Achieving near-optimal oracle complexity in decentralized stochastic optimization with channel noise

Soham Mukherjee[1] and Mrityunjoy Chakraborty[2], *Senior Member, IEEE*

***Abstract*— We study a decentralized non-convex stochastic optimization problem in which a group of agents/nodes seek to minimize a global objective function that can be expressed as a sum of local component functions, such that each node in the network has access to exactly one component function. The nodes collaborate with their neighbors by sharing their estimates over communication links that are assumed to be corrupted by additive noise. To address this problem, we propose a computationally efficient and robust algorithm which builds on a probabilistic technique for stochastic gradient computation, which we believe will be applicable to a wide range of problems in decentralized information processing, learning and control. Specifically, we show that the proposed method achieves an oracle complexity (computational complexity) of $\mathsf{O}(1/\epsilon^2)$ for smooth and non-convex functions with stochastic gradients, which is known to be sharp for its respective function class, and is an improvement over the computational cost obtained in previous works. Additionally we retain the $\mathsf{O}(1/\epsilon^3)$ rate for the communication cost, which is at par with the communication cost obtained in previous works. We also show how the proposed algorithm has robust performance in environments with unreliable computational resources. Finally, the theoretical findings are validated via numerical experiments.**

***Index Terms*— Decentralized Optimization, Stochastic Gradients, Non-Convex Optimization, Channel noise.**

I. Introduction

We consider a distributed stochastic optimization problem in which a collection of m nodes connected over a network try to minimize $f(x) = (1/m)\sum_{i=1}^{m} f_i(x)$, such that each component function $f_i(\cdot)$ is known exclusively to node i. In particular, we assume that only node i can access stochastic gradients associated with the function $f_i(\cdot)$. The goal is to develop an algorithm in which the nodes cooperate with each other by performing local computations and exchanging relevant information to estimate the global minimizer $x^* = \arg\min_x f(x)$ or compute a first-order stationary point depending on the function class. Such a framework captures a wide variety of practical problems that frequently arise in distributed control [1], distributed machine learning [2], big data analytics [3], IoT applications [4] and power networks [5] amongst others. Several distributed algorithms have been proposed in recent past to solve this problem, by building on centralized schemes such as gradient descent based algorithms ([6]- [8]), Alternating Direction Method of Multipliers ([9], [10]), Newton methods ([11], [12]) and primal-dual methods ([13], [14]) to name a few. In this paper, we focus our attention on gradient descent based algorithms owing to their simplicity, low overhead and widespread use. However, even among first order gradient descent based algorithms, the vast majority assume prefect communication between nodes without any channel noise ([15]- [18]), noise caused by quantization of transmitted information ([19]- [22]) or noise introduced for privacy preservation ([23]- [25]) - an ideal assumption hardly satisfied in real-world settings. Among the works that took into account noise effects, notable are [15] where the authors analyze the effects of channel noise (both, under the zero-mean bounded variance noise assumption as well as under a Markovian noise sequence assumption) in the average consensus problem, [16]- [18] which consider communication noise in the decentralized optimization setting where the communication noise is modelled as zero-mean bounded variance additive noise and [19]- [22] which consider decentralized optimization problems with quantized communication between nodes where again the quantization noise is modeled as a separate additive noise component having zero-mean and bounded variance. All of these works [16]- [22] consider the *Noisy Consensus + (Stochastic) Gradient Descent* framework for developing their algorithms, with the works in [16] and [18] incorporating an additional gradient tracking step to enable individual nodes in the network to estimate the global gradient. The treatments in [16], [17], [19], [20] and [22], however, restrict them to a setting where exact gradients are available. This requirement is lifted in [18] and [21] which use a stochastic setting. In [18], the authors derive conditions needed to ensure *almost sure* convergence under the zero-mean bounded variance communication noise assumption, with their results for *almost sure* convergence also holding under a more relaxed assumption that allows the noise variance to grow with increasing number of iterations under certain conditions. However, explicit characterization of the computational and communication costs are not provided in [18]. In [21], the authors propose a *QuanTimed-DSGD* algorithm which considers zero-mean, bounded variance communication noise and deploys a deadline based stochastic

1 and 2 are with the Department of Electronics and Electrical Communication Engineering, Indian Institute of Technology, Kharagpur, INDIA (e.mail : [1]sohammukherjee5898@gmail.com, [2]mrityun@ece.iitkgp.ernet.in). This work was funded in part by a grant from the SERB, Govt. of India.

gradient computation framework to mitigate the effects of stragglers. It is shown that this algorithm requires $O(1/\epsilon^{2+\delta})$ (where $\delta > 0$ can be made arbitrarily small) stochastic gradient computations and communication rounds to compute an ϵ-accurate solution for smooth and strongly-convex functions ($O(1/\epsilon^3)$ in the case of smooth, non-convex functions [1]). The computational cost of $O(1/\epsilon^{2+\delta})$ ($O(1/\epsilon^3)$) incurred by the *QuanTimed-DSGD* algorithm, however, makes it suboptimal vis-a-vis the centralized Stochastic Gradient Descent (SGD) algorithm which requires only $O(1/\epsilon)$ ($O(1/\epsilon^2)$) stochastic gradient computations to compute an ϵ-accurate solution. In fact in all of the works [16]- [17], [19]- [22] a common observation is that the *introduction of zero-mean bounded variance communication noise in the problem causes the computational complexity to increase as compared to their centralized counterparts* (i.e., the $m = 1$ case for which there is no communication noise), as displayed in Table 1. This increase in computational cost is not observed in the noiseless perfect communication setting, but appears to be a bottleneck when communication noise is introduced in the system. The study in this paper is primarily motivated by this gap in the computational complexity, especially since the volume of data to be processed has increased multifold over the last few years. Moreover, recent studies have shown that complex models such as over-parametrized neural networks perform well both in theory and in practice ([26], [27]). This combination of data deluge and increasing model complexity has pushed up the demand on computational resources enormously. This also poses a significant limitation in resource-constrained or unreliable environments where there might be occasional failures in carrying out such computationally intensive operations, thus necessitating the development of algorithms which have robust performance in such challenging settings.

In this work, we propose a computationally efficient and robust decentralized algorithm which operates with noisy communication links and incorporates a probabilistic technique for stochastic gradient computation in the widely used *Noisy Consensus + Stochastic Gradient Descent* framework. Specifically, it allows individual nodes to randomly skip the stochastic gradient computation step in a given iteration with some probability. Our main contributions are as follows. **(i).** We show that the computational complexity of the proposed method achieves at par dependence on the required accuracy ϵ with its centralized counterpart (the centralized Stochastic Gradient Descent Algorithm with a first-order oracle [28]). This is a marked improvement over previous works, where it has been seen that the introduction of noise in the information exchange process causes the computational complexity to increase as compared to the computational complexity that can be achieved using centralized versions of the algorithm (please refer to Table 1). Specifically, for smooth and non-convex functions, the proposed method requires $O(1/m\epsilon^3)$ communication rounds and $O(1/m\epsilon^2)$ stochastic gradient computations. The $1/\epsilon^2$ dependence of stochastic gradient computations in the proposed method improves the $1/\epsilon^3$ dependence obtained in [21], resulting in significant computational savings while maintaining at par dependence on ϵ for the number of iterations and the number of communication rounds as in [21]. This decrease in the number of stochastic gradient computations is achieved by carefully choosing the probability with which the gradient computations step is skipped in a given iteration. **(ii).** Additionally, the $O(1/\epsilon^2)$ dependence on required accuracy ϵ for the number of stochastic gradient computations is sharp, in the sense that it matches the lower bound for the number of stochastic gradient computations required to compute an ϵ-accurate solution [29], though in the "expected" sense since the actual number of stochastic gradient computations in our proposed method is a random variable. We, however, show, by developing high-probability bounds that the actual number of stochastic gradient computations is concentrated around its expected value. **(iii).** We also show how, in the noisy communication setting, the proposed algorithm is robust to environments which have unreliable computational resources (e.g., frequent power outages, straggler nodes), where successful stochastic gradient computation in an iteration occurs with some probability less than 1. Specifically, we show that, in the presence of zero-mean and bounded variance communication noise, the overall convergence of the algorithm remains unaffected as long as the probability of successful stochastic gradient computation is above a small threshold, whose value we explicitly characterize in subsequent sections. **(iv).** Lastly, the presence of the $1/m$ factor in the computational and communication costs indicates that the proposed method achieves a linear speed-up proportional to the number of nodes in the network m, which is an improvement over the result obtained in [21], where this effect is not fully realized.

Notation. We use $\boldsymbol{I}_p$ to denote the $p \times p$ identity matrix, $\mathbf{1}_p$ to denote the p-dimensional column vector of all 1s, $\mathbf{0}_p$ to denote the p-dimensional column vector of all 0s and $||\cdot||$ to denote the 2-norm for vectors. For matrices, $||\cdot||$ denotes the spectral norm and $||\cdot||_F$ denotes the Frobenius norm. The expression $Pr\{\mathcal{E}\}$ is used to denote the probability of occurrence of the event $\mathcal{E}$. The expression $Bernoulli(p)$ is used to denote the Bernoulli distribution which, when sampled, outputs 1 with probability p and 0 with probability $1-p$, with $0 \leq p \leq 1$. Lastly, we use $a \wedge b$ to denote $\max\{a, b\}$.

II. Problem Formulation

A. The Decentralized Stochastic Optimization Problem

We consider a setting where the network of m nodes wish to solve the following unconstrained optimization problem.

$$\text{P1: } \underset{\boldsymbol{x}\in R^d}{\text{minimize}}\ f(\boldsymbol{x}) = \frac{1}{m}\sum_{i=1}^{m} f_i(\boldsymbol{x}). \tag{1}$$

Here, the function $f_i(\cdot)$ is known only to node i. Specifically, we assume that only node i can compute stochastic gradients of the function $f_i(\cdot)$. When node i queries a point $\boldsymbol{x}_i(t) \in R^d$, at some time $t \geq 0$, the corresponding Stochastic First-Order Oracle (SFO) returns a random vector $\boldsymbol{g}_i\big(\boldsymbol{x}_i(t)\big) \in R^d$ satisfying

$$E\big[\boldsymbol{g}_i\big(\boldsymbol{x}_i(t)\big)\big|\boldsymbol{x}_i(t)\big] = \nabla f_i\big(\boldsymbol{x}_i(t)\big), \tag{2a}$$

$$E\big[||\boldsymbol{g}_i\big(\boldsymbol{x}_i(t)\big) - \nabla f_i\big(\boldsymbol{x}_i(t)\big)||^2\big|\boldsymbol{x}_i(t)\big] \leq \sigma_g^2, \tag{2b}$$

[1]Refer to Section III.B for the definition of an ϵ-accurate solution for smooth decentralized non-convex optimization problems.

Reference	Problem Type	Gradient Type	Gradient Computations	Communication Rounds	Gradient Computations (centralized counterpart)
[19]	Non-smooth, strongly-convex	Exact	$O(1/\epsilon^3)$	$O(1/\epsilon^3)$	$O(1/\epsilon)$, [30]
[19]	Non-smooth, convex	Exact	$O(1/\epsilon^4)$	$O(1/\epsilon^4)$	$O(1/\epsilon^2)$, [30]
[20]	Smooth, strongly-convex	Exact	$O(1/\epsilon^{2+\delta_1})$ [2]	$O(1/\epsilon^{2+\delta_1})$ [2]	$O(log(1/\epsilon))$, [30]
[21]	Smooth, strongly-convex	Stochastic	$O(1/\epsilon^{2+\delta_2})$ [3]	$O(1/\epsilon^{2+\delta_2})$ [3]	$O(1/\epsilon)$, [30]
[21]	Smooth, non-convex	Stochastic	$O(1/\epsilon^3)$	$O(1/\epsilon^3)$	$O(1/\epsilon^2)$, [28]
[22]	Smooth, strongly-convex	Exact	$O((log(1/\epsilon))^4/\epsilon^2)$	$O((log(1/\epsilon))^4/\epsilon^2)$	$O(log(1/\epsilon))$, [30]
[16]	Smooth, strongly-convex	Exact	Converges to a fixed neighborhood around the optimum. Arbitrarily small ϵ-accuracy cannot be achieved.		$O(log(1/\epsilon))$, [30]
[17]	Smooth, strongly-convex	Exact	$O(1/\epsilon^2)$	$O(1/\epsilon^2)$	$O(log(1/\epsilon))$, [30]
[18] [4]	Smooth, non-convex	Exact	Not explicit	Not explicit	$O(1/\epsilon)$, [31]
[18] [4]	Smooth, non-convex	Stochastic	Not explicit	Not explicit	$O(1/\epsilon^2)$, [28]
This work	Smooth, non-convex	Stochastic	$O(1/\epsilon^2)$	$O(1/\epsilon^3)$	$O(1/\epsilon^2)$, [28]

TABLE I: Comparison with existing methods which solve decentralized optimization problems in the presence of additive zero-mean and bounded variance communication noise.

where σ_g is a positive constant. Equation (2a) implies that the stochastic approximation $\boldsymbol{g}_i(\boldsymbol{x}_i)$ is an unbiased estimate of the true gradient $\nabla f_i(\boldsymbol{x}_i)$ and equation (2b) bounds the variance of the stochastic gradient. These conditions are very common in stochastic optimization literature (see for eg. [18], [21], [28], [32], [33]).

B. Network Model

We consider a network of m nodes, which communicate over a static undirected graph $G = (V, E)$ with $V = \{1, \ldots, m\}$ representing the individual nodes and E representing the connections between the nodes. Two nodes i, j are said to be connected if and only if $(i, j) \in E$. We further associate a $m \times m$ weight matrix $\boldsymbol{W}$ with the graph G, with W_{ij} denoting the weight the node i applies to the signal received from node j. For each $i \in V$, we use $\mathcal{N}_i$ to denote the neighbours of node i, or more formally $\mathcal{N}_i = \{j : j \neq i, W_{ij} > 0\}$. For all $i \in V$, we assume that node i can exchange information with some $j \in V$ if and only if $j \in \mathcal{N}_i$. Our assumptions about the graph G and the network matrix $\boldsymbol{W}$ are stated below.

Assumption 1. The graph G is strongly connected. The weight matrix $\boldsymbol{W}$ associated with the undirected graph G is Hermitian, doubly stochastic and satisfies $W_{ii} > 0 \ \forall \ i \in V$, $W_{ij} \geq 0 \ \forall \ i, j \in V$ and $W_{ij} > 0$ if and only if $(i, j) \in E$.

These assumptions imply that the spectral norm of the matrix $\boldsymbol{W} - (1/m)\mathbf{1}_m\mathbf{1}_m^T$, which we denote by ω, is strictly smaller than 1. This means that for any $\boldsymbol{x} \in R^m$, we have the following contraction $||\boldsymbol{W}\boldsymbol{x} - \mathbf{1}_m\overline{\boldsymbol{x}}|| \leq \omega||\boldsymbol{x} - \mathbf{1}_m\overline{x}||$, where $\overline{x} = (1/m)\mathbf{1}_m^T\boldsymbol{x}$.

C. Communication Noise Model

Consider two neighboring nodes i, j, where node i sends to node j a quantity $\boldsymbol{x}_i(t) \in R^d$, at a time $t \geq 0$, which, in an imperfect communication setting, is corrupted by additive noise and is received as $\widetilde{\boldsymbol{x}}_{ij}(t)$, given by,

$$\widetilde{\boldsymbol{x}}_{ij}(t) = \boldsymbol{x}_i(t) + \boldsymbol{n}_{ij}(t), \tag{3}$$

where $\boldsymbol{n}_{ij}(t) \in R^d$ represents the noise introduced in the link from node i to node j at time t. We assume that for all $i \in V$,

Quantity	Description
K	Number of iterations
$\chi_i(t)$	Bernoulli random variable drawn by node i at iteration t
θ	Probability with which $\chi_i(t)$ takes the value 1
$\boldsymbol{x}_i(t)$	Node i's estimate at iteration t
$\widetilde{\boldsymbol{x}}_{ji}(t)$	Node j's noise-corrupted estimate received by node i at iteration t
$\boldsymbol{h}_i(t)$	Quantity used by node i as a stochastic approximation of its local gradient $\nabla f_i(\boldsymbol{x}_i(t))$ at iteration t
$\boldsymbol{g}_i(\boldsymbol{x}_i(t))$	Stochastic gradient computed by node i at iteration t if and only if $\chi_i(t) = 1$
η	Step-size parameter associated with the gradient update term
c	Step-size parameter associated with the consensus update term

TABLE II: Summary of variables in Algorithm 1.

the following conditions hold

$$E\big[\boldsymbol{n}_{ij}(t)\big|x_i(t)\big] = 0, \tag{4a}$$

$$E\big[\big|\big|\boldsymbol{n}_{ij}(t)\big|\big|^2\big|\boldsymbol{x}_i(t)\big] \leq \sigma_c^2, \tag{4b}$$

where σ_c is a positive constant. We also assume that for any $t \geq 0$, the set of random variables $\{\boldsymbol{n}_{ij}(t)\}$, $(i, j) \in E$, are pairwise independent conditioned on $\boldsymbol{x}_1(t), \cdots, \boldsymbol{x}_m(t)$.

III. Proposed Method and Convergence Result

A. Algorithm Description

Let $\boldsymbol{x}_i(t)$ denote node i's estimate at iteration t. Also, let $\chi_i(t) \sim Bernoulli(\theta)$ $(0 < \theta < 1)$ be a random variable associated with node i at time t which is independent of all other sources of randomness arising due to stochastic gradients and channel noise. Using the above, we summarize the proposed method in *Algorithm 1*.

In (5a)-(5b), the algorithm calculates $\boldsymbol{h}_i(t)$ and uses it as an estimate for the local gradient in (5c), which is a standard *Noisy Consensus + Stochastic Gradient Descent (SGD)* update

[2] $\delta_1 > 0$ can be made arbitrarily small.
[3] $\delta_2 > 0$ can be made arbitrarily small.
[4] Analysis is focused on establishing *almost sure* convergence and does not provide finite-time results.

Algorithm 1

Input: K, θ, η, c.

Initialization: For each $i \in V$, initialize $\boldsymbol{x}_i(0) = \mathbf{0}_d$.

for $t = 0, 1, \cdots, K-1$ **do**

for each $i \in V$ **do**

1. Draw $\chi_i(t) \sim Bernoulli(\theta)$.

2. **if** $(\chi_i(t) == 1)$ **then**

Compute $\boldsymbol{g}_i\big(\boldsymbol{x}_i(t)\big)$ and

set $\boldsymbol{h}_i(t) = (1/\theta)\boldsymbol{g}_i\big(\boldsymbol{x}_i(t)\big)$. (5a)

else

Set $\boldsymbol{h}_i(t) = \mathbf{0}_d$. (5b)

end if

3. Node i sends $x_i(t)$ to all $j \in \mathcal{N}_i$ and receives $\widetilde{x}_{ji}(t)$ from all $j \in \mathcal{N}_i$.

4. Update $\boldsymbol{x}_i(t+1) = \underbrace{\boldsymbol{x}_i(t) - \eta \boldsymbol{h}_i(t)}_{\texttt{SGD}} + \underbrace{c \sum_{j \in \mathcal{N}_i} W_{ij}\big(\widetilde{\boldsymbol{x}}_{ji}(t) - \boldsymbol{x}_i(t)\big)}_{\texttt{Noisy Consensus}}$. (5c)

end for

end for

step ([17], [19]- [22]). Please refer to *Table II* for a list of variables used in *Algorithm 1*.

Focusing on (5a)-(5b), we see that $\boldsymbol{g}_i\big(\boldsymbol{x}_i(t)\big)$ is itself a stochastic approximation of the true gradient $\nabla f_i(\boldsymbol{x}_i(t))$. The gradient estimator $\boldsymbol{h}_i(t)$ adds another layer of randomness by randomly choosing whether or not to compute a stochastic gradient at time t. Note that (5a)-(5b) can be written compactly as $\boldsymbol{h}_i(t) = \frac{\chi_i(t)}{\theta}\boldsymbol{g}_i\big(\boldsymbol{x}_i(t)\big)$. Then we have the following result on $h_i(t)$, which follows trivially from (2a), (2b) and statistical independence of $\chi_i(t)$ from all other sources of randomness at node i, with $E[\chi_i(t)] = \theta,\ E[\chi_i^2(t)] = \theta$.

Lemma 1. The following holds for all $t \geq 0$

$$E\big[\boldsymbol{h}_i(t)\big|\boldsymbol{x}_i(t)\big] = \nabla f_i(\boldsymbol{x}_i(t)), \tag{6a}$$

$$E\big[\big|\big|\boldsymbol{h}_i(t) - \nabla f_i(\boldsymbol{x}_i(t))\big|\big|^2\big|\boldsymbol{x}_i(t)\big] \leq \frac{1}{\theta}\sigma_g^2 + \left(\frac{1}{\theta} - 1\right)||\nabla f_i(\boldsymbol{x}_i(t))||^2. \tag{6b}$$

Eq. (6a) shows that $\boldsymbol{h}_i(t)$ is an unbiased estimator of the exact gradient $\nabla f_i(\boldsymbol{x}_i(t))$ conditioned on $\boldsymbol{x}_i(t)$. However, comparing (6b) and (2b), we observe that $\boldsymbol{h}_i(t)$ has higher variance than $\boldsymbol{g}_i\big(\boldsymbol{x}_i(t)\big)$ (The constant term σ_g^2 is scaled up by a factor of $1/\theta$ and there is an extra $(1/\theta - 1)||\nabla f_i(\boldsymbol{x}_i(t))||^2$ term.). However notice that at time t, the stochastic gradient is computed with probability θ (where, as we shall see in the next section, θ will be chosen to be of the form $\theta = \Theta(1/(m^\mu K^\tau))$, with $\mu > 0$ and $\tau > 0$). This means that the stochastic gradient computation step is skipped in many iterations and $\boldsymbol{h}_i(t)$ is set to a zero vector in such iterations. Thus there are two opposing forces in play: higher variance associated with the gradient estimate $\boldsymbol{h}_i(t)$ and lower expected computational cost per iteration (where the expectation is taken on $\chi_i(t)$) by virtue of skipping the stochastic gradient computation step in some iterations. This raises the question of whether such an operation reduces the overall computational complexity, by requiring fewer stochastic gradient computations totalled over all iterations, while at the same time ensuring that the increased variance of the gradient estimate $\boldsymbol{h}_i(t)$ does not have any detrimental effect on the overall convergence of the algorithm. We answer this question positively in the next section.

B. Convergence result for the general non-convex case

In this section, we make the following assumptions about the local component functions and the global objective function.

Assumption 2. The local component functions f_i's are L-smooth, i.e., they have L-Lipschitz continuous gradients. This means, for all $i \in V$ and for all $\boldsymbol{x}, \boldsymbol{y} \in R^d$, we have

$$||\nabla f_i(\boldsymbol{x}) - \nabla f_i(\boldsymbol{y})|| \leq L||\boldsymbol{x} - \boldsymbol{y}||. \tag{7}$$

Note that this assumption implies that

$$f_i(\boldsymbol{x}) \leq f_i(\boldsymbol{y}) + \langle \nabla f_i(\boldsymbol{y}), \boldsymbol{x} - \boldsymbol{y} \rangle + \frac{L}{2}||\boldsymbol{x} - \boldsymbol{y}||^2. \tag{8}$$

Assumption 3. The global objective function $f(\cdot)$ is lower bounded by f^*, i.e., for all $\boldsymbol{x} \in R^d$, we have

$$f(\boldsymbol{x}) \geq f^*. \tag{9}$$

We introduce a quantity called the network-averages gradient disagreement, which is defined for all $\boldsymbol{x} \in R^d$ as

$$\mathcal{NAGD}(\boldsymbol{x}) \triangleq \frac{1}{m}\sum_{i=1}^{m}||\nabla f_i(\boldsymbol{x}) - \nabla f(\boldsymbol{x})||^2. \tag{10}$$

The function $\mathcal{NAGD}(\boldsymbol{x})$ captures the heterogeneity of the data available to the different nodes in the network at $\boldsymbol{x}$. We make the following assumption on $\mathcal{NAGD}(\cdot)$:

Assumption 4. For all $\boldsymbol{x} \in R^d$, the network-averaged gradient disagreement satisfies

$$\mathcal{NAGD}(\boldsymbol{x}) = \frac{1}{m}\sum_{i=1}^{m}||\nabla f_i(\boldsymbol{x}) - \nabla f(\boldsymbol{x})||^2 \leq \sigma_G^2, \tag{11}$$

for some $\sigma_G^2 < \infty$. These assumptions are very common in decentralized stochastic non-convex optimization (see for e.g. [21], [32]).

Definition 1. Consider problem P1. Under Assumptions 1-4, an algorithm is said to compute an ϵ-accurate solution at iteration K if the following conditions are satisfied :

$$\frac{1}{K}\sum_{t=0}^{K-1}E\big[||\nabla f\big(\overline{\boldsymbol{x}}(t)\big)||^2\big] \leq \epsilon, \tag{12a}$$

$$\frac{1}{mK}\sum_{t=0}^{K-1}\sum_{i=1}^{m}E\big[||\boldsymbol{x}_i(t) - \overline{\boldsymbol{x}}(t)||^2\big] \leq \epsilon, \tag{12b}$$

where $\overline{\boldsymbol{x}}(t) = \frac{1}{m}\sum_{i=1}^{n}\boldsymbol{x}_i(t)$. This is a commonly used metric while characterizing the performance of decentralized non-convex optimization algorithms (see, for example, the works, [21], [32]). We then have the following result characterizing the performance of the proposed method in the general non-convex setting.

Theorem 1. *Let Assumptions 1-4 hold and α, β and γ be constants. Then Algorithm 1 with the parameter choices*

$$\eta = \frac{m^{\alpha}}{K^{2/3}}, \quad c = \frac{m^{\beta}}{K^{1/2}}, \quad \theta = \frac{m^{\gamma}}{K^{1/3}}, \tag{13}$$

satisfies the following bounds for $K \geq K_0$.

$$\begin{aligned}\frac{1}{K}\sum_{t=0}^{K-1} E\big[||\nabla f(\overline{\boldsymbol{x}}(t))||^2\big] &\leq O\bigg(\frac{m^{-\alpha}}{K^{1/3}}\bigg)\big[f(\overline{\boldsymbol{x}}(0)) - f^*\big] \\ &+ O\bigg(\frac{m^{(2\beta-1-\alpha)}}{K^{1/3}} + \frac{m^{\beta}}{K^{1/2}}\bigg)\sigma_c^2 \\ &+ O\bigg(\frac{m^{(\alpha-\gamma-1)}}{K^{1/3}} + \frac{m^{(2\alpha-\beta-\gamma)}}{K^{1/2}}\bigg)\sigma_g^2 \\ &+ O\bigg(\frac{m^{(\alpha-\gamma-1)} + m^{2(\alpha-\beta)}}{K^{1/3}} + \frac{m^{(2\alpha-\beta-\gamma)}}{K^{1/2}} + \frac{m^{\alpha}}{K^{2/3}}\bigg)\sigma_G^2,\end{aligned} \tag{14}$$

$$\begin{aligned}&\frac{1}{mK}\sum_{t=0}^{K-1}\sum_{i=1}^{m} E\big[||\boldsymbol{x}_i(t) - \overline{\boldsymbol{x}}(t)||^2\big] \\ &\leq O\bigg(\frac{m^{2(\alpha-\beta)}}{K^{1/3}} + \frac{m^{(2\alpha-\beta-\gamma)}}{K^{1/2}}\bigg)\frac{1}{K}\sum_{t=0}^{K-1} E\big[||\nabla f(\overline{x}(t))||^2\big] \\ &+ O\bigg(\frac{m^{2(\alpha-\beta)}}{K^{1/3}} + \frac{m^{(2\alpha-\beta-\gamma)}}{K^{1/2}}\bigg)\sigma_G^2 + O\bigg(\frac{m^{(2\alpha-\beta-\gamma)}}{K^{1/2}}\bigg)\sigma_g^2 \\ &+ O\bigg(\frac{m^{\beta}}{K^{1/2}}\bigg)\sigma_c^2,\end{aligned} \tag{15}$$

where K_0 is given by

$$K_0 = max\{1, K_1, K_2\}, \tag{16}$$

with

$$K_1 = \bigg(\frac{36m^{(2\alpha-2\beta)}L^2}{(1-\omega)^2} + \frac{192m^{(2\alpha-2\beta-\gamma)}L^2}{(1-\omega)}\bigg)^3, \tag{17}$$

$$K_2 = (8m^{\alpha-\gamma-1}L + 8m^{\alpha}L)^3. \tag{18}$$

Moreover, for any node $i \in \{1, \cdots, m\}$, the expected number of stochastic gradient computations up to time K is given by

$$\sum_{s=0}^{K-1} E\big[\chi_i(s)\big] = m^{\gamma}K^{2/3}, \tag{19}$$

with the actual number of stochastic gradient computations satisfying the following high-probability bound

$$Pr\bigg\{\sum_{s=0}^{K-1}\chi_i(s) \leq 2m^{\gamma}K^{2/3}\bigg\} \geq 1 - exp\big(-2m^{2\gamma}K^{1/3}\big). \tag{20}$$

Proof : Given in Section IV ("Convergence Analysis").

It is to be noted that the parameters α, β and γ in Theorem 1 are left as free parameters and that the slowest decaying term is $O(1/K^{1/3})$. We choose α, β and γ to obtain the best possible dependence on m in the $O(1/K^{1/3})$ term in the following result.

Corollary 1. *Choosing $\alpha = 1/3$, $\beta = 1/2$ and $\gamma = -1/3$ in Theorem 1 gives us the following bound for $K \geq K_0$.*

$$\frac{1}{K}\sum_{t=0}^{K-1} E\big[||\nabla f(\overline{\boldsymbol{x}}(t))||^2\big] \leq O\bigg(\frac{1}{m^{1/3}K^{1/3}} + \frac{m^{1/2}}{K^{1/2}}\bigg), \tag{21}$$

$$\frac{1}{mK}\sum_{t=0}^{K-1}\sum_{i=1}^{m} E\big[||\boldsymbol{x}_i(t) - \overline{\boldsymbol{x}}(t)||^2\big] \leq O\bigg(\frac{1}{m^{1/3}K^{1/3}} + \frac{m^{1/2}}{K^{1/2}}\bigg). \tag{22}$$

If we let K_ϵ denote the number of iterations required for Algorithm 1 to obtain an ϵ-accurate solution according to Definition 1, then under the conditions in Corollary 1, we have

$$K_\epsilon \leq O\bigg(\frac{1}{m\epsilon^3} \wedge \frac{m}{\epsilon^2}\bigg). \tag{23}$$

Consequently, the expected number of stochastic gradient computations at any given node $i \in V$ is given by

$$\sum_{s=0}^{K-1} E\big[\chi_i(s)\big] = \frac{K_\epsilon^{2/3}}{m^{1/3}} \leq O\bigg(\frac{1}{m\epsilon^2} \wedge \frac{m^{1/3}}{\epsilon^{4/3}}\bigg). \tag{24}$$

Thus, if ϵ is chosen to be sufficiently small while m is moderate (as is typically the case), the communication and computational costs scale as $O\big(1/(m\epsilon^3)\big)$ and $O\big(1/(m\epsilon^2)\big)$ respectively, thus improving over the $O\big(1/(\epsilon^3)\big)$ computational and communication costs obtained in [21].

Focusing specifically on the computational complexity, it is interesting to note that a properly tuned value of θ in *Algorithm 1* has allowed us to obtain a $1/\epsilon^2$ dependence on the required accuracy ϵ for the number of stochastic gradient computations. This matches the $1/\epsilon^2$ dependence obtained in [28] for the number of stochastic gradient computations in centralized first-order Stochastic Gradient Descent with smooth and non-convex functions. Additionally, the $1/\epsilon^2$ dependence is also known to be sharp, in the sense that it matches the $1/\epsilon^2$ dependence obtained in the lower bound established in [29] for the number of stochastic gradient computations in smooth and non-convex optimization. Thus, the randomized gradient computation framework in *Algorithm 1* has allowed us to achieve near-optimal dependence on ϵ in the computational complexity. We say near-optimal because we state our results about the computational complexity in terms of the expected number of stochastic gradient computations which is to be contrasted with the fact that the results in [28] and [29] consider deterministic number of stochastic gradient computations. However, we make up for this by providing a high-probability bound in (20) which shows that the actual number of stochastic gradient computations is close to its expected value in the order sense with high probability.

Moreover, the presence of the $1/m$ factor in both the communication and computational costs indicates that the proposed method is able to achieve a linear speed-up proportional to the number of nodes in the network. This is expected and has been seen in previous works which assume noiseless perfect communication settings (see for example [32]). The result obtained in [21] uses step-sizes which are independent of the network size m and is therefore not able to fully capture this effect, as opposed to the current work and the work in [32], which are able to capture this effect by using step-sizes which are dependent on m.

IV. Convergence Analysis

For our analysis, we use a strategy similar to the ones found in [19] and [21]. At a high level, we try to bound the disagreements between the estimates computed by the various nodes and the network average of these estimates. We then analyze how well the network average approximates a first-order stationary point. These intermediate results are then combined to obtain our final result in Theorem 1.

Consider the update step in (5c). Separating out the channel noise term, we rewrite (5c) as

$$\boldsymbol{x}_i(t+1) = \boldsymbol{x}_i(t) + c\sum_{j\in\mathcal{N}_i} W_{ij}\big(\boldsymbol{x}_j(t) - \boldsymbol{x}_i(t)\big) - \eta \boldsymbol{h}_i(t) + c\underbrace{\sum_{j\in\mathcal{N}_i} W_{ij}\boldsymbol{n}_{ji}(t)}_{\boldsymbol{n}_i(t)}. \tag{25}$$

We now stack the variables associated with various nodes to define the following $d\times m$ matrices.

$$\boldsymbol{X}(t) = \big[\boldsymbol{x}_1(t),\cdots,\boldsymbol{x}_m(t)\big] \in R^{d\times m}, \tag{26a}$$
$$\boldsymbol{H}(t) = \big[\boldsymbol{h}_1(t),\cdots,\boldsymbol{h}_m(t)\big] \in R^{d\times m}, \tag{26b}$$
$$\boldsymbol{N}(t) = \big[\boldsymbol{n}_1(t),\cdots,\boldsymbol{n}_m(t)\big] \in R^{d\times m}. \tag{26c}$$

Then, from the communication noise model described in Section II.C, the following holds for all $i\in V$.

$$E\big[\boldsymbol{n}_i(t)\big|\boldsymbol{X}(t)\big] = 0, \tag{27a}$$
$$E\big[||\boldsymbol{n}_i(t)||^2\big|\boldsymbol{X}(t)\big] \le \sigma_c^2, \tag{27b}$$

with the set of random variables $\{\boldsymbol{n}_i(t)\}$, $i\in V$ being pairwise-independent conditioned on $\boldsymbol{X}(t)$.

Further, let us define

$$F\big(\boldsymbol{X}(t)\big) = \sum_{i=1}^m f_i(\boldsymbol{x}_i(t)). \tag{28}$$

Then, we denote

$$\nabla F\big(\boldsymbol{X}(t)\big) = \big[\nabla f_1\big(\boldsymbol{x}_1(t)\big),\cdots,\nabla f_m\big(\boldsymbol{x}_m(t)\big)\big] \in R^{d\times m}. \tag{29}$$

Additionally, we define a lazy version of W as follows.

$$\boldsymbol{W}_c = (1-c)\boldsymbol{I}_m + c\boldsymbol{W}, \tag{30}$$

It is easy to see that the spectral norm of the matrix $\boldsymbol{W}_c - (1/m)\boldsymbol{1}_m\boldsymbol{1}_m^T$, which we denote by ω_c, satisfies $\omega_c \le (1-c)+c\omega$.

With the above quantities defined, the update step (5c) of *Algorithm 1* can be written compactly as

$$\boldsymbol{X}(t+1) = \boldsymbol{X}(t)\boldsymbol{W}_c^T - \eta\boldsymbol{H}(t) + c\boldsymbol{N}(t), \tag{31}$$

We also define the following network averages.

$$\overline{\nabla F}(\boldsymbol{X}(t)) = \frac{1}{m}\sum_{i=1}^m \nabla f_i(\boldsymbol{x}_i(t)), \quad \overline{\boldsymbol{x}}(t) = \frac{1}{m}\sum_{i=1}^m \boldsymbol{x}_i(t),$$
$$\overline{\boldsymbol{h}}(t) = \frac{1}{m}\sum_{i=1}^m \boldsymbol{h}_i(t), \quad \overline{\boldsymbol{n}}(t) = \frac{1}{m}\sum_{i=1}^m \boldsymbol{n}_i(t).$$

A. Auxiliary Results

Lemma 2 (Evolution of the network average). The following holds at all times $t\ge 0$.

$$\overline{\boldsymbol{x}}(t+1) = \overline{\boldsymbol{x}}(t) - \eta\overline{\boldsymbol{h}}(t) + c\overline{\boldsymbol{n}}(t). \tag{32}$$

Proof: Given in Appendix A ("Proofs of auxiliary results").

Lemma 3 (Disagreement Bound). Under the assumption that $c\le 1$ in (5c), the following holds for all $t\ge 0$.

$$\begin{aligned} &E\big[||\boldsymbol{X}(t+1) - \overline{\boldsymbol{x}}(t+1)\boldsymbol{1}_m^T||_F^2\big] \\ &\le \big(1-(1-\omega)c\big)E\big[||\boldsymbol{X}(t) - \overline{\boldsymbol{x}}(t)\boldsymbol{1}_m^T||_F^2\big] \\ &\quad + \left(\frac{4\eta^2L^2}{(1-\omega)c} + \frac{2\eta^2L^2}{\theta}\right)E\big[||\boldsymbol{X}(t) - \overline{\boldsymbol{x}}(t)\boldsymbol{1}_m^T||_F^2\big] \\ &\quad + \left(\frac{4\eta^2}{(1-\omega)c} + \frac{2\eta^2}{\theta}\right)E[||\nabla F\big(\overline{\boldsymbol{x}}(t)\boldsymbol{1}_m^T\big)||_F^2] \\ &\quad + \frac{\eta^2 m\sigma_g^2}{\theta} + mc^2\sigma_c^2. \end{aligned} \tag{33}$$

Proof: Given in Appendix A ("Proofs of auxiliary results").

Lemma 4 (Effect of the network average). The following holds at all times $t\ge 0$.

$$\begin{aligned} &E\big[f\big(\overline{\boldsymbol{x}}(t+1)\big)\big] \le E\big[f\big(\overline{\boldsymbol{x}}(t)\big)\big] - \frac{\eta}{2}E\big[||\nabla f\big(\overline{\boldsymbol{x}}(t)\big)||^2\big] \\ &+ \left(\frac{\eta L^2 + 2\eta^2L^3}{2m} + \frac{\eta^2L^3}{\theta m^2}\right)E\big[||\boldsymbol{X}(t) - \overline{\boldsymbol{x}}(t)\boldsymbol{1}_m^T||_F^2\big] \\ &+ \left(\frac{\eta^2L}{\theta m^2} + \frac{\eta^2L}{m}\right)E\big[||\nabla F\big(\overline{\boldsymbol{x}}(t)\boldsymbol{1}_m^T\big)||_F^2\big] + \frac{\eta^2L\sigma_g^2}{2m\theta} + \frac{c^2L\sigma_c^2}{2m}. \end{aligned} \tag{34}$$

Proof: Given in Appendix A ("Proofs of auxiliary results").

B. Proof of Theorem 1

Using Assumption 3, we have

$$\begin{aligned} &\big|\big|\nabla F\big(\overline{x}(t)\boldsymbol{1}_m^T\big)\big|\big|_F^2 \\ &\le 2\big|\big|\nabla F\big(\overline{\boldsymbol{x}}(t)\boldsymbol{1}_m^T\big) - \nabla f\big(\overline{\boldsymbol{x}}(t)\big)\boldsymbol{1}_m^T\big|\big|_F^2 + 2\big|\big|\nabla f\big(\overline{\boldsymbol{x}}(t)\big)\boldsymbol{1}_m^T\big|\big|_F^2 \\ &\le 2m\sigma_G^2 + 2\big|\big|\nabla f\big(\overline{\boldsymbol{x}}(t)\big)\boldsymbol{1}_m^T\big|\big|_F^2. \end{aligned} \tag{35}$$

Using this, we can rewrite the result in lemma 3 as

$$\begin{aligned}
&E\big[||\boldsymbol{X}(t+1)-\overline{\boldsymbol{x}}(t+1)\mathbf{1}_m^T||_F^2\big]\\
&\le \big(1-(1-\omega)c\big)E\big[||\boldsymbol{X}(t)-\overline{\boldsymbol{x}}(t)\mathbf{1}_m^T||_F^2\big]\\
&\quad+\left(\frac{4\eta^2L^2}{(1-\omega)c}+\frac{2\eta^2L^2}{\theta}\right)E\big[||\boldsymbol{X}(t)-\overline{\boldsymbol{x}}(t)\mathbf{1}_m^T||_F^2\big]\\
&\quad+\left(\frac{4\eta^2}{(1-\omega)c}+\frac{2\eta^2}{\theta}\right)2mE\big[||\nabla f\big(\overline{\boldsymbol{x}}(t)\big)||^2\big]\\
&\quad+\frac{4m\eta^2(\sigma_G^2+\sigma_g^2)}{\theta}+\frac{8m\eta^2\sigma_G^2}{(1-\omega)c}+mc^2\sigma_c^2\\
&\overset{(a)}{\le}\left(1-\frac{(1-\omega)c}{2}\right)E\big[||\boldsymbol{X}(t)-\overline{\boldsymbol{x}}(t)\mathbf{1}_m^T||_F^2\big]\\
&\quad+\left(\frac{4\eta^2}{(1-\omega)c}+\frac{2\eta^2}{\theta}\right)2mE\big[||\nabla f\big(\overline{\boldsymbol{x}}(t)\big)||^2\big]\\
&\quad+\frac{4m\eta^2(\sigma_G^2+\sigma_g^2)}{\theta}+\frac{8m\eta^2\sigma_G^2}{(1-\omega)c}+mc^2\sigma_c^2\\
&\le\left(1-\frac{(1-\omega)c}{2}\right)^{t+1}\big[||\boldsymbol{X}(0)-\overline{\boldsymbol{x}}(0)\mathbf{1}_m^T||_F^2\big]\\
&\quad+\left(\frac{4\eta^2}{(1-\omega)c}+\frac{2\eta^2}{\theta}\right)2m\\
&\quad\times\sum_{s=0}^{t}\left[\left(1-\frac{(1-\omega)c}{2}\right)^{t-s}E\big[||\nabla f\big(\overline{\boldsymbol{x}}(s)\big)||^2\big]\right]\\
&\quad+\left(\frac{4m\eta^2(\sigma_G^2+\sigma_g^2)}{\theta}+\frac{8m\eta^2\sigma_G^2}{(1-\omega)c}+mc^2\sigma_c^2\right)\\
&\quad\times\sum_{s=0}^{t}\left(1-\frac{(1-\omega)c}{2}\right)^{s},
\end{aligned}\tag{36}$$

where the condition

$$(1-\omega)c\ge\frac{8\eta^2L^2}{(1-\omega)c}+\frac{4\eta^2L^2}{\theta}\tag{37}$$

used in (a) holds for the values of η, c and θ given in the statement of Theorem 1 for all $K\ge K_0$. Using the fact that $\sum_{s=0}^{t}\left(1-\frac{(1-\omega)c}{2}\right)^s<\sum_{s=0}^{\infty}\left(1-\frac{(1-\omega)c}{2}\right)^s=\frac{2}{(1-\omega)c}$, and unrolling the recursion in (36), we get

$$\begin{aligned}
&\sum_{t=0}^{K-1}E\big[||\boldsymbol{X}(t+1)-\overline{\boldsymbol{x}}(t+1)\mathbf{1}_m^T||_F^2\big]\\
&\le\sum_{t=0}^{K-1}\left[\left(1-\frac{(1-\omega)c}{2}\right)^{t+1}\right]\big[||\boldsymbol{X}(0)-\overline{\boldsymbol{x}}(0)\mathbf{1}_m^T||_F^2\big]\\
&\quad+\left(\frac{4\eta^2}{(1-\omega)c}+\frac{2\eta^2}{\theta}\right)2m\\
&\quad\times\sum_{t=0}^{K-1}\left[E\big[||\nabla f\big(\overline{\boldsymbol{x}}(t)\big)||^2\big]\left\{\sum_{s=0}^{K-t-1}\left(1-\frac{(1-\omega)c}{2}\right)^s\right\}\right]\\
&\quad+\frac{8m\eta^2(\sigma_G^2+\sigma_g^2)}{(1-\omega)c\theta}+\frac{16m\eta^2\sigma_G^2}{(1-\omega)^2c^2}+\frac{2mc\sigma_c^2}{(1-\omega)}.
\end{aligned}\tag{38}$$

Adding $||\boldsymbol{X}(0)-\overline{\boldsymbol{x}}(0)\mathbf{1}_m^T||^2$ on both sides, we get

$$\begin{aligned}
&\frac{1}{K}\sum_{t=0}^{K}E\big[||\boldsymbol{X}(t)-\overline{\boldsymbol{x}}(t)\mathbf{1}_m^T||_F^2\big]\\
&\le\frac{2}{(1-\omega)cK}||\boldsymbol{X}(0)-\overline{\boldsymbol{x}}(0)\mathbf{1}_m^T||^2\\
&\quad+\left(\frac{8\eta^2}{(1-\omega)^2c^2}+\frac{4\eta^2}{(1-\omega)c\theta}\right)\frac{2m}{K}\sum_{t=0}^{K-1}E\big[||\nabla f\big(\overline{\boldsymbol{x}}(t)\big)||^2\big]\\
&\quad+\frac{8m\eta^2(\sigma_G^2+\sigma_g^2)}{(1-\omega)c\theta}+\frac{16m\eta^2\sigma_G^2}{(1-\omega)^2c^2}+\frac{2mc\sigma_c^2}{(1-\omega)}.
\end{aligned}\tag{39}$$

From the initialization condition, we have $||\boldsymbol{X}(0)-\overline{\boldsymbol{x}}(0)\mathbf{1}_m^T||_F^2=0$. Plugging in the values of $\eta,\ c$ and θ as given in the statement of Theorem 1 in (39), (15) follows.

Again, we use (35) and the fact that $\eta L<\theta<1$ holds for large K to rewrite the result in lemma 4 as follows.

$$\begin{aligned}
f^*\le E\big[f\big(\overline{\boldsymbol{x}}(t+1)\big)\big]&\le E\big[f\big(\overline{\boldsymbol{x}}(t)\big)\big]-\frac{\eta}{2}E\big[||\nabla f\big(\overline{\boldsymbol{x}}(t)\big)||^2\big]\\
&\quad+\frac{3\eta L^2}{m}E\big[||\boldsymbol{X}(t)-\overline{\boldsymbol{x}}(t)\mathbf{1}_m^T||_F^2\big]\\
&\quad+2\left(\frac{\eta^2L}{\theta m}+\eta^2L\right)E\big[||\nabla f\big(\overline{\boldsymbol{x}}(t)\big)||^2\big]+\frac{c^2L\sigma_c^2}{2m}\\
&\quad+2\eta^2L\sigma_G^2+\frac{\eta^2L(\sigma_g^2+\sigma_G^2)}{m\theta}.
\end{aligned}\tag{40}$$

From the choice of the parameters η, c and θ, it is clear the the following condition holds for all $K\ge K_0$.

$$\frac{8\eta L}{\theta m}+8\eta L\le 1.\tag{41}$$

This then allows us to rewrite (40) as

$$\begin{aligned}
&\frac{1}{K}\sum_{t=0}^{K-1}E\big[||\nabla f\big(\overline{\boldsymbol{x}}(t)\big)||^2\big]\\
&\le\frac{4\big[f\big(\overline{\boldsymbol{x}}(0)\big)-f^*\big]}{\eta K}+\frac{2c^2L\sigma_c^2}{\eta m}+8\eta L\sigma_G^2+\frac{4\eta L(\sigma_g^2+2\sigma_G^2)}{m\theta}\\
&\quad+\frac{12L^2}{m}\frac{1}{K}\sum_{t=0}^{K-1}E\big[||\boldsymbol{X}(t)-\overline{\boldsymbol{x}}(t)\mathbf{1}_m^T||_F^2\big]\\
&\le\frac{4\big[f\big(\overline{\boldsymbol{x}}(0)\big)-f^*\big]}{\eta K}+\frac{2c^2L\sigma_c^2}{\eta m}+8\eta L\sigma_G^2+\frac{4\eta L(\sigma_g^2+2\sigma_G^2)}{m\theta}\\
&\quad+\left(\frac{96\eta^2L^2}{(1-\omega)^2c^2}+\frac{48L^2\eta^2}{(1-\omega)c\theta}\right)\frac{2}{K}\sum_{t=0}^{K-1}E\big[||\nabla f\big(\overline{\boldsymbol{x}}(t)\big)||^2\big]\\
&\quad+\frac{96\eta^2L^2(\sigma_g^2+\sigma_G^2)}{(1-\omega)c\theta}+\frac{192\eta^2L^2\sigma_G^2}{(1-\omega)^2c^2}+\frac{24L^2c\sigma_c^2}{(1-\omega)}\\
&\quad+\frac{12L^2}{(1-\omega)cK}\frac{1}{m}||\boldsymbol{X}(0)-\overline{\boldsymbol{x}}(0)\mathbf{1}_m^T||_F^2.
\end{aligned}\tag{42}$$

Again, the following inequality holds for the choice of the parameters η, c and θ for all $K\ge K_0$.

$$\frac{96\eta^2L^2}{(1-\omega)^2c^2}+\frac{48L^2\eta^2}{(1-\omega)c\theta}\le\frac{1}{4}.\tag{43}$$

Using the above, we can simplify (42) as follows.

$$\frac{1}{K}\sum_{t=0}^{K-1} E\big[||\nabla f\big(\overline{\boldsymbol{x}}(t)\big)||^2\big] \leq \frac{8\big[f\big(\overline{\boldsymbol{x}}(0)\big)-f^*\big]}{\eta K} + \frac{4c^2L\sigma_c^2}{\eta m} + 16\eta L\sigma_G^2 + \frac{8\eta L(\sigma_g^2+\sigma_G^2)}{m\theta} + \frac{192\eta^2L^2(\sigma_g^2+\sigma_G^2)}{(1-\omega)c\theta} + \frac{384\eta^2L^2\sigma_G^2}{(1-\omega)^2c^2} + \frac{48L^2c\sigma_c^2}{(1-\omega)} + \frac{24L^2}{(1-\omega)cK}\frac{1}{m}||\boldsymbol{X}(0)-\overline{\boldsymbol{x}}(0)\mathbf{1}_m^T||_F^2. \quad (44)$$

Recalling that $||\boldsymbol{X}(0)-\overline{\boldsymbol{x}}(0)\mathbf{1}_m^T||_F^2 = 0$ holds from the initialization condition and plugging in the values of η, c and θ from the statement of Theorem 1, one obtains (14).

Finally, we need to prove that the actual number of stochastic gradient computations is concentrated around its expected value. To show this, we note that for any given $i \in \{1,\cdots,m\}$, the random variables $\{\chi_i(s)\}$, $s=0,\cdots,K-1$, are independent Bernoulli random variables, bounded between 0 and 1. Then, applying Hoeffding's inequality for some $\phi>0$, we get

$$Pr\Big\{\sum_{s=0}^{K-1}\chi_i(s) \geq \sum_{s=0}^{K-1} E\big[\chi_i(s)\big]+\phi K\Big\} \leq exp(-2K\phi^2). \quad (45)$$

Substituting the value of $\sum_{s=0}^{K-1} E\big[\chi_i(s)\big]$ from (19) and choosing $\phi = \frac{m^\gamma}{K^{1/3}}$, we obtain (20). This concludes the proof ■.

Remark 1. *In this work, we have used fixed step-sizes, which are dependent on K. This is because the required accuracy ϵ is usually known in advance, which in turn allows us to calculate the number of iterations K to be run, as we have done in* (23)-(24) *following the result in Corollary 1. This is indeed a commonly adopted practice in both theoretical analysis and practical implementations (see, for example, the works [21], [28], [33], [32]), and also allows for relatively easier analysis. However, there might be situations where ϵ is not known in advance, and in such cases one can follow the proof technique presented in the current work, along with some minor modifications, to show that the use of a time-dependent step-size scheme given by $\eta_k = m^{1/3}/(1+k)^{2/3}$, $c_k = m^{1/2}/(1+k)^{1/2}$, $\theta_k = 1/(m^{1/3}(1+k)^{1/3})$ at iteration k, in Algorithm 1, yields a $O(log(k)/(m^{1/3}k^{1/3}))$ convergence rate, which ultimately translates to $O((log(1/\epsilon))^3/(m\epsilon^2))$ computational cost and $O((log(1/\epsilon))^3/(m\epsilon^3))$ communication cost. Thus, there in an extra $(log(1/\epsilon))^3$ factor when using the time-dependent step-size scheme described above as compared to the result obtained in* (23)-(24) *using fixed step-sizes, which is not too onerous.*

V. Further Insights and Robustness

Note that setting θ to 1 in *Algorithm 1* reduces it to the algorithms proposed in [17], [19]- [22]. One of the key contributions of this work is to show that one can obtain similar number of iterations as in previous works with a significantly smaller value of θ (which in turn leads to reduction in computational complexity). To see this in more detail, we consider the parameters η, c and θ as given by (13). Leaving the exponent of K in θ as a free parameter τ, we have,

$$\eta = \frac{m^\alpha}{K^{2/3}}, \quad c = \frac{m^\beta}{K^{1/2}}, \quad \theta = \frac{m^\gamma}{K^\tau}. \quad (46)$$

Here we clarify that the dependencies of the parameters η and c in (13) (and by extension in (46)) on K are chosen so as to obtain the best possible dependence on K in the RHS of (44), which is then reflected in the result obtained in Theorem 1 [5]. With the values of η and c decided, we try to determine the range of values that τ can take, for which, we take into account the relationships between the step-sizes η, c and θ that need to be satisfied. These are given by (37), (41) and (43). Plugging in the values of η, c and θ from (46) in (37), (41) and (43), it is straight forward to see that τ needs to be in the range $0 \leq \tau < 2/3$ for the inequalities (37), (41) and (43) to hold for sufficiently large and finite K. We then plug in the values from (46) in (44) and recall that $||\boldsymbol{X}(0)-\overline{\boldsymbol{x}}(0)\mathbf{1}_m^T||_F^2 = 0$ holds from the initialization condition to obtain the following bound for $0 \leq \tau < 2/3$.

$$\frac{1}{K}\sum_{t=0}^{K-1} E\big[||\nabla f\big(\overline{\boldsymbol{x}}(t)\big)||^2\big] \leq O\Big(\frac{m^{-\alpha}}{K^{1/3}}\Big)\big[f\big(\overline{\boldsymbol{x}}(0)\big)-f^*\big] + O\Big(\frac{m^{(2\beta-1-\alpha)}}{K^{1/3}}+\frac{m^\beta}{K^{1/2}}\Big)\sigma_c^2 + O\Big(\frac{m^{(\alpha-1-\gamma)}}{K^{2/3-\tau}}+\frac{m^{(2\alpha-\beta-\gamma)}}{K^{5/6-\tau}}\Big)\sigma_g^2 + O\Big(\frac{m^{(\alpha-1-\gamma)}}{K^{2/3-\tau}}+\frac{m^{2(\alpha-\beta)}}{K^{1/3}}+\frac{m^{(2\alpha-\beta-\gamma)}}{K^{5/6-\tau}}+\frac{m^\alpha}{K^{2/3}}\Big)\sigma_G^2. \quad (47)$$

Restricting our focus solely on the dependence on K, the above expression simplifies to

$$\frac{1}{K}\sum_{t=0}^{K-1} E\big[||\nabla f\big(\overline{\boldsymbol{x}}(t)\big)||^2\big] \leq \begin{cases} O\Big(\frac{1}{K^{1/3}}\Big) & \text{if } 0 \leq \tau \leq 1/3. \\ O\Big(\frac{1}{K^{2/3-\tau}}\Big) & \text{if } 1/3 < \tau < 2/3. \end{cases} \quad (48)$$

Remarkably, this suggests that as long as $\tau \leq 1/3$, the dependence of the convergence rate on the number of iterations K remains unaffected and scales as $O(1/K^{1/3})$, thus matching the iteration and communication complexities of previous works while achieving order of magnitude reduction in the computational complexity if $\tau > 0$ (specifically, when expressed in terms of the required accuracy ϵ, the computational and communication costs, for $0 < \tau \leq 1/3$, scale as $O(1/\epsilon^{(3(1-\tau))})$ and $O(1/\epsilon^3)$ respectively with the $O(1/\epsilon^{(3(1-\tau))})$ computational cost improving over the $O(1/\epsilon^3)$ computational cost obtained in [21] and the $O(1/\epsilon^3)$ communication cost matching the communication cost obtained in [21]). Intuitively, we believe that this is due to the fact that the convergence rate of the algorithm is slowed down

[5]This can be verified by initially setting the exponents of K in η and c as free parameters, say, τ_1 and τ_2 and then performing algebraic calculations to optimize for the dependence on K in the RHS of (44)

more by the noise introduced in the information exchange process than by the noise introduced due to stochastic gradient computation. This is also visible from the fact the if we set $\tau = 0$ in (47), the convergence rate decay term associated with the variance of the stochastic gradient σ_g^2 scales as $1/K^{2/3}$, which is a much smaller decay term as compared to the $1/K^{1/3}$ decay term associated with the the communication noise variance σ_c^2, the network averaged gradient disagreement σ_G^2 and the initialization error term $f\big(\overline{\boldsymbol{x}}(0)\big) - f^*$. As a result, one can get away with even worse gradient estimates (gradient estimates having higher variance than σ_g^2) without any effect on the dependence of the convergence rate on the number of iterations K up to a certain extent. This "certain extent" is quantified in (48) through the choice of τ in the parameter θ, which governs how frequently the stochastic gradient computation step takes place and is given by the range $0 \leq \tau \leq 1/3$. However, once $\tau > 1/3$ (i.e. the gradient computation probability starts decreasing further), the convergence rate starts deteriorating and no longer retains the $O(1/K^{1/3})$ dependence as evident from (48). In (13) in Theorem 1, we have chosen the highest possible value of τ to allow for the lowest value of the probability of stochastic gradient computation, while at the same time retaining the best possible overall convergence rate of $O(1/K^{1/3})$. As it turns out from (48), this condition amounts to choosing $\tau = 1/3$.

The above discussion also suggests that the proposed method is robust to environments where the availability of computational resources is unreliable. This can occur in situations where there are frequent power outages or straggler nodes in the network, which make it difficult to compute the stochastic gradient reliably across all iterations. This problem is particularly relevant in today's day and age where models are becoming increasingly complex and data-hungry, as discussed in the *Introduction* section. Such computationally unreliable environments with random failures in stochastic gradient computation can be modelled using the framework developed in *Algorithm 1*. For example, one could model power outages to cause successful stochastic gradient computation to occur in a given iteration with probability $\theta < 1$. Until now, it was up to the algorithm developer to choose the value of θ, where as here, the value of θ will be set by the environment. Remarkably, our result shows that if zero-mean bounded variance communication noise is present, unreliable computation does not have any effect on the overall convergence of the algorithm as long as the value of τ lies in the range $0 \leq \tau \leq 1/3$. To give an example with realistic numbers, suppose a particular application requires the accuracy to be $\epsilon = 0.01$. Then, the number of iterations (i.e. K) to be run would scale as $1/\epsilon^3 = 1000000$. Our result in (48) then suggests that the number of iterations would remain unaffected as long as the probability of successful stochastic gradient computation is above a certain small threshold which scales as $1/K^{1/3}$, which in the current example evaluates to 0.01. This means that power outages could randomly disrupt the stochastic gradient computation step roughly 99 out of 100 times on average and it still would not affect the overall convergence rate, which from (48) would still enjoy a $1/K^{1/3}$ dependence.

VI. Numerical Experiments

We consider a network consisting of $m = 20$ nodes, which we generate randomly, such that for each pair of nodes i, j, the probability of an edge connecting them is 0.75. The network matrix $\boldsymbol{W} \in R^{20\times 20}$ is set to $\boldsymbol{W} = \boldsymbol{I}_{20} - \big(3/4\lambda_{max}(\boldsymbol{L})\big)\boldsymbol{L}$, where $\boldsymbol{L} \in R^{20\times 20}$ is the Laplacian of the graph and $\lambda_{max}(\boldsymbol{L})$ is its maximum eigenvalue.

We consider the *Phishing* dataset which is publicly available at *openML.org*. It consists of a total of 11056 data points and 30 features. We select the first 10000 data points and all 30 features to construct our loss function, for which we assign 500 non-overlapping data points to each of the 20 nodes. For our study, we consider the following local non-convex loss functions at node i, the first of which is a robust regression model ([36]), and the second is a non-convex logistic regression model ([37]).

$$f_i(\boldsymbol{x}) = \frac{1}{500}\sum_{j=1}^{500} \frac{(\langle \boldsymbol{a}_{ij}, \boldsymbol{x}\rangle - b_{ij})^2}{1 + (\langle \boldsymbol{a}_{ij}, x\rangle - b_{ij})^2}, \tag{49}$$

$$f_i(\boldsymbol{x}) = \frac{1}{500}\sum_{j=1}^{500} log\big(1 + e^{-\langle \boldsymbol{a}_{ij}, \boldsymbol{x}\rangle b_{ij}}\big) + \lambda \sum_{k=1}^{d} \frac{x_k^2}{1 + x_k^2}. \tag{50}$$

Here, $d = 30$, $\boldsymbol{a}_{ij} \in R^{30}$ are the feature vectors at node i and $b_{ij} \in R$ are the corresponding labels. In (50), x_k denotes the entry at the k^{th} position of the column vector $\boldsymbol{x} \in R^{30}$ and $\lambda = 0.0001$ denotes the regularization parameter. Among the first 10000 points in the dataset, there are 4437 data points with label -1 (i.e. their b_{ij} value is -1), 1 data point with label 0 and 5562 data points with label 1. Nodes 1-8 are assigned data points with label -1, node 9 is assigned data points having a mix of labels -1 and 1 and the single data point with label 0, and nodes 10-20 are assigned data points with label 1. This ensures that there is sufficient heterogeneity in the data available at different nodes ([34]), which is often a critical issue encountered in decentralized optimization problems.

In the current work, we have incorporated the skipping technique in the widely used *Noisy Consensus + (Stochastic) Gradient Descent* framework to create *Algorithm 1*. We shall abbreviate the *Noisy Consensus + (Stochastic) Gradient Descent* method as `NCSGD` for brevity and use `NCSGD+skip` to abbreviate the *Noisy Consensus + (Stochastic) Gradient Descent* method with the skipping technique incorporated in it. In fact, `NCSGD` can be obtained as a special case of `NCSGD+skip` by setting the gradient computation probability as $\theta = 1$ in `NCSGD+skip`. In our experiments, for both `NCSGD` and `NCSGD + skip`, we consider stochastic gradients (specifically, we use minibatch size $= 1$ for stochastic gradient computation by each node), and use zero-mean Gaussian noise with variance 0.1 to model communication noise (specifically, the entries of $\boldsymbol{N}(t)$ in (31) are drawn from a Gaussian distribution with mean 0 and variance 0.1.).

The results corresponding to (49) and (50) are shown in Fig. 1 and Fig. 2 respectively, with each plot averaged over 100 runs, and the step-size parameters corresponding to each plot appropriately tuned over a grid. In particular, for both (49)

and (50), we consider NCSGD first and tune the parameters η and c over the grids $\{0.0033, 0.01, 0.033, 0.1, 0.33\}$ for (49) and $\{0.01, 0.033, 0.1, 0.33, 1\}$ for (50), meaning that we test out $5 \times 5 = 25$ combinations of parameters for each of considered loss functions. Following this, for each of the considered loss functions (49) and (50), we consider NCSGD + skip, for which we use the previously tuned values of η and c from the vanilla NCSGD method, and tune θ over the grid $\{1/2, 1/3, 1/4, 1/5\}$ conditioned on these fixed choices for η and c (note here that we do not consider all $5 \times 5 \times 4$ combinations for the triplet (η, c, θ)).

In both Figs. 1(a) and 2(a), we study the decrease in the loss, i.e., the average of the global objective function evaluated at the different nodes' estimates $(1/m)\sum_{i=1}^{m} f(x_i(t))$ as a function of the total number of stochastic gradient computations up to time t (i.e.,$\sum_{s=0}^{t}\sum_{i=1}^{m}\chi_i(s)$). Here, it is clear that the skipping technique is able to reduce computational cost since NCSGD+skip requires fewer stochastic gradient computations to reach a certain value of the loss, as compared to NCSGD. Similarly, in Figs. 1(b) and 2(b), we study the decrease in the quantity $(1/m)\sum_{i=1}^{m} f(x_i(t))$ as a function of the iteration number t. Here again we see that there is very little degradation in the overall convergence of the algorithm when the skipping technique is introduced. This confirms reduction in computational cost while retaining overall algorithm convergence as claimed earlier in the paper.

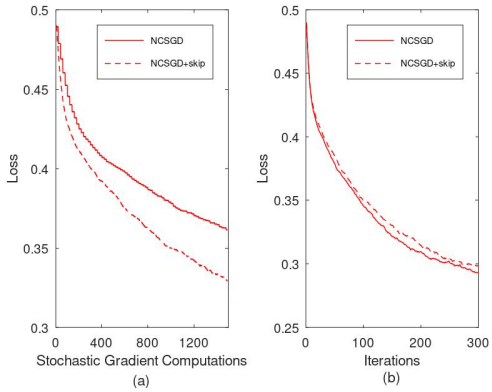

Fig. 1: Robust regression problem (49)

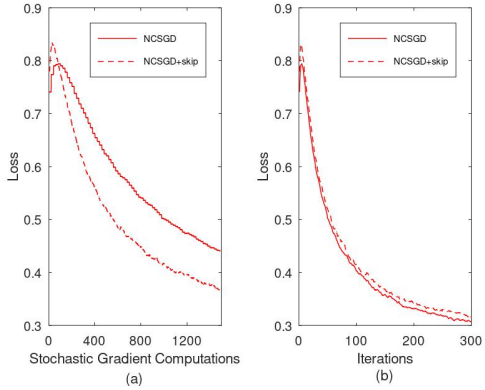

Fig. 2: Non-convex logistic regression problem (50)

VII. Conclusion and Future Work

In this paper, we incorporated a skipping technique in the *Noisy Consensus + Stochastic Gradient Descent* framework, and showed that it helps us achieve a significant reduction in computational cost without hurting overall algorithm convergence. In future, we wish to extend this skipping technique to other famous algorithms, e.g., *Robust Gradient Tracking* based algorithms ([16], [18]) for decentralized optimization in the presence of zero-mean and bounded variance communication noise, and conduct theoretical analysis of the same to derive convergence rate, and computational and communication costs.

References

[1] F. Bullo, J. Cortes, and S. Martinez, *Distributed control of robotic networks: A mathematical approach to motion coordination algorithms,* Princeton University Press, 2009.

[2] P. Forero, A. Cano, and G. Giannakis, "Consensus-Based Distributed Support Vector Machines," *Journal of Machine Learning Research*, vol. 11, no. 55, pp. 1663–1707, 2010.

[3] V. Cevher, S. Becker and M. Schmidt, "Convex Optimization for Big Data: Scalable, randomized, and parallel algorithms for big data analytics," *IEEE Signal Processing Magazine*, vol. 31, no. 5, pp. 32-43, Sept. 2014, doi: 10.1109/MSP.2014.2329397.

[4] M. Stolpe, "The Internet of Things: Opportunities and challenges for distributed data analysis," *ACM SIGKDD Explorations Newslett.*, vol. 18, no. 1, pp. 15–34, June 2016.

[5] L. Gan, U. Topcu, and S. Low, "Optimal Decentralized Protocol for Electric Vehicle Charging," *IEEE Transactions on Power Systems*, vol. 28, no. 2, pp. 940-951, 2013.

[6] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Transactions on Automatic Control*, vol. 54, no. 1, pp. 48–61, 2009.

[7] G. Qu and N. Li, "Harnessing smoothness to accelerate distributed optimization," *IEEE Transactions on Control of Network Systems*, vol. 5, no. 3, pp. 1245–1260, 2017.

[8] B. Swenson, R. Murray , H. V. Poor, and S. Kar, "Distributed stochastic gradient descent: Nonconvexity, nonsmoothness, and convergence to local minima," *Journal of Machine Learning Research*, vol. 23, no. 1, pp. 14751-14812, 2022.

[9] E. Wei and A. Ozdaglar, "Distributed Alternating Direction Method of Multipliers," in *2012 51st IEEE Conference on Decision and Control (CDC)*, Maui, HI, USA, 2012, pp. 5445-5450.

[10] W. Shi, Q. Ling, K. Yuan, G. Wu, and W. Yin, "On the linear convergence of the ADMM in decentralized consensus optimization," *IEEE Transactions on Signal Processing*, vol. 62, no. 7, pp. 1750–1761, 2014.

[11] A. Mokhtari, Q. Ling, and A. Ribeiro, "Network Newton distributed optimization methods," *IEEE Transactions on Signal Processing*, vol. 65, no. 1, pp. 146–161, 2016.

[12] R. Tutunov, H. Bou-Ammar and A. Jadbabaie, "Distributed Newton Method for Large-Scale Consensus Optimization," *IEEE Transactions on Automatic Control*, vol. 64, no. 10, pp. 3983-3994, Oct. 2019.

[13] Y. Li, P. G. Voulgaris, D. M. Stipanović and N. M. Freris, "Communication Efficient Curvature Aided Primal-Dual Algorithms for Decentralized Optimization," *IEEE Transactions on Automatic Control*, vol. 68, no. 11, pp. 6573-6588, Nov. 2023.

[14] S. Liang, L. Y. Wang and G. Yin, "Distributed Smooth Convex Optimization With Coupled Constraints," *IEEE Transactions on Automatic Control*, vol. 65, no. 1, pp. 347-353, Jan. 2020.

[15] S. Kar and J. M. F. Moura, "Distributed Consensus Algorithms in Sensor Networks With Imperfect Communication: Link Failures and Channel Noise," *IEEE Transactions on Signal Processing*, vol. 57, no. 1, pp. 355-369, Jan. 2009.

[16] S. Pu, "A Robust Gradient Tracking Method for Distributed Optimization over Directed Networks," in *2020 59th IEEE Conference on Decision and Control (CDC)*, Jeju, Korea (South), 2020, pp. 2335-2341.

[17] H. Reisizadeh, B. Touri and S. Mohajer, "Distributed Optimization Over Time-Varying Graphs With Imperfect Sharing of Information," *IEEE Transactions on Automatic Control*, vol. 68, no. 7, pp. 4420-4427, July 2023.

[18] Y. Wang and T. Başar, "Gradient-Tracking-Based Distributed Optimization With Guaranteed Optimality Under Noisy Information Sharing," *IEEE Transactions on Automatic Control*, vol. 68, no. 8, pp. 4796-4811, Aug. 2023.

[19] T. T. Doan, S. T. Maguluri and J. Romberg,"Convergence Rates of Distributed Gradient Methods Under Random Quantization: A Stochastic Approximation Approach," *IEEE Transactions on Automatic Control*, vol. 66, no. 10, pp. 4469-4484, Oct. 2021.

[20] A. Reisizadeh, A. Mokhtari, H. Hassani and R. Pedarsani, "An Exact Quantized Decentralized Gradient Descent Algorithm," *IEEE Transactions on Signal Processing*, vol. 67, no. 19, pp. 4934-4947, 2019.

[21] A. Reisizadeh, H. Taheri, A. Mokhtari, H. Hassani, and R. Pedarsani, "Robust and Communication-Efficient Collaborative Learning," in *Advances in Neural Information Processing Systems (NeurIPS) 33*, pp. 8388-8399, 2019.

[22] M. M. Vasconcelos, T. T. Doan and U. Mitra, "Improved Convergence Rate for a Distributed Two-Time-Scale Gradient Method under Random Quantization," in *2021 60th IEEE Conference on Decision and Control (CDC)*, Austin, TX, USA, pp. 3117-3122, 2021.

[23] J. He, L. Cai and X. Guan, "Differential Private Noise Adding Mechanism and Its Application on Consensus Algorithm," *IEEE Transactions on Signal Processing*, vol. 68, pp. 4069-4082, 2020.

[24] Y. Wang and T. Başar, "Quantization Enabled Privacy Protection in Decentralized Stochastic Optimization," *IEEE Transactions on Automatic Control*, vol. 68, no. 7, pp. 4038-4052, July 2023.

[25] Y. Wang and A. Nedić, "Tailoring Gradient Methods for Differentially Private Distributed Optimization," *IEEE Transactions on Automatic Control*, vol. 69, no. 2, pp. 872-887, Feb. 2024.

[26] Z. Allen-Zhu, Y. Li, and Y. Liang, "Learning and generalization in overparameterized neural networks, going beyond two layers," in *Advances in Neural Information Processing Systems (NeurIPS) 33*, pp. 6155–6166, 2019.

[27] X. Hu, L. Chu, L, and J. Pei, "Model complexity of deep learning: a survey," *Springer Knowledge Information Systems*, vol. 63, pp. 2585–2619, 2021.

[28] S. Ghadimi and G. Lan, "Stochastic first-and zeroth-order methods for nonconvex stochastic programming," *SIAM Journal on Optimization*, vol. 23, no. 4, pp. 2341–2368, 2013.

[29] Y. Arjevani, Y. Carmon, J. C. Duchi, D. J. Foster, N. Srebro, B. Woodworth, "Lower bounds for non-convex stochastic optimization," *Mathematical Programming*, vol. 199, pp. 165–21, 2023.

[30] S. Bubeck, "Convex optimization: Algorithms and complexity," *Foundations and Trends in Machine Learning*, vol. 8, no. 3-4, pp 231-357, 2015.

[31] Y. E. Nesterov, *Introductory Lectures on Convex Optimization: a basic course,* Kluwer Academic Publishers, Massachusetts, 2004.

[32] X. Lian, C. Zhang, H.Zhang, C.Hsieh, W.Zhang, J. Liu, "Can Decentralized Algorithms Outperform Centralized Algorithms? A Case Study for Decentralized Parallel Stochastic Gradient Descent," in *Advances in Neural Information Processing Systems (NeurIPS) 30*, pp. 5330-5340, 2017.

[33] O. Dekel, R. Gilad-Bachrach, O. Shamir, and L. Xiao, "Optimal distributed online prediction using mini-batches," *Journal of Machine Learning Research*, vol. 13, pp. 165–202, 2012.

[34] H. Tang, X. Lian, M. Yan, C. Zhang, and J. Liu, "D2: Decentralized training over decentralized data," in *Proceedings of the 35 th International Conference on Machine Learning, Stockholm, Sweden*, vol. 80, pp. 4848-4856, 2018.

[35] R. Mohammad, L. McCluskey, Phishing Websites [Dataset] (2012), *UCI Machine Learning Repository.* https://doi.org/10.24432/C51W2X.

[36] A. E. Beaton and J. W. Tukey, "The fitting of power series, meaning polynomials, illustrated on band-spectroscopic data," *Technometrics*, vol. 16, no. 3, pp. 147–185, 1974.

[37] A. Antoniadis, I. Gijbels, and M. Nikolova, "Penalized likelihood regression for non-quadratic penalties," *Annals of the Institute of Statistical Mathematics*, vol. 63, no. 3, pp. 585–615, 2011.

APPENDIX

A. Proofs of auxiliary results

Let $\mathcal{F}_t$ denote the history of the algorithm up to time t.

Proof of Lemma 2: The proof follows trivially by post-multiplying both the LHS and the RHS of (31) by $\frac{1}{m}\mathbf{1}_m$ and noting that $\boldsymbol{W}_c$ is doubly stochastic meaning $\boldsymbol{W}_c^T\mathbf{1}_m = \mathbf{1}_m$. ■

Proof of Lemma 3: From (31) and (32), we have

$$\begin{aligned}
&\boldsymbol{X}(t+1) - \overline{\boldsymbol{x}}(t+1)\mathbf{1}_m^T \\
&= \big(\boldsymbol{X}(t) - \overline{\boldsymbol{x}}(t)\mathbf{1}_m^T\big)\boldsymbol{W}_c^T - \eta\big(\boldsymbol{H}(t) - \overline{\boldsymbol{h}}(t)\mathbf{1}_m^T\big) \\
&\quad + c\big(\boldsymbol{N}(t) - \overline{\boldsymbol{n}}(t)\mathbf{1}_m^T\big) \\
&= \big(\boldsymbol{X}(t) - \overline{\boldsymbol{x}}(t)\mathbf{1}_m^T\big)\boldsymbol{W}_c^T - \eta\big(\nabla F(\boldsymbol{X}(t)) - \overline{\nabla F}(\boldsymbol{X}(t))\mathbf{1}_m^T\big) \\
&\quad - \eta\big(\boldsymbol{H}(t) - \overline{\boldsymbol{h}}(t)\mathbf{1}_m^T\big) + \eta\big(\nabla F(\boldsymbol{X}(t)) - \overline{\nabla F}(\boldsymbol{X}(t))\mathbf{1}_m^T\big) \\
&\quad + c\big(\boldsymbol{N}(t) - \overline{\boldsymbol{n}}(t)\mathbf{1}_m^T\big). \qquad (51)
\end{aligned}$$

Denoting

$$\begin{aligned}
\boldsymbol{\mathcal{A}}_1(t) &= \big(\boldsymbol{X}(t) - \overline{\boldsymbol{x}}(t)\mathbf{1}_m^T\big)\boldsymbol{W}_c^T \\
&\quad - \eta\big(\nabla F(\boldsymbol{X}(t)) - \overline{\nabla F}(\boldsymbol{X}(t))\mathbf{1}_m^T\big), \qquad (52) \\
\boldsymbol{\mathcal{A}}_2(t) &= -\eta\big[\boldsymbol{H}(t) - \overline{\boldsymbol{h}}(t)\mathbf{1}_m^T\big] \\
&\quad + \eta\big[\nabla F(\boldsymbol{X}(t)) - \overline{\nabla F}(\boldsymbol{X}(t))\mathbf{1}_m^T\big] \\
&\quad + c\big(\boldsymbol{N}(t) - \overline{\boldsymbol{n}}(t)\mathbf{1}_m^T\big), \qquad (53)
\end{aligned}$$

we get

$$\begin{aligned}
&E\big[\big|\big|\boldsymbol{X}(t+1) - \overline{\boldsymbol{x}}(t+1)\mathbf{1}_m^T\big|\big|_F^2\big|\mathcal{F}_t\big] \\
&= ||\boldsymbol{\mathcal{A}}_1(t)||_F^2 + E\big[||\boldsymbol{\mathcal{A}}_2(t)||_F^2\big|\mathcal{F}_t\big] + 2E\big[\langle\boldsymbol{\mathcal{A}}_1(t), \boldsymbol{\mathcal{A}}_2(t)\rangle\big|\mathcal{F}_t\big] \\
&= ||\boldsymbol{\mathcal{A}}_1(t)||_F^2 + E\big[||\boldsymbol{\mathcal{A}}_2(t)||_F^2\big|\mathcal{F}_t\big]. \qquad (54)
\end{aligned}$$

We bound $||\boldsymbol{\mathcal{A}}_1(t)||_F^2$ as follows.

$$\begin{aligned}
||\boldsymbol{\mathcal{A}}_1(t)||_F^2 &\overset{(a)}{\leq} \left(1+\phi_t\right)||\big(\boldsymbol{X}(t) - \overline{\boldsymbol{x}}(t)\mathbf{1}_m^T\big)W_c^T||_F^2 \\
&\quad + \left(1+\frac{1}{\phi_t}\right)\eta^2||\nabla F(\boldsymbol{X}(t)) - \overline{\nabla F}(\boldsymbol{X}(t))\mathbf{1}_m^T||_F^2 \\
&\overset{(b)}{\leq} \big(1-(1-\omega)c\big)||\boldsymbol{X}(t) - \overline{\boldsymbol{x}}(t)\mathbf{1}_m^T||_F^2 \\
&\quad + \frac{2\eta^2}{(1-\omega)c}||\nabla F(\boldsymbol{X}(t))||_F^2 \\
&\overset{(b)}{\leq} \big(1-(1-\omega)c\big)||\boldsymbol{X}(t) - \overline{\boldsymbol{x}}(t)\mathbf{1}_m^T||_F^2 \\
&\quad + \frac{4\eta^2}{(1-\omega)c}||\nabla F(\boldsymbol{X}(t)) - \nabla F\big(\overline{\boldsymbol{x}}(t)\mathbf{1}_m^T\big)||_F^2 \\
&\quad + \frac{4\eta^2}{(1-\omega)c}||\nabla F\big(\overline{\boldsymbol{x}}(t)\mathbf{1}_m^T\big)||_F^2 \\
&\leq \big(1-(1-\omega)c\big)||\boldsymbol{X}(t) - \overline{\boldsymbol{x}}(t)\mathbf{1}_m^T||_F^2 \\
&\quad + \frac{4\eta^2L^2}{(1-\omega)c}||\boldsymbol{X}(t) - \overline{\boldsymbol{x}}(t)\mathbf{1}_m^T||_F^2 \\
&\quad + \frac{4\eta^2}{(1-\omega)c}||\nabla F\big(\overline{\boldsymbol{x}}(t)\mathbf{1}_m^T\big)||_F^2, \qquad (55)
\end{aligned}$$

where (a) follows from Young's inequality for some $\phi_t \geq 0$ and (b) follows by substituting $\phi_t = (1-\omega)c$. We bound

$E[||\boldsymbol{\mathcal{A}}_2(t)||^2|\mathcal{F}_t]$ as follows.

$$
\begin{aligned}
&E\big[||\boldsymbol{\mathcal{A}}_2(t)||_F^2\big|\mathcal{F}_t\big] \\
&\leq c^2E\big[\big|\big|\boldsymbol{N}(t)-\overline{\boldsymbol{n}}(t)\mathbf{1}_m^T\big|\big|_F^2\big|\mathcal{F}_t\big] \\
&\quad+\eta^2E\big[\big|\big|\big(\boldsymbol{H}(t)-\nabla F(\boldsymbol{X}(t))\big) \\
&\qquad\qquad-\big(\overline{\boldsymbol{h}}(t)-\overline{\nabla F}(\boldsymbol{X}(t))\big)\mathbf{1}_m^T\big|\big|_F^2\big|\mathcal{F}_t\big] \\
&\leq c^2E\big[\big|\big|\boldsymbol{N}(t)\big|\big|_F^2\big|\mathcal{F}_t\big]+\eta^2E\big[\big|\big|\boldsymbol{H}(t)-\nabla F(\boldsymbol{X}(t))\big|\big|_F^2\big|\mathcal{F}_t\big] \\
&\overset{(a)}{\leq} mc^2\sigma_c^2+\frac{\eta^2}{\theta}\big(m\sigma_g^2+||\nabla F(\boldsymbol{X}(t))||_F^2\big) \\
&\leq mc^2\sigma_c^2+\frac{m\eta^2\sigma_g^2}{\theta}+\frac{2\eta^2L^2}{\theta}||\boldsymbol{X}(t)-\overline{x}(t)\mathbf{1}_m^T||_F^2 \\
&\quad+\frac{2\eta^2}{\theta}||\nabla F\big(\overline{\boldsymbol{x}}(t)\mathbf{1}_m^T\big)||_F^2, \qquad (56)
\end{aligned}
$$

where in (a), we have used the result from Lemma 1. Combining (54), (55) and (56) and taking expectations with respect to $\mathcal{F}_t$, we obtain the desired result. ■

Proof of Lemma 4: Using L-smoothness of the global objective function

$$
\begin{aligned}
&f\big(\overline{\boldsymbol{x}}(t+1)\big) \\
&\leq f\big(\overline{\boldsymbol{x}}(t)\big)+\big\langle\nabla f\big(\overline{\boldsymbol{x}}(t)\big),\overline{\boldsymbol{x}}(t+1)-\overline{\boldsymbol{x}}(t)\big\rangle \\
&\quad+\frac{L}{2}||\overline{\boldsymbol{x}}(t+1)-\overline{\boldsymbol{x}}(t)||^2 \\
&=f\big(\overline{\boldsymbol{x}}(t)\big)-\big\langle\nabla f\big(\overline{\boldsymbol{x}}(t)\big),\eta\overline{\boldsymbol{h}}(t)-c\overline{\boldsymbol{n}}(t)\big\rangle \\
&\quad+\frac{L}{2}||\eta\overline{\boldsymbol{h}}(t)-c\overline{\boldsymbol{n}}(t)||^2. \qquad (57)
\end{aligned}
$$

Taking expectations conditioned on $\mathcal{F}_t$, we get

$$
\begin{aligned}
&E\big[f\big(\overline{\boldsymbol{x}}(t+1)\big)\big|\mathcal{F}_t\big] \\
&\leq f\big(\overline{\boldsymbol{x}}(t)\big)-\eta\big\langle\nabla f\big(\overline{\boldsymbol{x}}(t)\big),\overline{\nabla F}\big(\boldsymbol{X}(t)\big)\big\rangle \\
&\quad+\frac{\eta^2L}{2}E\big[||\overline{\boldsymbol{h}}(t)||^2\big|\mathcal{F}_t\big]+\frac{c^2L\sigma_c^2}{2m}. \qquad (58)
\end{aligned}
$$

We observe that

$$
\overline{\boldsymbol{h}}(t)=\frac{1}{m}\sum_{i=1}^m\underbrace{\big[\boldsymbol{h}_i(t)-\nabla f_i\big(\boldsymbol{x}_i(t)\big)\big]}_{\boldsymbol{\delta}_i(t)}+\frac{1}{m}\sum_{i=1}^m\nabla f_i\big(\boldsymbol{x}_i(t)\big).
$$

We note that the set of random variables $\{\boldsymbol{\delta}_i(t)\}$, $i\in V$ are zero-mean independent random variables conditioned on $\mathcal{F}_t$. Consequently, we have

$$
\begin{aligned}
&E\big[||\overline{\boldsymbol{h}}(t)||^2\big|\mathcal{F}_t\big] \\
&\leq\frac{1}{m^2}\sum_{i=1}^m E\big[||\boldsymbol{\delta}_i(t)||^2\big|\mathcal{F}_t\big]+\frac{1}{m^2}\big|\big|\sum_{i=1}^m\nabla f_i\big(\boldsymbol{x}_i(t)\big)\big|\big|^2 \\
&\overset{(a)}{\leq}\frac{1}{\theta m^2}\sum_{i=1}^m\big[\sigma_g^2+||\nabla f_i\big(\boldsymbol{x}_i(t)\big)||^2\big]+\frac{1}{m}\sum_{i=1}^m\big|\big|\nabla f_i\big(\boldsymbol{x}_i(t)\big)\big|\big|^2 \\
&\leq\frac{\sigma_g^2}{\theta m}+\left(\frac{1}{\theta m^2}+\frac{1}{m}\right)\big|\big|\nabla F\big(\boldsymbol{X}(t)\big)\big|\big|_F^2 \\
&\leq\frac{\sigma_g^2}{\theta m}+2L^2\left(\frac{1}{\theta m^2}+\frac{1}{m}\right)\big|\big|\boldsymbol{X}(t)-\overline{\boldsymbol{x}}(t)\mathbf{1}_m^T\big|\big|_F^2 \\
&\quad+2\left(\frac{1}{\theta m^2}+\frac{1}{m}\right)\big|\big|\nabla F\big(\overline{\boldsymbol{x}}(t)\mathbf{1}_m^T\big)\big|\big|_F^2, \qquad (59)
\end{aligned}
$$

where (a) follows from the result in (6b). Next, we observe that

$$
\begin{aligned}
&-\eta\big\langle\nabla f\big(\overline{\boldsymbol{x}}(t)\big),\overline{\nabla F}\big(\boldsymbol{X}(t)\big)\big\rangle \\
&=-\eta\big\langle\nabla f\big(\overline{\boldsymbol{x}}(t)\big),\nabla f\big(\overline{\boldsymbol{x}}(t)\big)-\overline{\nabla F}\big(\boldsymbol{X}(t)\big)\big\rangle-\eta||\nabla f\big(\overline{\boldsymbol{x}}(t)\big)||^2 \\
&\leq\frac{\eta}{2}\big|\big|\nabla f\big(\overline{\boldsymbol{x}}(t)\big)-\overline{\nabla F}\big(\boldsymbol{X}(t)\big)\big|\big|^2-\frac{\eta}{2}||\nabla f\big(\overline{\boldsymbol{x}}(t)\big)||^2 \\
&\leq\frac{\eta}{2m}\sum_{i=1}^m||\nabla f_i\big(\overline{\boldsymbol{x}}(t)\big)-\nabla f_i\big(\boldsymbol{x}_i(t)\big)||^2-\frac{\eta}{2}||\nabla f\big(\overline{\boldsymbol{x}}(t)\big)||^2 \\
&\leq\frac{\eta L^2}{2m}||\boldsymbol{X}(t)-\overline{\boldsymbol{x}}(t)\mathbf{1}_n^T||_F^2-\frac{\eta}{2}||\nabla f\big(\overline{\boldsymbol{x}}(t)\big)||^2. \qquad (60)
\end{aligned}
$$

Combining (58)-(60) and taking expectations w.r.t. $\mathcal{F}_t$, we obtain the desired result. ■

Soham Mukherjee Soham Mukherjee obtained his B.Tech and M.Tech degrees in Electronics and Electrical Communication Engineering from the Indian Institute of Technology Kharagpur in 2021. He is currently working as a quantitative analyst in the financial services industry. His current research interests include optimization and learning in large scale settings.

Mrityunjoy Chakraborty (Senior Member, IEEE) received bachelor of engineering degree in electronics and telecommunication engineering from Jadavpur University, Kolkata, India in 1983, master of technology degree in electrical engineering from the Indian Institute of Technology Kanpur, India in 1985, and the Doctor of Philosophy degree in electrical engineering from the Indian Institute of Technology Delhi, India in 1994.

He joined the Department of Electronics and Electrical Communication Engineering, Indian Institute of Technology Kharagpur, India, as a Lecturer in 1994, where he currently holds the position of a senior professor. He has been the chair of the department during 2021-2023 and held the prestigious Prithviraj Banerjee and Swati Banerjee Chair Professor position during 2022-2024. His research interests include digital and adaptive signal processing, linear algebra, optimization, compressive sensing and graph signal processing.

He has been an Associate Editor of the IEEE Transactions on Circuits and Systems (TCAS), Part I during 2004–2007 and 2010–2012, and of the TCAS-II during 2008–2009 and 2022-2023. During 2017—2020, he was a Senior Editorial Board (SEB) Member of the IEEE Signal Processing Magazine and during 2016-2017, he was a SEB Member of the IEEE Journal on Emerging Techniques in Circuits and Systems. During 2016—2018, he served as the Chair of the DSP Technical Committee (TC) of the IEEE Circuits and Systems Society. He has also been the Guest Editor of the EURASIP Journal on Advances in Signal Processing and Special Issues of the TCAS II, the DSP Track Co-Chair of ISCAS 2015–2023, the TPC Co-Chair of IEEE SIPS-2018, the Special Session Co-Chair of DSP-18, and the Gabor Track Chair of DSP-15. He is a Co-Founder of the Asia Pacific Signal and Information Processing Association (APSIPA), was a Member of the APSIPA BOG during 2013–2016, and was also the Chair of the APSIPA TC on Signal and Information Processing Theory and Methods. In 2012 and 2020, he was the General Chair of the National Conference on Communications.

Prof. Chakraborty is a Fellow of the National Academy of Sciences, India, and the Indian National Academy of Engineering (INAE). Recently, he received the prestigious Chair Professorship of the INAE. During 2012–2013, he was selected as a Distinguished Lecturer of the APSIPA.