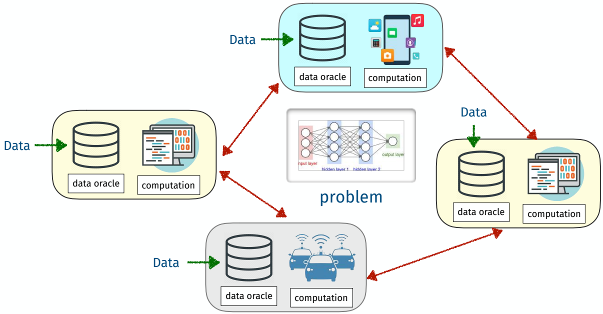
- *Problem class* — what type of optimization problem are we solving?
- *Data acquisition* — how do we acquire data?
- *Communication* — how devices should communicate to each other?
- *Computation* — what algorithms can we use under the above premises?

Distributed Learning: Key Aspects



- For machine learning (ML), it is common to use non-convex cost functions (e.g., neural networks).
- Data can be acquired in a *batch, streaming, or dynamical* manner.
- To achieve real time processing, *communication and computation* in the implementation have to be efficient.

Goal of This Talk



- Tutorial-style review on **state-of-the-art** distributed learning algorithms in handling **non-convex optimization** problems[1].
- Focus on the **data oracle** affecting the **algorithm design**.
- Introducing **future research directions**.

[1] To simplify the presentation, most technical details will be skipped.

Roadmap

1. Background & Mathematical Preliminary

2. Non-convex Distributed Learning: Algorithms and Theory

   Batch Data

   Streaming Data

3. Extensions

   Dynamic Data

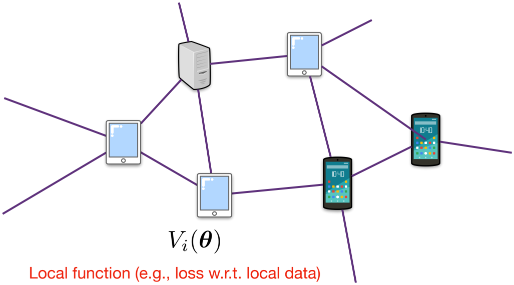
   Other Extensions

4. Wrapping up & Open Problems

Background & Mathematical Preliminary

Setup and Notations

- Multi-agent (device) system on a graph $G = (V, E)$.



- Consider n agents and a possibly non-convex optimization problem:

$$\min_{\boldsymbol{\Theta}=(\boldsymbol{\theta}_1;\ldots;\boldsymbol{\theta}_n)\in\mathbb{R}^{n\times d}} F(\boldsymbol{\Theta}) := \frac{1}{n}\sum_{i=1}^{n} f_i(\boldsymbol{\theta}_i) \text{ s.t. } \boldsymbol{\theta}_i = \boldsymbol{\theta}_j,\ \forall\ (i,j) \in E. \quad \text{(P)}$$

- $f_i : \mathbb{R}^d \to \mathbb{R}$ is a function known **only** to the ith agent.

Problem Class

- Throughout this talk, we limit ourselves to problems satisfying:

H1 For $i = 1, ..., n$, the cost function f_i is L-smooth, i.e.,

$$\|\nabla f_i(\boldsymbol{\theta}) - \nabla f_i(\boldsymbol{\theta}')\| \leq L\|\boldsymbol{\theta} - \boldsymbol{\theta}'\|, \ \forall \ \boldsymbol{\theta}, \boldsymbol{\theta}' \in \mathbb{R}^d,$$

and the averaged function F is lower bounded over $\mathbb{R}^d$.

- It is a mild assumption for *common cost functions*, e.g., logistic loss with neural network.
- **Remark**: by H1, we excluded constrained non-convex learning.

H2 The graph G is undirected and connected.

- This implies $\boldsymbol{\theta}_i = \boldsymbol{\theta}_j$ for **any** $i, j \Rightarrow$ agents learn a *common model*.

Example

- **Binary Classifier Training**: Agent i has

$$\{\xi_{i,1}, \ldots, \xi_{i,M_i}\} \quad \text{with} \quad \xi_{i,\ell} = (\underbrace{\boldsymbol{x}_{i,\ell}}_{\text{feature}}, \underbrace{y_{i,\ell}}_{\text{label}}) \in \mathbb{R}^d \times \{0,1\}.$$

- *Goal*: to train the weights $\boldsymbol{\theta}$ of a neural net (NN),

$$f_i(\boldsymbol{\theta}; \xi_{i,\ell}) = (1 - y_{i,\ell}) \log(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x}_{i,\ell})) + y_{i,\ell} \log h_{\boldsymbol{\theta}}(\boldsymbol{x}_{i,\ell})$$

  where $h_{\boldsymbol{\theta}} : \mathbb{R}^d \to \mathbb{R}$ is the output of the NN.

- If *activation function* is smooth, e.g., sigmoid, H1 is satisfied and (P) is non-convex.

- Other applications: **matrix factorization**, **policy optimization**, etc..

Data Oracle

- During *learning*, data is revealed via a **local oracle map** of first-order information:

$$\mathsf{DO}_i : \mathbb{R}^d \to \mathbb{R}^d$$

- **Batch Data —** entire data available at anytime (**easiest setting**),

$$\mathsf{DO}_i(\boldsymbol{\theta}_i^t) = \nabla f_i(\boldsymbol{\theta}_i^t) \quad \text{with} \quad f_i(\boldsymbol{\theta}) = M_i^{-1} \sum_{\ell=1}^{M_i} f_i(\boldsymbol{\theta}; \xi_{i,\ell})$$

- **Streaming Data —** data revealed in an online fashion,

$$\mathsf{DO}_i(\boldsymbol{\theta}_i^t) = \nabla f_i(\boldsymbol{\theta}_i^t; \xi_i^t), \ \xi_i^t \overset{i.i.d.}{\sim} \pi_i(\cdot) \quad \text{with} \quad f_i(\boldsymbol{\theta}) = \mathbb{E}_{\xi \sim \pi_i(\cdot)} f_i(\boldsymbol{\theta}; \xi)$$

- In addition, **dynamic data** is drawn from distribution depending on $\boldsymbol{\theta}_i^t$.

Prior Works

Many algorithms have been proposed for **convex optimization**:

- Distributed gradient (DGD) method (Nedić and Ozdaglar, 2009),
- ADMM based algorithms (Boyd et al., 2011; Jakovetic et al., 2011),
- EXTRA (Shi et al., 2015) and its time varying graph extension DIG-ing (Nedic et al., 2017),
- Gradient Tracking techniques (Qu and Li, 2017) and extension to directed graphs (Xi and Khan, 2017; Pu et al., 2020),
- Optimal algorithms (Scaman et al., 2017; Uribe et al., 2020),
- and many others ...

Do they work on (P) in general?

Consider a **non-convex** problem for 2 agents, satisfying H1-H2:

$$\min_{\boldsymbol{\Theta}=(\theta_1,\theta_2)\in\mathbb{R}^2} \frac{\theta_1^2}{2} + \left(-\frac{\theta_2^2}{2}\right) \equiv f_1(\theta_1) + f_2(\theta_2) \quad \text{s.t.} \quad \theta_1 = \theta_2.$$

The DGD algorithm (Nedić and Ozdaglar, 2009) yields the recursion

$$\boldsymbol{\theta}^{t+1} = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} \end{pmatrix} \boldsymbol{\theta}^t - \gamma \begin{pmatrix} \theta_1^t \\ -\theta_2^t \end{pmatrix} = \begin{pmatrix} \frac{1}{2} - \gamma & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} + \gamma \end{pmatrix} \boldsymbol{\theta}^t$$

For any $\gamma > 0$, DGD diverges as $|\theta_1^t - \theta_2^t| \to \infty$!

Take-away point: *caution needed when tackling non-convex problems.*

Non-convex Distributed Learning: Algorithms and Theory

What is a 'Good' Solution to (P)?

- Even with H1, H2, solving (P) to global optimum can be **NP-hard**.
- We resort to finding *stationary and consensual solution* with small gradient and variables are in consensus:

Def. Let $\epsilon > 0$, $\boldsymbol{\Theta} = (\boldsymbol{\theta}_1; \ldots; \boldsymbol{\theta}_n)$ is an **ϵ-stationary** solution if

$$\mathsf{Gap}(\boldsymbol{\Theta}) = \|n^{-1} \sum_{j=1}^n \nabla f_j(\bar{\boldsymbol{\theta}})\|^2 + \sum_{j=1}^n \|\boldsymbol{\theta}_j - \bar{\boldsymbol{\theta}}\|^2 \leq \epsilon,$$

where $\bar{\boldsymbol{\theta}} := n^{-1} \sum_{j=1}^n \boldsymbol{\theta}_j$ in the averaged solution.

Goal: *find an ϵ-stationary solution $\boldsymbol{\Theta}$ satisfying* $\mathsf{Gap}(\boldsymbol{\Theta}) \leq \epsilon$ *using a distributed learning algorithm*[2].

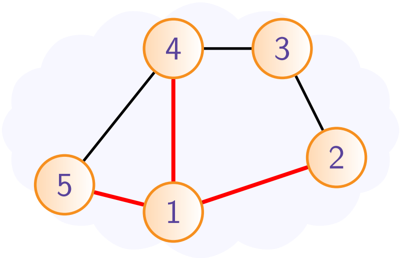
[2] Stronger notions of stationarity, e.g., second order stationary point, can be considered $\Leftarrow$ skipped in the interest of time; see (Vlaski and Sayed, 2021; Daneshmand et al., 2020).

Distributed Processing on Networks

- Let $\boldsymbol{W} \in \mathbb{R}^{n\times n}$ be a **mixing matrix** satisfying

$$W_{ij} = \begin{cases} \in (0,1], & \text{if } (i,j) \in E \\ 0, & \text{otherwise.} \end{cases} \quad \text{and} \quad \begin{array}{ll} \text{(i)} & \mathrm{null}\{\boldsymbol{I} - \boldsymbol{W}\} = \mathrm{span}\{1\}, \\ \text{(ii)} & -\boldsymbol{I} \preceq \boldsymbol{W} \preceq \boldsymbol{I} \end{array}$$

- Note $x_i' = \sum_{j=1}^n W_{ij} x_j$ performs **local averaging**. As $\boldsymbol{W}^\infty = 11^\top/n$, it attains *consensus asymptotically* (Tsitsiklis, 1984). For example,



$$\boldsymbol{W} = \begin{pmatrix} \frac14 & \frac14 & 0 & \frac14 & \frac14 \\ \frac14 & \frac12 & \frac14 & 0 & 0 \\ 0 & \frac14 & \frac12 & \frac14 & 0 \\ \frac14 & 0 & \frac14 & \frac14 & \frac14 \\ \frac14 & 0 & 0 & \frac14 & \frac12 \end{pmatrix}$$

Batch Data: Basic Algorithm

- **Goal**: want a consensual + stationary solution, i.e., $\text{Gap}(\boldsymbol{\Theta}) \leq \epsilon$.
- **Batch data**: can access to $\nabla f_i(\boldsymbol{\theta})$ for any $\boldsymbol{\theta} \in \mathbb{R}^d$.
- Distributed gradient (DGD) algorithm (Nedić and Ozdaglar, 2009) — let $\boldsymbol{\theta}_i^0 \in \mathbb{R}^d$ be an initial solution at agent i, it follows

$$\boldsymbol{\theta}_i^{t+1} = \underbrace{\sum_{j=1}^{n} W_{ij}\boldsymbol{\theta}_j^t}_{\text{consensus}} - \underbrace{\gamma_t}_{\text{step size}} \underbrace{\nabla f_i(\boldsymbol{\theta}_i^t)}_{\text{local gradient}} \quad \left(\Leftrightarrow \begin{array}{l} \text{in matrix notation,} \\ \boldsymbol{\Theta}^{t+1} = \boldsymbol{W}\boldsymbol{\Theta}^t - \gamma_t \nabla \boldsymbol{f}(\boldsymbol{\Theta}^t) \end{array} \right)$$

- Each iteration uses only the **local gradient** $\nabla f_i(\boldsymbol{\theta}_i^t)$.

Fact (Zeng and Yin, 2018, Theorem 2) – Suppose that $\gamma_t = c/t$ for some $c > 0$ and the gradients are bounded, then $\text{Gap}(\boldsymbol{\Theta}^t) \to 0$.

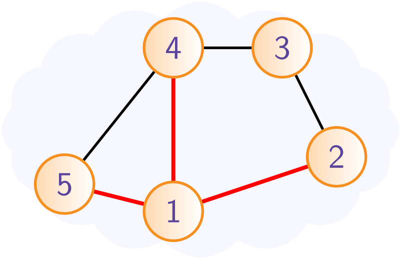
— assumptions violated by counterexample; see (Bianchi et al., 2013).

Batch Data: Primal-dual Algorithm

- Consider the consensus constraint in (P): let $\boldsymbol{A} \in \mathbb{R}^{|E| \times n}$,

$$\boldsymbol{\theta}_i = \boldsymbol{\theta}_j,\ \forall\,(i,j) \in E \;\Leftrightarrow\; \boldsymbol{A}\boldsymbol{\Theta} = 0 \;\text{ s.t. }\; A_{e,i} = \begin{cases} 1, & \text{if } i \in e, i < j \\ -1, & \text{if } i \in e, i > j \\ 0, & \text{otherwise} \end{cases}$$

with $e \equiv (i,j) \in E$ and $\boldsymbol{A}$ is the graph incidence matrix.



$$\boldsymbol{A} = \begin{pmatrix} 1 & -1 & 0 & 0 & 0 \\ 1 & 0 & 0 & -1 & 0 \\ 1 & 0 & 0 & 0 & -1 \\ 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 1 & -1 \end{pmatrix}$$

Note that $\boldsymbol{A}^\top \boldsymbol{A} = \boldsymbol{L}_G$, i.e., the graph Laplacian matrix.

Batch Data: Primal-dual Algo. (cont'd)

- The **augmented Lagrangian function** of (P): let $\mu \in \mathbb{R}^{|E| \times d}$, $c > 0$,

$$\mathcal{L}(\boldsymbol{\Theta}, \boldsymbol{\mu}) = \frac{1}{n} \sum_{i=1}^{n} f_i(\boldsymbol{\theta}_i) + \langle \boldsymbol{\mu}, \boldsymbol{A\Theta} \rangle + \frac{c}{2} \|\boldsymbol{A\Theta}\|_{\mathrm{F}}^2,$$

- We can apply a linearized **primal-dual algorithm** —

$$\boldsymbol{\Theta}^{t+1} \leftarrow \underset{\boldsymbol{\Theta} \in \mathbb{R}^n}{\arg\min} \Big\{ \Big\langle \underbrace{\nabla \boldsymbol{f}(\boldsymbol{\Theta}^t) + \boldsymbol{A}^\top \boldsymbol{\mu}^t + c\boldsymbol{A}^\top \boldsymbol{A\Theta}^t}_{\text{linearizing } \mathcal{L} \text{ at } (\boldsymbol{\Theta}^t, \boldsymbol{\mu}^t)}, \boldsymbol{\Theta} \Big\rangle + \frac{1}{2} \|\boldsymbol{\Theta} - \boldsymbol{\Theta}^t\|_{\tilde{\boldsymbol{D}}}^2 \Big\},$$

$$\boldsymbol{\mu}^{t+1} \leftarrow \boldsymbol{\mu}^t + c \underbrace{\boldsymbol{A\Theta}^{t+1}}_{\text{linearizing } \mathcal{L} \text{ at } (\boldsymbol{\Theta}^{t+1}, \boldsymbol{\mu}^t)}$$

- Looks complicated? we may set $\tilde{\boldsymbol{D}} = \boldsymbol{\Upsilon} + 2c\boldsymbol{D} \succ 0$, where $\boldsymbol{\Upsilon}$ is a diagonal matrix...

Batch Data: Primal-dual Algo. (cont'd)

- After some manipulation, we can derive the **Prox-GPDA** algorithm:

$$\boldsymbol{\Theta}^{t+1} = \underbrace{\left(\boldsymbol{I}_n - \beta \boldsymbol{A}^\top \boldsymbol{A}\right)}_{=\boldsymbol{W} \text{ if } \beta \text{ is small enough}} (2\boldsymbol{\Theta}^t - \boldsymbol{\Theta}^{t-1}) - \alpha \left\{ \nabla \boldsymbol{f}(\boldsymbol{\Theta}^t) - \nabla \boldsymbol{f}(\boldsymbol{\Theta}^{t-1}) \right\}.$$

- Need to keep the **previous** iterates and requires extra communication round but is still *decentralized*.

- *Example*: Gradient Tracking (GT) (Qu and Li, 2017):

$$\boldsymbol{\Theta}^{t+1} = \hat{\boldsymbol{W}} \boldsymbol{\Theta}^t - \alpha \boldsymbol{g}^t, \quad \boldsymbol{g}^{t+1} = \hat{\boldsymbol{W}} \boldsymbol{g}^t + \nabla \boldsymbol{f}(\boldsymbol{\Theta}^{t+1}) - \nabla \boldsymbol{f}(\boldsymbol{\Theta}^t)$$
$$\Leftrightarrow \ \boldsymbol{\Theta}^{t+1} = 2\hat{\boldsymbol{W}} \boldsymbol{\Theta}^t - \hat{\boldsymbol{W}}^2 \boldsymbol{\Theta}^{t-1} - \alpha \left(\nabla \boldsymbol{f}(\boldsymbol{\Theta}^t) - \nabla \boldsymbol{f}(\boldsymbol{\Theta}^{t-1}) \right)$$

- *Example*: EXTRA (Shi et al., 2015):

$$\boldsymbol{\Theta}^{t+1} = (\boldsymbol{I}_n + \tilde{\boldsymbol{W}}) \boldsymbol{\Theta}^t - \frac{1}{2} (\boldsymbol{I}_n + \tilde{\boldsymbol{W}}) \boldsymbol{\Theta}^{t-1} - \alpha [\nabla \boldsymbol{f}(\boldsymbol{\Theta}^t) - \nabla \boldsymbol{f}(\boldsymbol{\Theta}^{t-1})]$$

- Also includes DGD (Nedić and Ozdaglar, 2009) when $\boldsymbol{\mu} = 0$ ...

Batch Data: Primal-dual Algo. (cont'd)

- After some manipulation, we can derive the **Prox-GPDA** algorithm:

$$\Theta^{t+1} = \underbrace{(\boldsymbol{I}_n - \beta \boldsymbol{A}^\top \boldsymbol{A})}_{=\boldsymbol{W} \text{ if } \beta \text{ is small enough}} (2\Theta^t - \Theta^{t-1}) - \alpha \{\nabla \boldsymbol{f}(\Theta^t) - \nabla \boldsymbol{f}(\Theta^{t-1})\}.$$

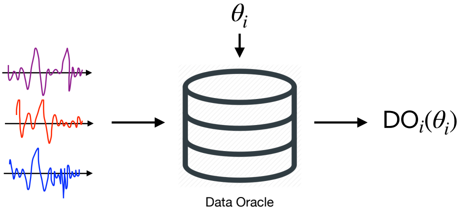
- Analyzing **Prox-GPDA** allows us to analyze EXTRA (Shi et al., 2015), GT (Qu and Li, 2017) in a unified fashion:

Fact (Hong et al., 2017, Theorem 1) – For sufficiently small α, we have $\mathrm{Gap}(\Theta^{(t)}) = \mathcal{O}(1/t)$ for Prox-GPDA.

Remark: does not require bounded gradient $\Rightarrow$ Prox-GPDA/EXTRA/GT algorithms converge in the counter example; also see (Di Lorenzo and Scutari, 2016; Scutari and Sun, 2019).

Streaming Data

- Data may arrive in a **streaming** fashion.



- Local gradient $\nabla f_i(\boldsymbol{\theta}_i^t)$ is difficult to obtain, e.g., when $M_i \gg 1$,

$$\nabla f_i(\boldsymbol{\theta}_i) = (1/M_i) \sum_{j=1}^{M_i} \nabla_{\boldsymbol{\theta}} f_i(\boldsymbol{\theta}_i; \xi_{i,j}) = \mathbb{E}_{\xi \sim \pi_i}[\nabla f_i(\boldsymbol{\theta}_i; \xi)]$$

- **Example**: $\mathrm{DO}_i(\boldsymbol{\theta}_i; \xi) = \nabla f_i(\boldsymbol{\theta}_i; \xi)$ and ξ is drawn with $\xi \sim \pi_i$, i.e.,

H3. For any $\boldsymbol{\theta}_i \in \mathbb{R}^d$, $\mathrm{DO}_i(\boldsymbol{\theta}_i)$ is unbiased with bounded variance:

$$\mathbb{E}[\mathrm{DO}_i(\boldsymbol{\theta}_i; \xi)] = \nabla f_i(\boldsymbol{\theta}_i), \quad \mathbb{E}_{\xi \sim \pi_i}[\|\mathrm{DO}_i(\boldsymbol{\theta}_i; \xi) - \nabla f_i(\boldsymbol{\theta}_i)\|^2] \leq \sigma.$$

Streaming Data: Basic Algorithm

- **Decentralized Stochastic Gradient Descent (DSGD)** ($\approx$ DGD):

$$\boldsymbol{\theta}_i^{t+1} = \sum_{j=1}^{n} W_{ij} \boldsymbol{\theta}_j^t - \gamma_t \underbrace{\mathsf{DO}_i(\boldsymbol{\theta}_i^t; \xi_i^t)}_{\text{replace } \nabla f_i(\boldsymbol{\theta}_i^t) \text{ by stoc. grad..}}$$

- **Strong assumption** needed for convergence ...

H4. For any $\boldsymbol{\theta} \in \mathbb{R}^d$, the local gradient is not far away from averaged gradient:

$$n^{-1} \sum_{i=1}^{n} \|\nabla f_i(\boldsymbol{\theta}) - \nabla F(\boldsymbol{\theta})\|^2 \leq \varsigma^2.$$

— the local function is *homogeneous*.

Fact (Lian et al., 2017) – Set $\gamma_t = \mathcal{O}(\sqrt{1/T})$ and under H1–H4, we have $\mathbb{E}[\mathrm{Gap}(\boldsymbol{\Theta}^{\bar{t}(T)})] = \mathcal{O}(\sigma/\sqrt{nT})$, where $\bar{t}(T) \sim \mathcal{U}\{1, ..., T\}$.

Streaming Data: Inhomogeneous Fct.

- To relax H4, we can utilize gradient tracking to derive the GT-based Non-convex SGD (**GNSD**) algorithm:

$$\boldsymbol{\Theta}^{t+1} = \hat{\boldsymbol{W}}\boldsymbol{\Theta}^{t} - \gamma \boldsymbol{g}^{t}, \quad \boldsymbol{g}^{t+1} = \hat{\boldsymbol{W}}\boldsymbol{g}^{t} + \mathsf{DO}(\boldsymbol{\Theta}^{t+1}; \boldsymbol{\xi}^{t+1}) - \mathsf{DO}(\boldsymbol{\Theta}^{t}; \boldsymbol{\xi}^{t})$$
$$\Leftrightarrow \; \boldsymbol{\Theta}^{t+1} = 2\hat{\boldsymbol{W}}\boldsymbol{\Theta}^{t} - \hat{\boldsymbol{W}}^2\boldsymbol{\Theta}^{t-1} - \gamma \left(\mathsf{DO}(\boldsymbol{\Theta}^{t}; \boldsymbol{\xi}^{t}) - \mathsf{DO}(\boldsymbol{\Theta}^{t-1}; \boldsymbol{\xi}^{t-1}) \right)$$

- **Insight**: gradient tracking (GT) *removes* inhomogeneity across fcts.
- GNSD converges with the same rate as DSGD, but with weaker assumption, i.e.,

Fact (Lu et al., 2019) – Set $\gamma = \mathcal{O}(\sqrt{1/T})$ and under H1–H3, we have $\mathbb{E}[\mathsf{Gap}(\boldsymbol{\Theta}^{\bar{t}(T)})] = \mathcal{O}(\sigma/\sqrt{nT})$, where $\bar{t}(T) \sim \mathcal{U}\{1, ..., T\}$.

Streaming Data: Observations

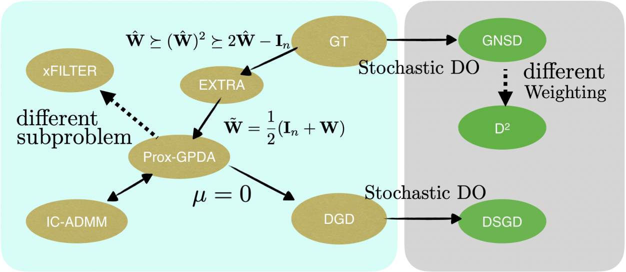
- The algorithms are similar to batch data algorithms but require different step size rule for convergence.
- We need $\gamma = \mathcal{O}(\sqrt{1/T})$[3] to combat the non-vanishing noise variance due to H3 in the DO — unlike the case of batch data.
- Convergence is based on a random stopping criteria, in fact,

$$\mathbb{E}[\mathrm{Gap}(\boldsymbol{\Theta}^{\bar{t}(T)})] = \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}[\mathrm{Gap}(\boldsymbol{\Theta}^{t})] \quad \text{with} \quad \bar{t}(T) \sim \mathcal{U}\{1, \dots, T\}.$$

A common criteria in non-convex stochastic optimization, where $\bar{t}(T)$ is treated as a *random stopping iteration*.

[3]Alternatively, a diminishing step size of order $\gamma_t = \Theta(1/\sqrt{t})$ can be used.

Short Summary



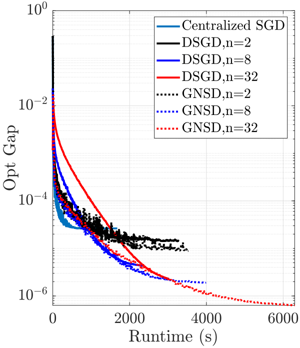
	DGD	Prox-GPDA	DSGD	GNSD
# Comm./Iter.	1	2	1	2
Assumptions	strong	weak	strong	weak
Gap($\mathbf{\Theta}^t$)	N/A	$\mathcal{O}(1/t)$	$\mathcal{O}(1/\sqrt{t})$	$\mathcal{O}(1/\sqrt{t})$

- **Strong assumptions** needed for simpler algorithms to converge.
- Primal-dual formulation unifies many existing algorithms $\Rightarrow$ lead to *optimal algorithms* such as xFILTER.

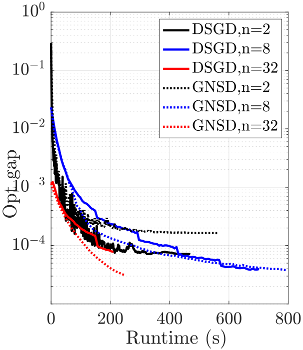
Numerical Experiments

- **Task**: handwritten digit classification from the MNIST dataset with 4.8×10^4 training samples, divided evenly across n nodes.
- **Setup**: classification using a 2-hidden-layer NN with $(512, 128)$ neurons, totaling $d = 4.68 \times 10^5$ parameters.
- **Environment** — AWS (Amazon) cluster has *better* communication efficiency than the MSI (UMN) cluster.
- Nodes connected on random regular graphs.

Numerical Experiments: Netwk. Scalability



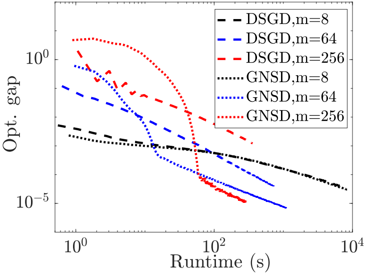
on MSI (slower comm.)

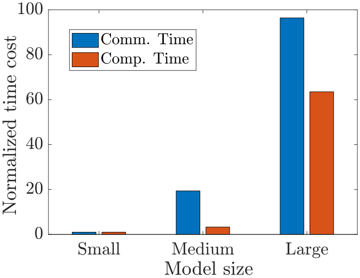


on AWS (better comm.)

- As n increases, both GNSD and DSGD achieves lower opt. gap.
- When *communication overhead* is large (limited by HW), increasing n slows down the convergence.

Numerical Experiments (cont'd)



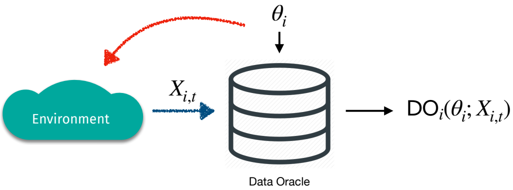


- With *heterogeneous data*, DSGD has worse solution than GNSD.
- GNSD benefits from increased batch size m.

- As model size increases, comm. cost $>$ compute cost.

Extensions

Dynamic Data: Biased DO



Dynamic DO. For any $\boldsymbol{\theta} \in \mathbb{R}^d$ and at time/iteration t, we have:

$$\mathsf{DO}_i(\boldsymbol{\theta}; X_{i,t}) = \mathcal{H}_i(\boldsymbol{\theta}; X_{i,t}) \text{ s.t. } \lim_{t\to\infty} \mathbb{E}[\mathcal{H}_i(\boldsymbol{\theta}; X_{i,t})] = \mathbb{E}_{X\sim\pi_i^{\boldsymbol{\theta}}(\cdot)}[\mathcal{H}_i(\boldsymbol{\theta}; X)]$$

E.g., $\{X_{i,t}\}_{t\geq 1}$ is Markov chain w/ stationary distribution $\pi_i^{\boldsymbol{\theta}}(\cdot)$.

- The t-th sample $\mathsf{DO}_i(\boldsymbol{\theta}; X_{i,t}) \neq$ **i.i.d.** and is *controlled* by $\boldsymbol{\theta}$.
- Mean field $h_i(\boldsymbol{\theta}) = \mathbb{E}_{X\sim\pi_i^{\boldsymbol{\theta}}(\cdot)}[\mathcal{H}_i(\boldsymbol{\theta}; X)]$ may be *non-gradient*.
- **Applications**: *reinforcement learning* – $\boldsymbol{\theta}$ = policy, ξ = current state, *strategic classification* – $\boldsymbol{\theta}$ = classifier, ξ = 'optimized' samples.

Dynamic Data: Basic Algorithm

- Similar to DGD and DSGD, we consider a general decentralized Stochastic Approximation (DSA) scheme:

$$\boldsymbol{\theta}_i^{t+1} = \sum_{j=1}^{n} W_{ij}\boldsymbol{\theta}_j^t - \gamma_t \underbrace{\mathcal{H}_i(\boldsymbol{\theta}_i^t; X_{i,t})}_{\text{replace } \nabla f_i(\boldsymbol{\theta}_i^t) \text{ by dynamic DO}}$$

- Let $h(\boldsymbol{\theta}) = (1/n)\sum_{i=1}^{n} h_i(\boldsymbol{\theta})$. We require this mean-field to be correlated with gradient of (P): $\exists c_0, d_0 > 0$,

$$\left\langle \overline{h}(\boldsymbol{\theta}) \,|\, \nabla F(\boldsymbol{\theta}) \right\rangle \geq c_0 \|\overline{h}(\boldsymbol{\theta})\|^2, \;\; d_0 \|\overline{h}(\boldsymbol{\theta})\|^2 \geq \|\nabla F(\boldsymbol{\theta})\|^2, \; \forall\, \boldsymbol{\theta} \in \mathbb{R}^d.$$

- **Example**: expectation-maximization (EM) algorithm for latent data model, policy gradient via REINFORCE (Karimi et al., 2019), etc..
- The DSA scheme finds a solution with $\|\overline{h}(\boldsymbol{\theta}_c)\| \approx 0$.

Dynamic Data: Preliminary Results

- **Challenges**: algorithm with non-i.i.d.+non-gradient DO.

H5 For any $\boldsymbol{\Theta} = (\boldsymbol{\theta}_1; \ldots; \boldsymbol{\theta}_n)$, there exists σ_o, σ_h s.t.

$$\sup_{x \in \mathsf{X}} \|\mathcal{H}_i(\boldsymbol{\theta}_i; x) - \frac{1}{n} \sum_{j=1}^n \mathcal{H}_j(\boldsymbol{\theta}_j; x)\| \leq \sigma_o \{ \frac{1}{n} + \frac{1}{n} \|\overline{h}(\tilde{\boldsymbol{\theta}}_c)\| + \|\boldsymbol{\theta}_i - \tilde{\boldsymbol{\theta}}_c\| \}.$$

$$\sup_{x \in \mathsf{X}} \left\| \frac{1}{n} \sum_{i=1}^n \mathcal{H}_i(\tilde{\boldsymbol{\theta}}_c; x) - \overline{h}(\tilde{\boldsymbol{\theta}}_c) \right\| \leq \sigma_h.$$

In addition, we assume $\pi_i^{\boldsymbol{\theta}}(\cdot) \equiv \pi_i(\cdot)$, i.e., an **uncontrolled MC**.

- Local DO $\neq$ far away from avg. $\leftarrow$ relaxed over Lian et al. (2017).
- DOs have uniformly bounded error from the mean field.

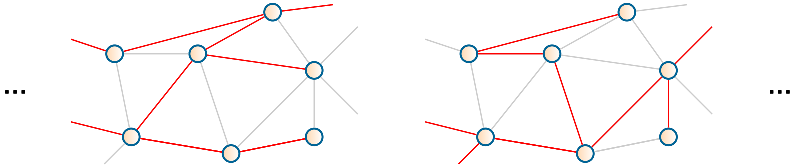
Fact (Wai, 2020) – Set $\gamma_t = \mathcal{O}(\sqrt{1/t})$ and under H1–H2, H5 with the **dynamical DO** setting. For any $T \geq 1$, $\mathbb{E}[\mathrm{Gap}(\boldsymbol{\Theta}^{\bar{t}(T)})] = \mathcal{O}(1/\sqrt{T})$, where $\mathbb{P}(\bar{t}(T) = t) = \gamma_t / \sum_{j=1}^T \gamma_j$.

Dynamic Data: Limitations

- Previous result handled non-i.i.d.+non-gradient DO, but ignored the possibility of controlled Markov chain.
- The latter is important for multi-agent reinforcement learning; see the model in (Wai et al., 2018).
- H5 assumes uniformly bounded error for DO which may fail if the state space X is unbounded.
- *Distributed non-convex learning* with dynamic data is still an open problem.

Other Extensions

- **Time varying graph** — some network links may fail, e.g.,

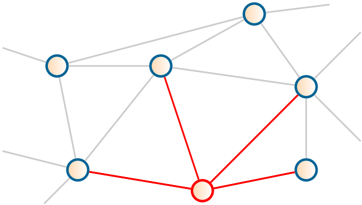


at t represented by $\boldsymbol{W}_t$ at $t+1$ represented by $\boldsymbol{W}_{t+1}$

- Easy to extend the analysis for DGD, DSGD, DSA, e.g., (Wai, 2020), with **similar** rate of convergence.
- Not so easy with Prox-GPDA, GNSD, see (Scutari and Sun, 2019; Nedic et al., 2017).
- Extensions to **directed graphs** are also closely related...

Other Extensions

- **Adversarial attack** — an attacker can easily misguide the whole network during the learning algorithm,



- With the *flat architecture* of distributed learning, it is easy to misguide the agents $\leftarrow$ *how to protect the algorithm against attackers?*
- Possible solutions: robust averaging (Yang and Bajwa, 2019); robust averaging + normalized gradient (Turan et al., 2021)

Other Extensions

- **Variance reduction (VR)** — most results on streaming data showed the rate of $\mathbb{E}[\text{GAP}(\boldsymbol{\Theta}^t)] = \mathcal{O}(1/\sqrt{t})$ – *Can it be improved?*

- Yes, improvable to $\mathbb{E}[\text{GAP}(\boldsymbol{\Theta}^t)] = \mathcal{O}(1/t^{2/3})$ – by GT + VR:

$$\boldsymbol{v}_i^t = \beta \underbrace{\nabla f_i(\boldsymbol{\theta}_i^t; \xi_i^t)}_{=\text{stoc. grad}} + (1-\beta) \underbrace{\{\boldsymbol{v}_i^{t-1} + \nabla f_i(\boldsymbol{\theta}_i^t; \xi_i^t) - \nabla f_i(\boldsymbol{\theta}_i^{t-1}; \xi_i^t)\}}_{=\text{variance reduction (Nguyen et al., 2017)}}$$

$$\boldsymbol{g}_i^t = \sum_{j=1}^n W_{ij} \boldsymbol{g}_i^{t-1} + \{\boldsymbol{g}_i^t - \boldsymbol{g}_i^{t-1}\}$$

  see (Xin et al., 2021; Pan et al., 2020).

- The rate of $\mathcal{O}(1/t^{2/3})$ is *optimal* (Cutkosky and Orabona, 2019).

Other Extensions

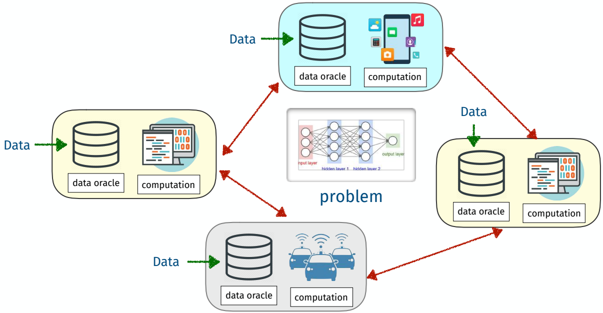
- **Compression** — each iter. needs ≥ 1 comm. to send d numbers – *Huge communication cost when $d \gg 1$ (e.g., large NN)!*
- **Idea**: compress before transmission via error compensation $\Rightarrow$ CHOCO-SGD algorithm (Koloskova et al., 2019)

$$\hat{\boldsymbol{\theta}}_{i,j}^{t+1} = \hat{\boldsymbol{\theta}}_{i,j}^{t} + \mathcal{Q}\Big(\underbrace{\boldsymbol{\theta}_j^t - \gamma_t \mathsf{DO}_j(\boldsymbol{\theta}_j^t) - \hat{\boldsymbol{\theta}}_{j,j}^t}_{\text{compressed + broadcast by agent } j} \Big), \forall\, j \in \mathcal{N}_i$$

  and update $\boldsymbol{\theta}_i^{t+1} = \sum_{j=1}^n W_{ij} \hat{\boldsymbol{\theta}}_{i,j}^{t+1}$.
- Each agent only broadcast a *compressed message* per round.
- The **CHOCO-SGD** algorithm mimics the DSGD algorithm, it can be shown that $\mathbb{E}[\mathsf{GAP}(\boldsymbol{\Theta}^t)] = \mathcal{O}(1/\sqrt{t})$.
- **Remark**: also work for constrained optimization (Wai et al., 2017a).

Wrapping up & Open Problems

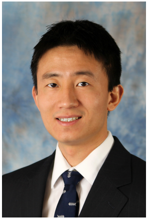
Take Home Points



- Strategies for distributed learning depend on *problem class, data model, computation* and *communication*.
- Trade-off between **simple** algo. and **weak** assumption for convergence $\leftarrow$ *primal-dual* formulation leads to many interesting designs.
- Recent results: *stronger convergence guarantees* & *general data model*.

Open Problems

- General convergence analysis for **fully dynamic** data.
  - *Can we consider* **controlled** *Markov chain and/or unbounded state space? see (Karimi et al., 2019), (Durmus et al., 2021).*
  - *What benefit does it bring if we combine gradient tracking?*
- **Privacy-preserving** decentralized learning.
  - *What measures shall be taken to encrypt message before communication? also see (Wai et al., 2017b).*
  - *Is there any tradeoff between rate of convergence and privacy?*
- Communication efficient algorithms for **high-dimensional** learning.
  - *Can we design new algorithms for communication efficient learning instead of building upon existing ones?*

Thank you!

Questions & Comments are welcomed. Online version on https://arxiv.org/abs/2001.04786.

References

Bianchi, P., Fort, G., and Hachem, W. (2013). Performance of a distributed stochastic approximation algorithm. *IEEE Transactions on Information Theory*, 59(11):7405–7418.

Boyd, S., Parikh, N., and Chu, E. (2011). *Distributed optimization and statistical learning via the alternating direction method of multipliers*. Now Publishers Inc.

Chang, T.-H., Hong, M., Wai, H.-T., Zhang, X., and Lu, S. (2020). Distributed learning in the non-convex world: From batch to streaming data, and beyond. *IEEE Signal Processing Magazine*.

Cutkosky, A. and Orabona, F. (2019). Momentum-based variance reduction in non-convex sgd. *Advances in neural information processing systems*, 32.

Daneshmand, A., Scutari, G., and Kungurtsev, V. (2020). Second-order guarantees of distributed gradient algorithms. *SIAM Journal on Optimization*, 30(4):3029–3068.

Di Lorenzo, P. and Scutari, G. (2016). Next: In-network nonconvex optimization. *IEEE Transactions on Signal and Information Processing over Networks*, 2(2):120–136.

Durmus, A., Moulines, E., Naumov, A., Samsonov, S., and Wai, H.-T. (2021). On the stability of random matrix product with markovian noise: Application to linear stochastic approximation and td learning. In *COLT*.

Hong, M., Hajinezhad, D., and Zhao, M.-M. (2017). Prox-pda: The proximal primal-dual algorithm for fast distributed nonconvex optimization and learning over networks. In *International Conference on Machine Learning*, pages 1529–1538. PMLR.

Jakovetic, D., Xavier, J., and Moura, J. M. (2011). Cooperative convex optimization in networked systems: Augmented lagrangian algorithms with directed gossip communication. *IEEE Transactions on Signal Processing*, 59(8):3889–3902.

Karimi, B., Miasojedow, B., Moulines, E., and Wai, H.-T. (2019). Non-asymptotic analysis of biased stochastic approximation scheme. In *COLT*.

Koloskova, A., Stich, S., and Jaggi, M. (2019). Decentralized stochastic optimization and gossip algorithms with compressed communication. In *International Conference on Machine Learning*, pages 3478–3487. PMLR.

Lian, X., Zhang, C., Zhang, H., Hsieh, C.-J., Zhang, W., and Liu, J. (2017). Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent. In *NeurIPS*, pages 5330–5340.

Lu, S., Zhang, X., Sun, H., and Hong, M. (2019). Gnsd: a gradient-tracking based nonconvex stochastic algorithm for decentralized optimization. In *IEEE DSW*, pages 315–321.

Nedic, A., Olshevsky, A., and Shi, W. (2017). Achieving geometric convergence for distributed optimization over time-varying graphs. *SIAM Journal on Optimization*, 27(4):2597–2633.

Nedić, A. and Ozdaglar, A. (2009). Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1):48–61.

Nguyen, L. M., Liu, J., Scheinberg, K., and Takáč, M. (2017). Sarah: A novel method for machine learning problems using stochastic recursive gradient. In *International Conference on Machine Learning*, pages 2613–2621. PMLR.

Pan, T., Liu, J., and Wang, J. (2020). D-spider-sfo: A decentralized optimization algorithm with faster convergence rate for nonconvex problems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 1619–1626.

Pu, S., Shi, W., Xu, J., and Nedić, A. (2020). Push–pull gradient methods for distributed optimization in networks. *IEEE Transactions on Automatic Control*, 66(1):1–16.

Qu, G. and Li, N. (2017). Harnessing smoothness to accelerate distributed optimization. *IEEE Transactions on Control of Network Systems*, 5(3):1245–1260.

Scaman, K., Bach, F., Bubeck, S., Lee, Y. T., and Massoulié, L. (2017). Optimal algorithms for smooth and strongly convex distributed optimization in networks. In *international conference on machine learning*, pages 3027–3036. PMLR.

Scutari, G. and Sun, Y. (2019). Distributed nonconvex constrained optimization over time-varying digraphs. *Mathematical Programming*, 176(1):497–544.

Shi, W., Ling, Q., Wu, G., and Yin, W. (2015). Extra: An exact first-order algorithm for decentralized consensus optimization. *SIAM Journal on Optimization*, 25(2):944–966.

Tsitsiklis, J. N. (1984). Problems in decentralized decision making and computation. Technical report, Massachusetts Inst of Tech Cambridge Lab for Information and Decision Systems.

Turan, B., Uribe, C. A., Wai, H.-T., and Alizadeh, M. (2021). Robust distributed optimization with randomly corrupted gradients. *arXiv preprint arXiv:2106.14956*.

Uribe, C. A., Lee, S., Gasnikov, A., and Nedić, A. (2020). A dual approach for optimal algorithms in distributed optimization over networks. In *2020 Information Theory and Applications Workshop (ITA)*, pages 1–37. IEEE.

Vlaski, S. and Sayed, A. H. (2021). Distributed learning in non-convex environments—part i: Agreement at a linear rate. *IEEE Transactions on Signal Processing*, 69:1242–1256.

Wai, H.-T. (2020). On the convergence of consensus algorithms with markovian noise and gradient bias. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pages 4897–4902. IEEE.

Wai, H.-T., Lafond, J., Scaglione, A., and Moulines, E. (2017a). Decentralized frank–wolfe algorithm for convex and nonconvex problems. *IEEE Transactions on Automatic Control*, 62(11):5522–5537.

Wai, H.-T., Scaglione, A., Lafond, J., and Moulines, E. (2017b). Fast and privacy preserving distributed low-rank regression. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4451–4455. IEEE.

Wai, H.-T., Yang, Z., Wang, Z., and Hong, M. (2018). Multi-agent reinforcement learning via double averaging primal-dual optimization. In *NeurIPS*, pages 9649–9660.

Xi, C. and Khan, U. A. (2017). Dextra: A fast algorithm for optimization over directed graphs. *IEEE Transactions on Automatic Control*, 62(10):4980–4993.

Xin, R., Khan, U. A., and Kar, S. (2021). A hybrid variance-reduced method for decentralized stochastic non-convex optimization. In *ICML*.

Yang, Z. and Bajwa, W. U. (2019). Byrdie: Byzantine-resilient distributed coordinate descent for decentralized learning. *IEEE Transactions on Signal and Information Processing over Networks*, 5(4):611–627.

Zeng, J. and Yin, W. (2018). On nonconvex decentralized gradient descent. *IEEE Transactions on signal processing*, 66(11):2834–2848.