LINEAR CONVERGENT DECENTRALIZED OPTIMIZATION WITH COMPRESSION

Anonymous authors
Paper under double-blind review

ABSTRACT

Communication compression has become a key strategy to speed up distributed optimization. However, existing decentralized algorithms with compression mainly focus on compressing DGD-type algorithms. They are unsatisfactory in terms of convergence rate, stability, and the capability to handle heterogeneous data. Motivated by primal-dual algorithms, this paper proposes the first LinEAr convergent Decentralized algorithm with compression, LEAD. Our theory describes the coupled dynamics of the inexact primal and dual update as well as compression error, and we provide the first consensus error bound in such settings without assuming bounded gradients. Experiments on convex problems validate our theoretical analysis, and empirical study on deep neural nets shows that LEAD is applicable to non-convex problems.

1 INTRODUCTION

Distributed optimization solves the following optimization problem

$$\boldsymbol{x}^* := \arg\min_{\boldsymbol{x}\in\mathbb{R}^d} \left[f(\boldsymbol{x}) := \frac{1}{n}\sum_{i=1}^{n} f_i(\boldsymbol{x}) \right] \tag{1}$$

with n computing agents and a communication network. Each $f_i(\boldsymbol{x}) : \mathbb{R}^d \to \mathbb{R}$ is a local objective function of agent i and typically defined on the data $\mathcal{D}_i$ settled at that agent. The data distributions $\{\mathcal{D}_i\}$ can be heterogeneous depending on the applications such as in federated learning. The variable $\boldsymbol{x} \in \mathbb{R}^d$ often represents model parameters in machine learning. A distributed optimization algorithm seeks an optimal solution that minimizes the overall objective function $f(\boldsymbol{x})$ collectively. According to the communication topology, existing algorithms can be conceptually categorized into centralized and decentralized ones. Specifically, centralized algorithms require global communication between agents (through central agents or parameter servers). While decentralized algorithms only require local communication between connected agents and are more widely applicable than centralized ones. In both paradigms, the computation can be relatively fast with powerful computing devices; efficient communication is the key to improve algorithm efficiency and system scalability, especially when the network bandwidth is limited.

In recent years, various communication compression techniques, such as quantization and sparsification, have been developed to reduce communication costs. Notably, extensive studies (Seide et al., 2014; Alistarh et al., 2017; Bernstein et al., 2018; Stich et al., 2018; Karimireddy et al., 2019; Mishchenko et al., 2019; Tang et al., 2019b; Liu et al., 2020) have utilized gradient compression to significantly boost communication efficiency for centralized optimization. They enable efficient large-scale optimization while maintaining comparable convergence rates and practical performance with their non-compressed counterparts. This great success has suggested the potential and significance of communication compression in decentralized algorithms.

While extensive attention has been paid to centralized optimization, communication compression is relatively less studied in decentralized algorithms because the algorithm design and analysis are more challenging in order to cover general communication topologies. There are recent efforts trying to push this research direction. For instance, DCD-SGD and ECD-SGD (Tang et al., 2018a) introduce difference compression and extrapolation compression to reduce model compression error. (Reisizadeh et al., 2019a;b) introduce QDGD and QuanTimed-DSGD to achieve exact convergence

with small stepsize. DeepSqueeze (Tang et al., 2019a) directly compresses the local model and compensates the compression error in the next iteration. CHOCO-SGD (Koloskova et al., 2019; 2020) presents a novel quantized gossip algorithm that reduces compression error by difference compression and preserves the model average. Nevertheless, most existing works focus on the compression of primal-only algorithms, i.e., reduce to DGD (Nedic & Ozdaglar, 2009; Yuan et al., 2016) or P-DSGD (Lian et al., 2017). They are unsatisfying in terms of convergence rate, stability, and the capability to handle heterogeneous data. Part of the reason is that they inherit the drawback of DGD-type algorithms, whose convergence rate is slow in heterogeneous data scenarios where the data distributions are significantly different from agent to agent.

In the literature of decentralized optimization, it has been proved that primal-dual algorithms can achieve faster converge rates and better support heterogeneous data (Ling et al., 2015; Shi et al., 2015; Li et al., 2019; Yuan et al., 2020). However, it is unknown whether communication compression is feasible for primal-dual algorithms and how fast the convergence can achieve with compression. In this paper, we attempt to bridge this gap by investigating the communication compression for primal-dual decentralized algorithms. Our major contributions can be summarized as:

- We delineate two key challenges in the algorithm design for communication compression in decentralized optimization, i.e., data heterogeneity and compression error, and motivated by primal-dual algorithms, we propose a novel decentralized algorithm with compression, LEAD.
- We prove that for LEAD, a constant stepsize in the range $(0, 2/(\mu + L)]$ is sufficient to ensure linear convergence for strongly convex and smooth objective functions. To the best of our knowledge, LEAD is the first linear convergent decentralized algorithm with compression. Moreover, LEAD provably works with unbiased compression with arbitrary precision.
- We further prove that if the stochastic gradient is used, LEAD converges linearly to the $O(\sigma^2)$ neighborhood of the optimum with constant stepsize. LEAD is also able to achieve exact convergence to the optimum with diminishing stepsize.
- Extensive experiments on convex problems validate our theoretical analysis, and the empirical study on training deep neural nets shows that LEAD is applicable for nonconvex problems. LEAD achieves state-of-art computation and communication efficiency in all experiments and significantly outperforms the baselines on heterogeneous data. Moreover, LEAD is robust to parameter settings and needs minor effort for parameter tuning.

2 Related Works

Decentralized optimization can be traced back to the work by Tsitsiklis et al. (1986). DGD (Nedic & Ozdaglar, 2009) is the most classical decentralized algorithm. It is intuitive and simple but converges slowly due to the diminishing stepsize that is needed to obtain the optimal solution (Yuan et al., 2016). Its stochastic version D-PSGD (Lian et al., 2017) has been shown effective for training nonconvex deep learning models. Algorithms based on primal-dual formulations or gradient tracking are proposed to eliminate the convergence bias in DGD-type algorithms and improve the convergence rate, such as D-ADMM (Mota et al., 2013), DLM (Ling et al., 2015), EXTRA (Shi et al., 2015), NIDS (Li et al., 2019), D^2 (Tang et al., 2018b), Exact Diffusion (Yuan et al., 2018), OPTRA(Xu et al., 2020), DIGing (Nedic et al., 2017), GSGT (Pu & Nedić, 2020), etc.

Recently, communication compression is applied to decentralized settings by Tang et al. (2018a). It proposes two algorithms, i.e., DCD-SGD and ECD-SGD, which require compression of high accuracy and are not stable with aggressive compression. (Reisizadeh et al., 2019a;b) introduce QDGD and QuanTimed-DSGD to achieve exact convergence with small stepsize and the convergence is slow. DeepSqueeze (Tang et al., 2019a) compensates the compression error to the compression in the next iteration. Motivated by the quantized average consensus algorithms, such as (Carli et al., 2010), the quantized gossip algorithm CHOCO-Gossip (Koloskova et al., 2019) converges linearly to the consensual solution. Combining CHOCO-Gossip and D-PSGD leads to a decentralized algorithm with compression, CHOCO-SGD, which converges sublinearly under the strong convexity and gradient boundedness assumptions. Its nonconvex variant is further provided in (Koloskova et al., 2020). A new compression scheme using the modulo operation is introduced in (Lu & De Sa, 2020) for decentralized optimization.

3 ALGORITHM

We first introduce notations and definitions used in this work. We use bold upper-case letters such as $\mathbf{X}$ to define matrices and bold lower-case letters such as $\boldsymbol{x}$ to define vectors. Let $\mathbf{1}$ and $\mathbf{0}$ be vectors with all ones and zeros, respectively. Their dimensions will be provided when necessary. Given two matrices $\mathbf{X},\ \mathbf{Y} \in \mathbb{R}^{n\times d}$, we define their inner product as $\langle \mathbf{X}, \mathbf{Y}\rangle = \mathrm{tr}(\mathbf{X}^\top \mathbf{Y})$ and the norm as $\|\mathbf{X}\| = \sqrt{\langle \mathbf{X}, \mathbf{X}\rangle}$. We further define $\langle \mathbf{X}, \mathbf{Y}\rangle_{\mathbf{P}} = \mathrm{tr}(\mathbf{X}^\top \mathbf{P}\mathbf{Y})$ and $\|\mathbf{X}\|_{\mathbf{P}} = \sqrt{\langle \mathbf{X}, \mathbf{X}\rangle_{\mathbf{P}}}$ for any given symmetric positive semidefinite matrix $\mathbf{P} \in \mathbb{R}^{n\times n}$. For simplicity, we will majorly use the matrix notation in this work. For instance, each agent i holds an individual estimate $\boldsymbol{x}_i \in \mathbb{R}^d$ of the global variable $\boldsymbol{x} \in \mathbb{R}^d$. Let $\mathbf{X}^k$ and $\nabla \mathbf{F}(\mathbf{X}^k)$ be the collections of $\{\boldsymbol{x}_i^k\}_{i=1}^n$ and $\{\nabla f_i(\boldsymbol{x}_i^k)\}_{i=1}^n$ which are defined below:

$$\mathbf{X}^k = \left[\boldsymbol{x}_1^k, \dots, \boldsymbol{x}_n^k\right]^\top \in \mathbb{R}^{n\times d}, \quad \nabla \mathbf{F}(\mathbf{X}^k) = \left[\nabla f_1(\boldsymbol{x}_1^k), \dots, \nabla f_n(\boldsymbol{x}_n^k)\right]^\top \in \mathbb{R}^{n\times d}. \tag{2}$$

We use $\nabla \mathbf{F}(\mathbf{X}^k; \xi^k)$ to denote the stochastic approximation of $\nabla \mathbf{F}(\mathbf{X}^k)$. With these notations, the update $\mathbf{X}^{k+1} = \mathbf{X}^k - \eta \nabla \mathbf{F}(\mathbf{X}^k; \xi^k)$ means that $\boldsymbol{x}_i^{k+1} = \boldsymbol{x}_i^k - \eta \nabla f_i(\boldsymbol{x}_i^k; \xi_i^k)$ for all i. In this paper, we need the average of all rows in $\mathbf{X}^k$ and $\nabla F(\mathbf{X}^k)$, so we define $\overline{\mathbf{X}}^k = (\mathbf{1}^\top \mathbf{X}^k)/n$ and $\overline{\nabla} \mathbf{F}(\mathbf{X}^k) = (\mathbf{1}^\top \nabla \mathbf{F}(\mathbf{X}^k))/n$. They are row vectors, and we will take a transpose if we need a column vector. The pseudoinverse of a matrix $\mathbf{M}$ is denoted as $\mathbf{M}^\dagger$. The largest, ith-largest, and smallest nonzero eigenvalues of a symmetric matrix $\mathbf{M}$ are $\lambda_{\max}(\mathbf{M})$, $\lambda_i(\mathbf{M})$, and $\lambda_{\min}(\mathbf{M})$.

Assumption 1 (Mixing matrix)**.** *The connected network $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ consists of a node set $\mathcal{V} = \{1, 2, \dots, n\}$ and an undirected edge set $\mathcal{E}$. The primitive symmetric doubly-stochastic matrix $\mathbf{W} = [w_{ij}] \in \mathbb{R}^{n\times n}$ encodes the network structure such that $w_{ij} = 0$ if nodes i and j are not connected and cannot exchange information.*

Assumption 1 implies that $-1 < \lambda_n(\mathbf{W}) \le \lambda_{n-1}(\mathbf{W}) \le \cdots \lambda_2(\mathbf{W}) \le \lambda_1(\mathbf{W}) = 1$ and $\mathbf{W}\mathbf{1} = \mathbf{1}$ (Xiao & Boyd, 2004; Shi et al., 2015). The matrix multiplication $\mathbf{X}^{k+1} = \mathbf{W}\mathbf{X}^k$ describes that agent i takes a weighted sum from its neighbors and itself, i.e., $\boldsymbol{x}_i^{k+1} = \sum_{j\in\mathcal{N}_i\cup\{i\}} w_{ij}\boldsymbol{x}_j^k$, where $\mathcal{N}_i$ denotes the neighbors of agent i.

3.1 THE PROPOSED ALGORITHM

The proposed algorithm LEAD to solve problem (1) is showed in Alg. 1 with matrix notations for conciseness. We will refer to the line number in the analysis. A complete algorithm description from the agent's perspective can be found in Appendix A. The motivation behind Alg. 1 is to achieve two goals: (a) consensus ($\boldsymbol{x}_i^k - (\overline{\mathbf{X}}^k)^\top \to \mathbf{0}$) and (b) convergence ($(\overline{\mathbf{X}}^k)^\top \to \boldsymbol{x}^*$). We first discuss how goal (a) leads to goal (b) and then explain how LEAD fulfills goal (a).

Algorithm 1 LEAD

Input: Stepsize η, parameter $(\alpha,\ \gamma)$, $\mathbf{X}^0,\ \mathbf{H}^1,\ \mathbf{D}^1 = (\mathbf{I} - \mathbf{W})\mathbf{Z}$ for any $\mathbf{Z}$

Output: $\mathbf{X}^K$ or $1/n\sum_{i=1}^n \mathbf{X}_i^K$

1: $\mathbf{H}_w^1 = \mathbf{W}\mathbf{H}^1$
2: $\mathbf{X}^1 = \mathbf{X}^0 - \eta\nabla\mathbf{F}(\mathbf{X}^0; \xi^0)$
3: **for** $k = 1, 2, \cdots, K-1$ **do**
4: $\quad \mathbf{Y}^k = \mathbf{X}^k - \eta\nabla\mathbf{F}(\mathbf{X}^k; \xi^k) - \eta\mathbf{D}^k$
5: $\quad \hat{\mathbf{Y}}^k, \hat{\mathbf{Y}}_w^k, \mathbf{H}^{k+1}, \mathbf{H}_w^{k+1} = \textsc{Comm}(\mathbf{Y}^k, \mathbf{H}^k, \mathbf{H}_w^k)$
6: $\quad \mathbf{D}^{k+1} = \mathbf{D}^k + \frac{\gamma}{2\eta}(\hat{\mathbf{Y}}^k - \hat{\mathbf{Y}}_w^k)$
7: $\quad \mathbf{X}^{k+1} = \mathbf{X}^k - \eta\nabla\mathbf{F}(\mathbf{X}^k; \xi^k) - \eta\mathbf{D}^{k+1}$
8: **end for**
9: **procedure** $\textsc{Comm}(\mathbf{Y}, \mathbf{H}, \mathbf{H}_w)$
10: $\quad \mathbf{Q} = \textsc{Compress}(\mathbf{Y} - \mathbf{H})$
11: $\quad \hat{\mathbf{Y}} = \mathbf{H} + \mathbf{Q}$
12: $\quad \hat{\mathbf{Y}}_w = \mathbf{H}_w + \mathbf{W}\mathbf{Q}$
13: $\quad \mathbf{H} = (1-\alpha)\mathbf{H} + \alpha\hat{\mathbf{Y}}$
14: $\quad \mathbf{H}_w = (1-\alpha)\mathbf{H}_w + \alpha\hat{\mathbf{Y}}_w$
15: $\quad$ **Return:** $\hat{\mathbf{Y}}, \hat{\mathbf{Y}}_w, \mathbf{H}, \mathbf{H}_w$
16: **end procedure**

In essence, LEAD runs the approximate SGD globally and reduces to the exact SGD under consensus. One key property for LEAD is $\mathbf{1}_{n\times 1}^\top \mathbf{D}^k = \mathbf{0}$, regardless of the compression error in $\hat{\mathbf{Y}}^k$. It holds because that for the initialization, we require $\mathbf{D}^1 = (\mathbf{I} - \mathbf{W})\mathbf{Z}$ for some $\mathbf{Z} \in \mathbb{R}^{n\times d}$, e.g., $\mathbf{D}^1 = \mathbf{0}^{n\times d}$, and that the update of $\mathbf{D}^k$ ensures $\mathbf{D}^k \in \mathbf{Range}(\mathbf{I} - \mathbf{W})$ for all k and

$\mathbf{1}_{n\times 1}^\top(\mathbf{I}-\mathbf{W})=\mathbf{0}$ as we will explain later. Therefore, multiplying $(1/n)\mathbf{1}_{n\times 1}^\top$ on both sides of Line 7 leads to a global average view of Alg. 1:

$$\overline{\mathbf{X}}^{k+1}=\overline{\mathbf{X}}^k-\eta\overline{\nabla}\mathbf{F}(\mathbf{X}^k;\xi^k), \tag{3}$$

which doesn't contain the compression error. Note that this is an approximate SGD step because, as shown in (2), the gradient $\nabla\mathbf{F}(\mathbf{X}^k;\xi^k)$ is not evaluated on a global synchronized model $\overline{\mathbf{X}}^k$. However, if the solution converges to the consensus solution, i.e., $\boldsymbol{x}_i^k-(\overline{\mathbf{X}}^k)^\top\to\mathbf{0}$, then $\mathbb{E}_{\xi^k}[\overline{\nabla}\mathbf{F}(\mathbf{X}^k;\xi^k)-\nabla f(\overline{\mathbf{X}}^k;\xi^k)]\to\mathbf{0}$ and (3) gradually reduces to exact SGD.

With the establishment of how consensus leads to convergence, the obstacle becomes how to achieve consensus under local communication and compression challenges. It requires addressing two issues, i.e., data heterogeneity and compression error. Existing algorithms, such as DCD-SGD, ECD-SGD, QDGD, DeepSqueeze, Moniqua, and CHOCO-SGD need diminishing or a constant but small stepsize depending on the total number of iterations to deal with these issues. However, these choices unavoidably cause slower convergence and bring in the difficulty of parameter tuning. In contrast, LEAD takes a different way to solve these issues, as explained below.

Data heterogeneity. It is common in distributed settings that there exists data heterogeneity among agents, especially in real-world applications where different agents collect data from different scenarios. In other words, we generally have $f_i(\boldsymbol{x})\neq f_j(\boldsymbol{x})$ for $i\neq j$. The optimality condition of problem (1) gives $\mathbf{1}_{n\times 1}^\top\nabla\mathbf{F}(\mathbf{X}^*)=\mathbf{0}$, where $\mathbf{X}^*=[\boldsymbol{x}^*,\cdots,\boldsymbol{x}^*]$ is a consensual and optimal solution. The data heterogeneity and optimality condition imply that there exist at least two agents i and j such that $\nabla f_i(\boldsymbol{x}^*)\neq\mathbf{0}$ and $\nabla f_j(\boldsymbol{x}^*)\neq\mathbf{0}$. As a result, a simple D-PSGD algorithm cannot converge to the consensual and optimal solution as $\mathbf{X}^*\neq\mathbf{W}\mathbf{X}^*-\eta\mathbb{E}_\xi\nabla\mathbf{F}(\mathbf{X}^*;\xi)$ even when the stochastic gradient variance is zero.

Gradient correction. Primal-dual algorithms or gradient tracking algorithms can handle the data heterogeneity issue to achieve a much faster convergence rate than DGD-type algorithms, as introduced in Section 2. Specifically, LEAD is motivated by the design of primal-dual algorithm NIDS (Li et al., 2019; Li & Yan, 2019), and the connection is established in Appendix B. $\mathbf{X}^k$ and $\mathbf{D}^k$ in Alg. 1 are the primal and dual variables respectively[1]. The dual variable $\mathbf{D}^k$ plays the role of gradient correction. As $k\to\infty$, we expect $\mathbf{D}^k\to-\nabla\mathbf{F}(\mathbf{X}^*)$ and $\mathbf{X}^k$ will converge to $\mathbf{X}^*$ in expectation via the update $\mathbf{X}^{k+1}=\mathbf{X}^k-\eta\nabla\mathbf{F}(\mathbf{X}^k;\xi)-\eta\mathbf{D}^{k+1}$ (Line 7) since $\mathbf{D}^{k+1}$ corrects the nonzero gradient $\nabla\mathbf{F}(\mathbf{X}^k)$ asymptotically. Nevertheless, how to compress the communication and how fast the convergence we can attain with compression error are unknown. In the following, we propose to carefully control the compression error by difference compression and error compensation such that the inexact dual update (Line 6) and primal update (Line 7) can still guarantee the convergence as proved in Section 4.

Compression error. Different from existing works, which typically compress the primal variable $\mathbf{X}^k$ or its difference, LEAD first construct an intermediate variable $\mathbf{Y}^k$ and apply compression to obtain its coarse representation $\hat{\mathbf{Y}}^k$ as shown in the procedure $\textsc{Comm}(\mathbf{Y},\mathbf{H},\mathbf{H}_w)$:

- Compress the difference between $\mathbf{Y}$ and the state variable $\mathbf{H}$ as $\mathbf{Q}$;
- $\mathbf{Q}$ is encoded into the low-bit representation, which enables the efficient local communication step $\hat{\mathbf{Y}}_w=\mathbf{H}_w+\mathbf{W}\mathbf{Q}$. It is *the only communication step* in each iteration.
- Each agent recovers its estimate $\hat{\mathbf{Y}}$ by $\hat{\mathbf{Y}}=\mathbf{H}+\mathbf{Q}$ and we have $\hat{\mathbf{Y}}_w=\mathbf{W}\hat{\mathbf{Y}}$.
- States $\mathbf{H}$ and $\mathbf{H}_w$ are updated based on $\hat{\mathbf{Y}}$ and $\hat{\mathbf{Y}}_w$, respectively. We have $\mathbf{H}_w=\mathbf{W}\mathbf{H}$.

By this procedure, we expect when both $\mathbf{Y}^k$ and $\mathbf{H}^k$ converge to $\mathbf{X}^*$, the compression error vanishes asymptotically due to the assumption we make for the compression operator in Assumption 2.

Remark 1. *Note that difference compression is also applied in DCD-PSGD (Tang et al., 2018a) and CHOCO-SGD (Koloskova et al., 2019), but their state update is the simple integration of the compressed difference. We find this update is usually too aggressive and cause instability as showed in our experiments. Therefore, we adopt a momentum update* $\mathbf{H}=(1-\alpha)\mathbf{H}+\alpha\hat{\mathbf{Y}}$ *motivated from DIANA (Mishchenko et al., 2019), which reduces the compression error for gradient compression in centralized optimization.*

[1]Precisely speaking, $\mathbf{D}^k$ is a linear transformation of the dual variable in NIDS, but we call $\mathbf{D}^k$ the dual variable with a little bit abuse of notation.

Error compensation. On the other hand, even if the compression error exists, LEAD essentially compensates for the error in the inexact dual update (Line 6), making the algorithm more stable and robust. To illustrate how it works, let $\mathbf{E}^k = \hat{\mathbf{Y}}^k - \mathbf{Y}^k$ denote the compression error and $\boldsymbol{e}_i^k$ be its i-th row. The update of $\mathbf{D}^k$ gives

$$\mathbf{D}^{k+1} = \mathbf{D}^k + \frac{\gamma}{2\eta}(\hat{\mathbf{Y}}^k - \hat{\mathbf{Y}}_w^k) = \mathbf{D}^k + \frac{\gamma}{2\eta}(\mathbf{I} - \mathbf{W})\mathbf{Y}^k + \frac{\gamma}{2\eta}(\mathbf{E}^k - \mathbf{W}\mathbf{E}^k)$$

where $-\mathbf{W}\mathbf{E}^k$ indicates that agent i spreads total compression error $-\sum_{j\in\mathcal{N}_i\cup\{i\}} w_{ji}\boldsymbol{e}_i^k = -\boldsymbol{e}_i^k$ to all agents and $\mathbf{E}^k$ indicates that each agent compensates this error locally by adding $\boldsymbol{e}_i^k$ back. This error compensation also explains why the global view in (3) doesn't involve compression error.

Remark 2. *Note that in LEAD, the compression error is compensated into the model $\mathbf{X}^{k+1}$ through Line 6 and Line 7 such that the gradient computation in the next iteration is aware of the compression error. This has some subtle but important difference from the error compensation or error feedback in (Seide et al., 2014; Wu et al., 2018; Stich et al., 2018; Karimireddy et al., 2019; Tang et al., 2019b; Liu et al., 2020; Tang et al., 2019a), where the error is stored in the memory and only compensated after gradient computation and before the compression.*

4 Theoretical Analysis

In this section, we show the convergence rate for the proposed algorithm LEAD. Before showing the main theorem, we make some assumptions, which are commonly used for the analysis of decentralized optimization algorithms. All proofs are provided in Appendix E.

Assumption 2 (Unbiased and C-contracted operator)**.** *The compression operator $Q : \mathbb{R}^d \to \mathbb{R}^d$ is unbiased, i.e., $\mathbb{E}Q(\boldsymbol{x}) = \boldsymbol{x}$, and there exists $C \geq 0$ such that $\mathbb{E}\|\boldsymbol{x} - Q(\boldsymbol{x})\|_2^2 \leq C\|\boldsymbol{x}\|_2^2$ for all $\boldsymbol{x} \in \mathbb{R}^d$.*

Assumption 3 (Stochastic gradient)**.** *The stochastic gradient $\nabla f_i(\boldsymbol{x};\xi)$ is unbiased, i.e., $\mathbb{E}_\xi \nabla f_i(\boldsymbol{x};\xi) = \nabla f_i(\boldsymbol{x})$, and the stochastic gradient variance is bounded: $\mathbb{E}_\xi\|\nabla f_i(\boldsymbol{x};\xi) - \nabla f_i(\boldsymbol{x})\|_2^2 \leq \sigma_i^2$ for all $i \in [n]$. Denote $\sigma^2 = \frac{1}{n}\sum_{i=1}^n \sigma_i^2$.*

Assumption 4. *Each f_i is L-smooth and μ-strongly convex with $L \geq \mu > 0$, i.e., for $i = 1, 2, \ldots, n$ and $\forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$, we have*

$$f_i(\boldsymbol{y}) + \langle \nabla f_i(\boldsymbol{y}), \boldsymbol{x} - \boldsymbol{y}\rangle + \frac{\mu}{2}\|\boldsymbol{x} - \boldsymbol{y}\|^2 \leq f_i(\boldsymbol{x}) \leq f_i(\boldsymbol{y}) + \langle \nabla f_i(\boldsymbol{y}), \boldsymbol{x} - \boldsymbol{y}\rangle + \frac{L}{2}\|\boldsymbol{x} - \boldsymbol{y}\|^2.$$

Theorem 1 (Constant stepsize)**.** *Let $\{\mathbf{X}^k, \mathbf{H}^k, \mathbf{D}^k\}$ be the sequence generated from Alg. 1 and $\mathbf{X}^*$ is the optimal solution with $\mathbf{D}^* = -\nabla\mathbf{F}(\mathbf{X}^*)$. Under Assumptions 1-4, for any constant stepsize $\eta \in (0, 2/(\mu + L)]$, if the compression parameters α and γ satisfy*

$$\gamma \in \left(0, \min\left\{\frac{2}{(3C+1)\beta}, \frac{2\mu\eta(2-\mu\eta)}{[2-\mu\eta(2-\mu\eta)]C\beta}\right\}\right), \tag{4}$$

$$\alpha \in \left[\frac{C\beta\gamma}{2(1+C)}, \frac{1}{a_1}\min\left\{\frac{2-\beta\gamma}{4-\beta\gamma}, \mu\eta(2-\mu\eta)\right\}\right], \tag{5}$$

with $\beta := \lambda_{\max}(\mathbf{I} - \mathbf{W})$. Then, in total expectation we have

$$\frac{1}{n}\mathbb{E}\mathcal{L}^{k+1} \leq \rho\frac{1}{n}\mathbb{E}\mathcal{L}^k + \eta^2\sigma^2, \tag{6}$$

where

$$\mathcal{L}^k := (1 - a_1\alpha)\|\mathbf{X}^k - \mathbf{X}^*\|^2 + (2\eta^2/\gamma)\mathbb{E}\|\mathbf{D}^{k+1} - \mathbf{D}^*\|^2_{(\mathbf{I}-\mathbf{W})^\dagger} + a_1\|\mathbf{H}^k - \mathbf{X}^*\|^2,$$

$$\rho := \max\left\{\frac{1-\mu\eta(2-\mu\eta)}{1-a_1\alpha}, 1 - \frac{\gamma}{2\lambda_{\max}((\mathbf{I}-\mathbf{W})^\dagger)},\ 1-\alpha\right\} < 1, a_1 := \frac{4(1+C)}{C\beta\gamma + 2}$$

The result holds for $C \to 0$.

Corollary 1 (Complexity bounds)**.** *Define the condition numbers of the objective function and communication graph as $\kappa_f = \frac{L}{\mu}$ and $\kappa_g = \frac{\lambda_{\max}(\mathbf{I}-\mathbf{W})}{\lambda^+_{\min}(\mathbf{I}-\mathbf{W})}$, respectively. Under the same setting in Theorem 1, we can choose $\eta = \frac{1}{L}, \gamma = \min\{\frac{1}{C\beta\kappa_f}, \frac{1}{(1+3C)\beta}\}$, and $\alpha = \mathcal{O}(\frac{1}{(1+C)\kappa_f})$ such that*

$$\rho = \max\left\{1 - \mathcal{O}\Big(\frac{1}{(1+C)\kappa_f}\Big), 1 - \mathcal{O}\Big(\frac{1}{(1+C)\kappa_g}\Big), 1 - \mathcal{O}\Big(\frac{1}{C\kappa_f\kappa_g}\Big)\right\}.$$

With full-gradient (i.e., $\sigma = 0$), we obtain the following complexity bounds:

- *LEAD converges to the ϵ-accurate solution with the iteration complexity*

$$\mathcal{O}\Big(\big((1+C)(\kappa_f+\kappa_g)+C\kappa_f\kappa_g\big)\log\frac{1}{\epsilon}\Big).$$

- *When $C=0$ (i.e., there is no compression), we obtain $\rho=\max\{1-\mathcal{O}(\frac{1}{\kappa_f}),1-\mathcal{O}(\frac{1}{\kappa_g})\}$, and the iteration complexity $\mathcal{O}\Big((\kappa_f+\kappa_g)\log\frac{1}{\epsilon}\Big)$. This exactly recovers the convergence rate of NIDS (Li et al., 2019).*

- *When $C\leq\frac{\kappa_f+\kappa_g}{\kappa_f\kappa_g+\kappa_f+\kappa_g}$, the asymptotical complexity is $\mathcal{O}\Big((\kappa_f+\kappa_g)\log\frac{1}{\epsilon}\Big)$, which also recovers that of NIDS (Li et al., 2019) and indicates that the compression doesn't harm the convergence in this case.*

- *With $C=0$ (or $C\leq\frac{\kappa_f+\kappa_g}{\kappa_f\kappa_g+\kappa_f+\kappa_g}$) and fully connected communication graph (i.e., $\mathbf{W}=\frac{\mathbf{1}\mathbf{1}^\top}{n}$), we have $\beta=1$ and $\kappa_g=1$. Therefore, we obtain $\rho=1-\mathcal{O}(\frac{1}{\kappa_f})$ and the complexity bound $\mathcal{O}(\kappa_f log\frac{1}{\epsilon})$. This recovers the convergence rate of gradient descent (Nesterov, 2013).*

Remark 3. *Under the setting in Theorem 1, LEAD converges linearly to the $\mathcal{O}(\sigma^2)$ neighborhood of the optimum and converges linearly exactly to the optimum if full gradient is used, e.g., $\sigma=0$. The linear convergence of LEAD holds when $\eta<2/L$, but we omit the proof.*

Remark 4. *Pick any $\eta\in(0,2/(\mu+L)]$, based on the compression-related constant C and the network-related constant β, we can select γ and α in certain ranges to achieve the convergence. It suggests that LEAD supports unbiased compression with arbitrary precision, i.e., any $C>0$.*

Corollary 2 (Consensus error)**.** *Under the same setting in Theorem 1, let $\overline{\boldsymbol{x}}^k=\frac{1}{n}\sum_{i=1}^n\boldsymbol{x}_i^k$ be the averaged model and $\mathbf{H}^0=\mathbf{H}^1$, then all agents achieve consensus at the rate*

$$\frac{1}{n}\sum_{i=1}^n\mathbb{E}\left\|\boldsymbol{x}_i^k-\overline{\boldsymbol{x}}^k\right\|^2\leq\frac{2\mathcal{L}^0}{n}\rho^k+\frac{2\sigma^2}{1-\rho}\eta^2. \tag{7}$$

where ρ is defined as in Corollary 1 with appropriate parameter settings.

Theorem 2 (Diminishing stepsize)**.** *Let $\{\mathbf{X}^k,\mathbf{H}^k,\mathbf{D}^k\}$ be the sequence generated from Alg. 1 and $\mathbf{X}^*$ is the optimal solution with $\mathbf{D}^*=-\nabla\mathbf{F}(\mathbf{X}^*)$. Under Assumptions 1-4, if $\eta_k=\frac{2\theta_5}{\theta_3\theta_4\theta_5k+2}$ and $\gamma_k=\theta_4\eta_k$, by taking $\alpha_k=\frac{C\beta\gamma_k}{2(1+C)}$, in total expectation we have*

$$\frac{1}{n}\sum_{i=1}^n\mathbb{E}\left\|\boldsymbol{x}_i^k-\boldsymbol{x}^*\right\|^2\lesssim\mathcal{O}\left(\frac{1}{k}\right) \tag{8}$$

where $\theta_1,\theta_2,\theta_3,\theta_4$ and θ_5 are constants defined in the proof. The complexity bound for arriving at the ϵ-accurate solution is $\mathcal{O}(\frac{1}{\epsilon})$.

Remark 5. *Compared with CHOCO-SGD, LEAD requires unbiased compression and the convergence under biased compression is not investigated yet. The analysis of CHOCO-SGD relies on the bounded gradient assumptions, i.e., $\|\nabla f_i(\boldsymbol{x})\|^2\leq G$, which is restrictive because it conflicts with the strongly convexity while LEAD doesn't need this assumption. Moreover, in the theorem of CHOCO-SGD, it requires a specific point set of γ while LEAD only requires γ to be within a rather large range. This may explains the advantages of LEAD over CHOCO-SGD in terms of robustness to parameter setting.*

5 Numerical Experiments

We consider three machine learning problems – ℓ_2-regularized linear regression, logistic regression, and deep neural network. The proposed LEAD is compared with QDGD (Reisizadeh et al., 2019a), DeepSqueeze (Tang et al., 2019a), CHOCO-SGD (Koloskova et al., 2019), and two non-compressed algorithms DGD (Yuan et al., 2016) and NIDS (Li et al., 2019).

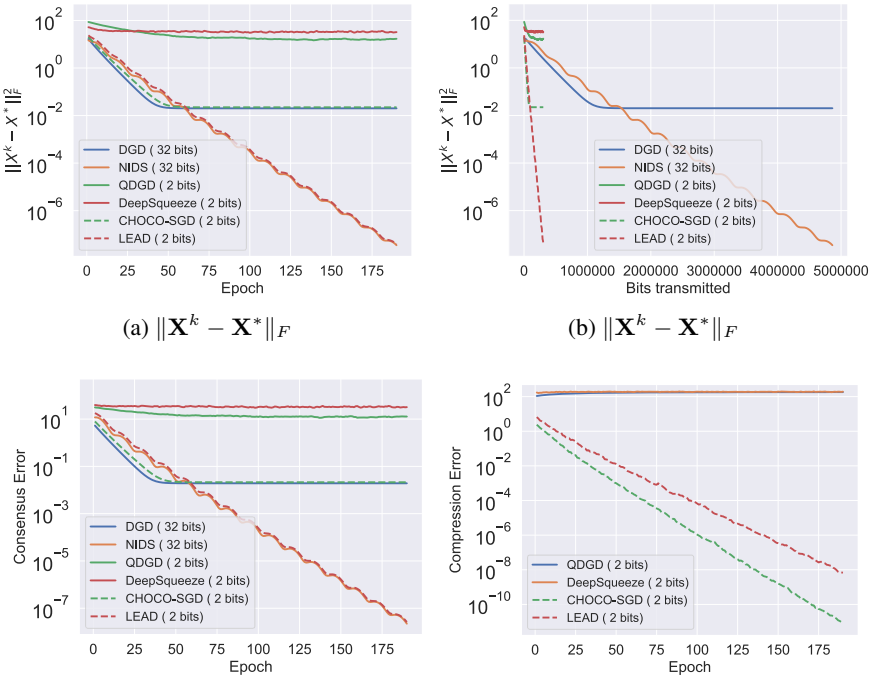
Setup. We consider eight machines connected in a ring topology network. Each agent can only exchange information with its two 1-hop neighbors. The mixing weight is simply set as $1/3$. For compression, we use the unbiased b-bits quantization method with ∞-norm

$$Q_\infty(\boldsymbol{x}) := \left(\|\boldsymbol{x}\|_\infty 2^{-(b-1)} \operatorname{sign}(\boldsymbol{x})\right) \cdot \left\lfloor \frac{2^{(b-1)}|\boldsymbol{x}|}{\|\boldsymbol{x}\|_\infty} + \boldsymbol{u} \right\rfloor, \tag{9}$$

where $\cdot$ is the Hadamard product, $|\boldsymbol{x}|$ is the elementwise absolute value of $\boldsymbol{x}$, and $\boldsymbol{u}$ is a random vector uniformly distributed in $[0,1]^d$. Only $\operatorname{sign}(\boldsymbol{x})$, norm $\|\boldsymbol{x}\|_\infty$, and integers in the bracket need to be transmitted. Note that this quantization method is similar to the quantization used in QSGD (Alistarh et al., 2017) and CHOCO-SGD (Koloskova et al., 2019), but we use the ∞-norm scaling instead of the 2-norm. This small change brings significant improvement on compression precision as justified both theoretically and empirically in Appendix C. In this section, we choose 2-bit quantization and quantize the data blockwise (block size = 512).

For all experiments, we tune the stepsize η from $\{0.01, 0.05, 0.1, 0.5\}$. For QDGD, CHOCO-SGD and Deepsqueeze, γ is tuned from $\{0.01, 0.1, 0.2, 0.4, 0.6, 0.8, 1.0\}$. Note that different notations are used in the original paper, here we uniformaly denote the stepsize as η and the additional parameter in these algorithms as γ for simplicity. For LEAD, we simply fix $\alpha = 0.5$ and $\gamma = 1.0$ for all experiments since we find LEAD is robust to parameter settings as we validate in the parameter sensitivity analysis in Appendix D.1. This indicates the minor effort needed for tuning LEAD. Detailed parameter settings for all experiments are summarized in Appendix D.3.

Linear regression. We consider the problem: $f(\boldsymbol{x}) = \sum_{i=1}^n (\|\mathbf{A}_i \boldsymbol{x} - \boldsymbol{b}_i\|^2 + \lambda \|\boldsymbol{x}\|^2)$. Data matrices $\mathbf{A}_i \in \mathbb{R}^{200\times 200}$ and the true solution $\boldsymbol{x}'$ is randomly synthesized. The values $\boldsymbol{b}_i$ are generated by adding Gaussian noise to $\mathbf{A}_i \boldsymbol{x}'$. We let $\lambda = 0.1$ and the optimal solution of the linear regression problem be $\boldsymbol{x}^*$. We use full-batch gradient to exclude the impact of gradient variance. The performance is showed in Fig. 1. The distance to $\boldsymbol{x}^*$ in Fig. 1a and the consensus error in Fig. 1c verify that LEAD converges exponentially to the optimal consensual solution. It significantly outperforms most baselines and matches NIDS well under the same number of iterations. Fig. 1b demonstrates the benefit of compression when considering the communication bits. Fig. 1d shows that the compression error vanishes for both LEAD and CHOCO-SGD while the compression error is pretty large for QDGD and DeepSqueeze because they directly compress the local models.

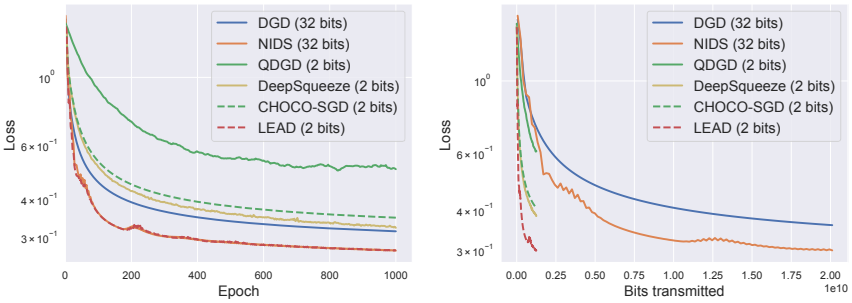


(a) $\|\mathbf{X}^k - \mathbf{X}^*\|_F$ (b) $\|\mathbf{X}^k - \mathbf{X}^*\|_F$

(c) $\|\mathbf{X}^k - \mathbf{1}_{n\times 1}\overline{\mathbf{X}}^k\|_F$ (d) Compression error

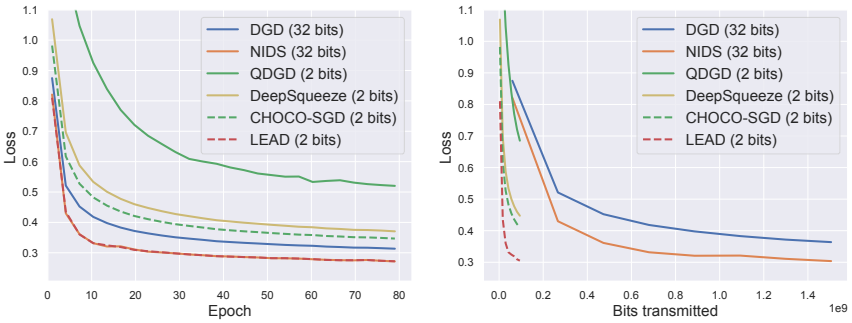
Figure 1: Linear regression problem.

Logistic regression. We further consider a logistic regression problem on the MNIST dataset. The regularization parameter is 10^{-4}. We consider both *homogeneous* and *heterogeneous* data settings. In the homogeneous setting, the data samples are randomly shuffled before being uniformly partitioned among all agents such that the data distribution from each agent is very similar. In the heterogeneous setting, the samples are first sorted by their labels and then partitioned among agents. Due to the space limit, we mainly present the results in heterogeneous setting here and defer the homogeneous setting to Appendix D.2. The results using full-batch gradient and mini-batch gradient (the mini-batch size is 512 for each agent) are showed in Fig. 2 and Fig. 3 respectively and both settings shows the faster convergence and higher precision of LEAD.



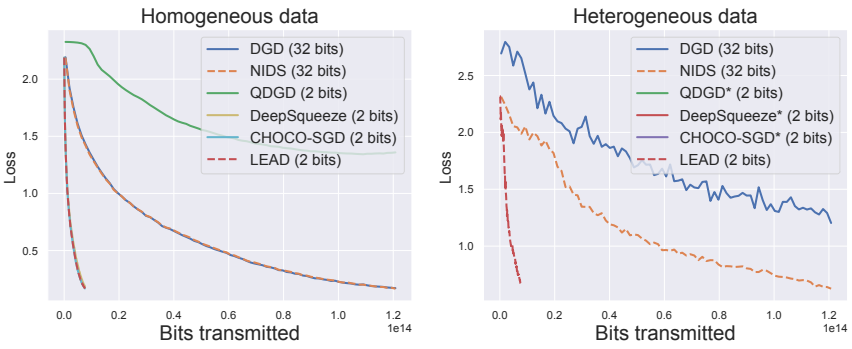
(a) Loss $f(\overline{\mathbf{X}}^k)$ (b) Loss $f(\overline{\mathbf{X}}^k)$

Figure 2: Logistic regression problem in the heterogeneous case (full-batch gradient).



(a) Loss $f(\overline{\mathbf{X}}^k)$ (b) Loss $f(\overline{\mathbf{X}}^k)$

Figure 3: Logistic regression in the heterogeneous case (mini-batch gradient).



(a) Loss $f(\overline{\mathbf{X}}^k)$ (b) Loss $f(\overline{\mathbf{X}}^k)$

Figure 4: Stochastic optimization on deep neural network ($*$ means divergence).

Neural network. We empirically study the performance of LEAD in optimizing deep neural network by training AlexNet (240 MB) on CIFAR10 dataset. The mini-batch size is 64 for each agents. Both the homogeneous and heterogeneous case are showed in Fig. 4. In the homogeneous case, CHOCO-SGD, DeepSqueeze and LEAD perform similarly and outperform the non-compressed

variants in terms of communication efficiency, but CHOCO-SGD and DeepSqueeze need more efforts for parameter tuning because their convergence is sensitive to the setting of γ. In the heterogeneous cases, LEAD achieves the fastest and stable convergence. Note that in this setting, sufficient information exchange is more important for convergence because models from different agents are moving to significantly diverse directions. In such case, DGD only converges with smaller stepsize and its communication compressed variants, including QDGD, DeepSqueeze and CHOCO-SGD, diverge in all parameter settings we try.

In summary, our experiments verify our theoretical analysis and show that LEAD is able to handle data heterogeneity very well. Furthermore, the performance of LEAD is robust to parameter settings and needs less effort for parameter tuning, which is critical in real-world applications.

6 Conclusion

In this paper, we investigate the communication compression in decentralized optimization. Motivated by primal-dual algorithms, a novel decentralized algorithm with compression, LEAD, is proposed to achieve faster convergence rate and to better handle heterogeneous data while enjoying the benefit of efficient communication. The nontrivial analyses on the coupled dynamics of inexact primal and dual updates as well as compression error establish the linear convergence of LEAD when full gradient is used and the linear convergence to the $O(\sigma^2)$ neighborhood of the optimum when stochastic gradient is used. Extensive experiments validate the theoretical analysis and demonstrate the state-of-the-art efficiency and robustness of LEAD. LEAD is also applicable to non-convex problems as empirically verified in the neural network experiments but we leave the non-convex analysis as future work.

REFERENCES

Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. QSGD: Communication-efficient sgd via gradient quantization and encoding. In *Advances in Neural Information Processing Systems*, pp. 1709–1720. 2017.

Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Animashree Anandkumar. SIGNSGD: compressed optimisation for non-convex problems. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 559–568, 2018.

Ruggero Carli, Fabio Fagnani, Paolo Frasca, and Sandro Zampieri. Gossip consensus algorithms via quantized communication. *Automatica*, 46(1):70–80, 2010.

Sai Praneeth Karimireddy, Quentin Rebjock, Sebastian Urban Stich, and Martin Jaggi. Error feedback fixes SignSGD and other gradient compression schemes. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 3252–3261. PMLR, 2019.

Anastasia Koloskova, Sebastian U. Stich, and Martin Jaggi. Decentralized stochastic optimization and gossip algorithms with compressed communication. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 3479–3487. PMLR, 2019.

Anastasia Koloskova, Tao Lin, Sebastian U Stich, and Martin Jaggi. Decentralized deep learning with arbitrary communication compression. In *International Conference on Learning Representations*, 2020.

Yao Li and Ming Yan. On linear convergence of two decentralized algorithms. *arXiv preprint arXiv:1906.07225*, 2019.

Zhi Li, Wei Shi, and Ming Yan. A decentralized proximal-gradient method with network independent step-sizes and separated convergence rates. *IEEE Transactions on Signal Processing*, 67 (17):4494–4506, 2019.

Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu. Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent. In *Advances in Neural Information Processing Systems*, pp. 5330–5340, 2017.

Qing Ling, Wei Shi, Gang Wu, and Alejandro Ribeiro. DLM: Decentralized linearized alternating direction method of multipliers. *IEEE Transactions on Signal Processing*, 63(15):4051–4064, 2015.

Xiaorui Liu, Yao Li, Jiliang Tang, and Ming Yan. A double residual compression algorithm for efficient distributed learning. *The 23rd International Conference on Artificial Intelligence and Statistics*, 2020.

Yucheng Lu and Christopher De Sa. Moniqua: Modulo quantized communication in decentralized SGD. In *Proceedings of the 37th International Conference on Machine Learning*, 2020.

Konstantin Mishchenko, Eduard Gorbunov, Martin Takáč, and Peter Richtárik. Distributed learning with compressed gradient differences. *arXiv preprint arXiv:1901.09269*, 2019.

Joao FC Mota, Joao MF Xavier, Pedro MQ Aguiar, and Markus Püschel. D-ADMM: A communication-efficient distributed algorithm for separable optimization. *IEEE Transactions on Signal Processing*, 61(10):2718–2723, 2013.

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

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

Yurii Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2013.

Shi Pu and Angelia Nedić. Distributed stochastic gradient tracking methods. *Mathematical Programming*, pp. 1–49, 2020.

Amirhossein Reisizadeh, Aryan Mokhtari, Hamed Hassani, and Ramtin Pedarsani. An exact quantized decentralized gradient descent algorithm. *IEEE Transactions on Signal Processing*, 67(19): 4934–4947, 2019a.

Amirhossein Reisizadeh, Hossein Taheri, Aryan Mokhtari, Hamed Hassani, and Ramtin Pedarsani. Robust and communication-efficient collaborative learning. In *Advances in Neural Information Processing Systems*, pp. 8388–8399, 2019b.

Frank Seide, Hao Fu, Jasha Droppo, Gang Li, and Dong Yu. 1-bit stochastic gradient descent and application to data-parallel distributed training of speech DNNs. In *Interspeech 2014*, September 2014.

Wei Shi, Qing Ling, Gang Wu, and Wotao Yin. EXTRA: An exact first-order algorithm for decentralized consensus optimization. *SIAM Journal on Optimization*, 25(2):944–966, 2015.

Sebastian U. Stich, Jean-Baptiste Cordonnier, and Martin Jaggi. Sparsified SGD with memory. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 4452–4463, 2018.

Hanlin Tang, Shaoduo Gan, Ce Zhang, Tong Zhang, and Ji Liu. Communication compression for decentralized training. In *Advances in Neural Information Processing Systems*, pp. 7652–7662. 2018a.

Hanlin Tang, Xiangru Lian, Ming Yan, Ce Zhang, and Ji Liu. D^2: Decentralized training over decentralized data. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 4848–4856, 2018b.

Hanlin Tang, Xiangru Lian, Shuang Qiu, Lei Yuan, Ce Zhang, Tong Zhang, and Ji Liu. Deepsqueeze: Decentralization meets error-compensated compression. *CoRR*, abs/1907.07346, 2019a. URL http://arxiv.org/abs/1907.07346.

Hanlin Tang, Chen Yu, Xiangru Lian, Tong Zhang, and Ji Liu. DoubleSqueeze: Parallel stochastic gradient descent with double-pass error-compensated compression. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 6155–6165, 2019b.

John Tsitsiklis, Dimitri Bertsekas, and Michael Athans. Distributed asynchronous deterministic and stochastic gradient optimization algorithms. *IEEE transactions on automatic control*, 31(9): 803–812, 1986.

Jiaxiang Wu, Weidong Huang, Junzhou Huang, and Tong Zhang. Error compensated quantized SGD and its applications to large-scale distributed optimization. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 5325–5333, 2018.

Lin Xiao and Stephen Boyd. Fast linear iterations for distributed averaging. *Systems & Control Letters*, 53(1):65–78, 2004.

Jinming Xu, Ye Tian, Ying Sun, and Gesualdo Scutari. Accelerated primal-dual algorithms for distributed smooth convex optimization over networks. In *International Conference on Artificial Intelligence and Statistics*, pp. 2381–2391. PMLR, 2020.

Kun Yuan, Qing Ling, and Wotao Yin. On the convergence of decentralized gradient descent. *SIAM Journal on Optimization*, 26(3):1835–1854, 2016.

Kun Yuan, Bicheng Ying, Xiaochuan Zhao, and Ali H Sayed. Exact diffusion for distributed optimization and learning—part i: Algorithm development. *IEEE Transactions on Signal Processing*, 67(3):708–723, 2018.

Kun Yuan, Wei Xu, and Qing Ling. Can primal methods outperform primal-dual methods in decentralized dynamic optimization? *arXiv preprint arXiv:2003.00816*, 2020.