


Echo-CGC: A Communication-Efficient Byzantine-Tolerant Distributed Machine Learning Algorithm in Single-Hop Radio Network

Qinzi Zhang

Boston College, Chestnut Hill, MA, USA
zhangbcu@bc.edu

Lewis Tseng 

Boston College, Chestnut Hill, MA, USA
lewis.tseng@bc.edu

Abstract

In the past few years, many Byzantine-tolerant distributed machine learning (DML) algorithms have been proposed in the point-to-point communication model. In this paper, we focus on a popular DML framework – the parameter server computation paradigm and iterative learning algorithms that proceed in rounds, e.g., [11, 8, 6]. One limitation of prior algorithms in this domain is the *high communication complexity*. All the Byzantine-tolerant DML algorithms that we are aware of need to send n d -dimensional vectors from worker nodes to the parameter server in each round, where n is the number of workers and d is the number of dimensions of the feature space (which may be in the order of millions). In a wireless network, power consumption is proportional to the number of bits transmitted. Consequently, it is extremely difficult, if not impossible, to deploy these algorithms in power-limited wireless devices. Motivated by this observation, we aim to reduce the *communication complexity* of Byzantine-tolerant DML algorithms in the *single-hop radio network* [1, 3, 14].

Inspired by the CGC filter developed by Gupta and Vaidya, PODC 2020 [11], we propose a gradient descent-based algorithm, Echo-CGC. Our main novelty is a mechanism to utilize the *broadcast properties* of the radio network to avoid transmitting the raw gradients (full d -dimensional vectors). In the radio network, each worker is able to overhear previous gradients that were transmitted to the parameter server. Roughly speaking, in Echo-CGC, if a worker “agrees” with a combination of prior gradients, it will broadcast the “echo message” instead of its raw local gradient. The echo message contains a vector of coefficients (of size at most n) and the ratio of the magnitude between two gradients (a float). In comparison, the traditional approaches need to send n local gradients in each round, where each gradient is typically a vector in a ultra-high dimensional space ($d \gg n$). The improvement on communication complexity of our algorithm depends on multiple factors, including number of nodes, number of faulty workers in an execution, and the cost function. We numerically analyze the improvement, and show that with a large number of nodes, Echo-CGC reduces 80% of the communication under standard assumptions.

2012 ACM Subject Classification Computing methodologies → Distributed computing methodologies; Computing methodologies → Machine learning

Keywords and phrases Distributed Machine Learning, Single-hop Radio Network, Byzantine Fault, Communication Complexity, Wireless Communication, Parameter Server

Digital Object Identifier 10.4230/LIPIcs.OPODIS.2020.7

Related Version A full version of the paper is available at <https://arxiv.org/abs/2011.07447>.

Acknowledgements The authors would like to acknowledge Nitin H. Vaidya and anonymous reviewers for their helpful comments.



© Qinzi Zhang and Lewis Tseng;

licensed under Creative Commons License CC-BY

24th International Conference on Principles of Distributed Systems (OPODIS 2020).

Editors: Quentin Bramas, Rotem Oshman, and Paolo Romano; Article No. 7; pp. 7:1–7:16

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

Machine learning has been widely adopted and explored recently [23, 16]. Due to the exponential growth of datasets and computation power required, distributed machine learning (DML) becomes a necessity. There is also an emerging trend [21, 13] to apply DML in power-limited wireless networked systems, e.g., sensor networks, distributed robots, smart homes, and Industrial Internet-of-Things (IIoT), etc. In these applications, the devices are usually small and fragile, and susceptible to malicious attacks and/or malfunction. More importantly, it is necessary to reduce communication complexity so that (over-)communication does not drain the device battery. Most prior research on fault-tolerant DML (e.g., [8, 4, 11, 6]) has focused on the use cases in clusters or datacenters. These algorithms achieve high resilience (number of faults tolerated), but also incur *high communication complexity*. As a result, most prior Byzantine-tolerant DML algorithms are extremely difficult, if not impossible, to be deployed in power-limited wireless networks.

Motivated by our observations, we aim to design a Byzantine DML algorithm with reduced communication complexity. We consider wireless systems that are modeled as a *single-hop radio network*, and focus on the popular parameter server computation paradigm (e.g., [11, 8, 6]). We propose *Echo-CGC*, and prove its correctness under typical assumptions [4, 8]. For the communication complexity, we formally analyze the expected number of bits that need to be sent from workers to the parameter server. The extension to multi-hop radio network is left as an interesting future work.

Recent Development in Distributed Machine Learning. Distributed Machine Learning (DML) is designed to handle a large amount of computation over big data. In the parameter server model, there is a centralized parameter server that distributes the computation tasks to n workers. These workers have the access to the same dataset (that may be stored externally). Similar to [4, 11, 6], we focus on the *synchronous gradient descent* DML algorithms, where the server and workers proceed in synchronous rounds. In each round, each worker computes a local gradient over the parameter received from the server, and the server then aggregates the gradients collected from workers, and updates the parameter. Under suitable assumptions, prior algorithms [4, 11, 6] converge to the optimal point in the d -dimensional space \mathbb{R}^d even if up to f workers may become Byzantine faulty.

To our knowledge, most Byzantine-tolerant DML or distributed optimization algorithms focused on the case of clusters and datacenters, which are modeled as a point-to-point network. For example, Reference [6], Krum [4], Kardam [7], and ByzSGD [8] focused on the stochastic gradient descent algorithms under several different settings (synchronous, asynchronous, and distributed parameter server). Reference [20, 11, 19] focused on the gradient descent algorithms for the general distributed optimization framework. Zeno [24] uses failure detection to improve the resilience. None of these works aimed to reduce communication complexity.

Another closely related research direction is on reducing the communication complexity of non-Byzantine-tolerant DML algorithms, e.g., [15, 13, 22]. These algorithms are *not* Byzantine fault-tolerant, and adopt a completely different design. For example, reference [15] utilizes relaxed consistency (of the underlying shared data), reference [22] discards coordinates (of the local gradients) aggressively, and reference [13] uses intermediate aggregation. It is not clear how to integrate these techniques with Byzantine fault-tolerance, as these approaches reduce the redundancy, making it difficult to mask the impact from Byzantine workers.

Single-Hop Radio Network. We consider the problem in a single-hop radio network, which is a proper theoretical model for wireless networks. Following [1, 3, 14], we assume that single-hop wireless communication is reliable and authenticated, and there is no jamming nor spoofing. Moreover, nodes follow a specific TDMA schedule so that there is no collision. In Section 2.1, we briefly argue why such an assumption is realistic to model wireless communication. In the single-hop radio network model, we aim to minimize the total number of bits to be transmitted in each round. If we directly adapt prior gradient descent-based algorithms [4, 11] to the radio network model, then each worker needs to broadcast a vector of size d , where d is the number of dimensions of the feature space. In practical applications (e.g., [9, 13]), d might be in the order of millions, and the gradients may require a few GBs. Since power consumption is proportional to the communication complexity in wireless channel, prior Byzantine DML algorithms are *not* adequate for power-limited wireless networks..

Main Contributions. Inspired by the CGC filter developed by Gupta and Vaidya, PODC 2020 [11], we propose a gradient descent-based algorithm, *Echo-CGC*, for the parameter server model in the single-hop radio network. Our main observation is that since workers can overhear gradients transmitted earlier, they can use this information to avoid sending the raw gradients in some cases. Particularly, if a worker “agrees” with some reference gradient(s) transmitted earlier in the same round, then they send a small message to “echo” with the reference gradient(s). The size of the echo message ($O(n)$ bits) is negligible compared to the raw gradient ($O(d)$ bits), since in typical ML applications, $d \gg n$.

Our proof is more sophisticated than the one in [11], even though Echo-CGC is inspired by the CGC filter. The reason is that the “echo message” does *not* necessarily contain worker i ’s local gradient; instead, it can be used to construct an approximate gradient, which intuitively equals a combined gradients between i ’s local gradients and the gradients broadcast by previous workers. We need to ensure that such an approximation does not affect the aggregation at the server. Moreover, CGC filter [11] works on deterministic gradients – each worker computes the gradient of its local cost function using the full dataset. In our case, each worker computes a stochastic gradient, a gradient over a small random data batch. We prove that with appropriate assumptions, Echo-CGC converges to the optimal point.

Echo-CGC is correct under the same set of assumptions in prior work [4]; however, there is an inherent trade-off between resilience, the proven bound on the communication complexity reduction, and the cost function. Fix the cost function. We derive necessary conditions on n so that Echo-CGC is guaranteed to perform better. We also perform numerical analysis to understand the trade-off. In general, Echo-CGC saves more and more communication if f/n becomes smaller and smaller. Moreover, our algorithm performs better when the variance of the data is relatively small. For example, our algorithm tolerates 10% of faulty workers and saves over 75% of communication cost when standard deviation of computed gradients is less than 10% of the true gradient.

2 Preliminaries

In this section, we formally define our models, and introduce the assumptions and notations.

2.1 Models

Single-Hop Radio Network. We consider the standard radio network model in the literature, e.g., [1, 3, 14]. In particular, the underlying communication layer ensures the *reliable local broadcast* property [3]. In other words, the channel is perfectly reliable, and a local broadcast

is correctly received by all neighbors. As noted in [1, 3], this assumption does not typically hold in the current deployed wireless networks, but it is possible to realize such a property with high probability in practice with the help from the MAC layer [2] or physical layer [17].

In our system, nodes can be uniquely identified, i.e., each node has a unique identifier. We assume that a faulty node may not spoof another node’s identity. The communication network is assumed to be single-hop; that is, each pair of nodes are within the communication range of each other. Moreover, time is divided into slots, and each node proceeds synchronously. Message collision is not possible because of the nodes follow a pre-determined TDMA schedule that determine the transmitting node in each slot and the transmission protocol is jam-resistant. Each slot is assumed to be large enough so that it is possible for a node to transmit a gradient. We also assume that each communication round (or communication step) is divided into n slots, and the TDMA schedule assigns each node to a unique slot. For ease of discussion, node i is scheduled to transmit at slot i .

Stochastic Gradient Descent and Parameter Server. In this work, we focus on the Byzantine-tolerant distributed Stochastic Gradient Descent (SGD) algorithms, which are popular in the optimization and machine learning literature [4, 8, 11, 5]. Given a cost function Q , the (sequential) SGD algorithm outputs an optimal parameter w^* such that

$$w^* = \operatorname{argmin}_{w \in \mathbb{R}^d} Q(w) \quad (1)$$

An SGD algorithm executes in an iterative fashion, where in each round t , the algorithm computes the gradient of the cost function Q at parameter w^t and updates the parameter with the gradient.

Synchronous Parameter Server Model: Computation of gradients is typically expensive and slow. One popular framework to speed up the computation is the *parameter server model*, in which the parameter server distributes the computation tasks to n workers and aggregates their computed gradients to update the parameter in each round. Following the convention, we will use node and worker interchangeably.

We assume a synchronous system, i.e., the computation and communication delays are bounded, and the server and workers know the bound. Consequently, if the server does not receive a message from worker i by the end of some round, then the server identifies that worker i is faulty.

Formally speaking, a distributed SGD algorithm in the parameter server model proceeds in synchronous rounds, and executes the following three steps in each round t :

1. The parameter server broadcasts parameter w^t to the workers.
2. Each worker j randomly chooses a random data batch ξ_j^t from the dataset (shared by all the workers) and computes an estimate, g_j^t , of the gradient $\nabla Q(w^t)$ of the cost function Q using ξ_j^t and w^t .
3. The server aggregates estimated gradients from all workers and updates the parameter using the gradient descent approach with step size η :

$$w^{t+1} = w^t - \eta \sum_{j=1}^n g_j^t \quad (2)$$

Fault Model and Byzantine SGD. Following [11, 4, 6], our system consists of n workers, up to f of which might be Byzantine faulty. We assume that the central parameter server is always fault-free.

Byzantine workers may be controlled by an omniscient adversary which has the knowledge of the current parameter (at the server) and the local gradient of all the other workers, and may have arbitrary behaviors. They *can* send arbitrary messages. However, due to the reliable local broadcast property of the radio network model, they *cannot* send inconsistent messages to the server and other workers. They also *cannot* spoof another node's identity. Our goal is therefore to design a distributed SGD algorithm that solves Equation (1) in the presence of up to f Byzantine workers.

Workers that are *not* Byzantine faulty are called fault-free workers. These workers follow the algorithm specification faithfully. For a given execution of the algorithm, we denote \mathcal{H} as the set of fault-free workers and \mathcal{B} as the set of Byzantine workers. For brevity, we denote $h = |\mathcal{H}|$ and $b = |\mathcal{B}|$; hence, we have $b \leq f$ and $h \geq n - f$.

Communication Complexity. We are interested in minimizing the total number of bits that need to be transmitted from workers to the parameter server in *each round*. Prior algorithms [11, 4] transmit n gradients in a d -dimensional space in each round, since each node needs to transmit its local gradient to the centralized server. Typically, each gradient consists of d floats or doubles (i.e., a single primitive floating point data structure for each dimension).

2.2 Assumptions and Notations

We assume that the cost function Q satisfies some standard properties used in the literature [4, 8, 6], including convexity, differentiability, Lipschitz smoothness, and strong convexity. Following the convention, we use $\langle a, b \rangle$ to represent the dot product of two vectors a and b in the d -dimensional space \mathbb{R}^d .

► **Assumption 1** (Convexity and smoothness). Q is convex and differentiable.

► **Assumption 2** (L -Lipschitz smoothness). There exists $L > 0$ such that for all $w, w' \in \mathbb{R}^d$,

$$\|\nabla Q(w) - \nabla Q(w')\| \leq L\|w - w'\| \quad (3)$$

► **Assumption 3** (μ -strong convexity). There exists $\mu > 0$ such that for all $w, w' \in \mathbb{R}^d$,

$$\langle \nabla Q(w) - \nabla Q(w'), w - w' \rangle \geq \mu\|w - w'\|^2 \quad (4)$$

We also assume that the random data batches are independently and identically distributed from the dataset. Before stating the assumptions, we formally introduce the concept of randomness in the framework. Similar to typical stochastic gradient descent algorithms, the only randomness is due to the random data batches ξ_j^t sampled by each fault-free worker $j \in \mathcal{H}$ in each round t , which further makes g_j^t as well as w^{t+1} non-deterministic. In the case when a worker uses the entire dataset to train model, $g_j^t = \nabla Q(w^t)$. Hence, the result is deterministic, i.e., each fault-free worker derives the same gradient. In practice, data batch is a small sample of the entire data set.¹

Formally speaking, we denote an operator $\mathbb{E}_{\Xi^t}(\cdot \mid w^t, \mathcal{G}_{\mathcal{B}}^t)$ as the *conditional expectation* operator over the set of random batches $\Xi^t = \{\xi_j^t, j = 1, 2, \dots, n\}$ in round t given (i) the parameter w^t , and (ii) the set of Byzantine gradients $\mathcal{G}_{\mathcal{B}}^t = \{g_j^t : j \in \mathcal{B}\}$. This conditional expectation operator allows us to treat w^t , $Q(w^t)$, and $\nabla Q(w^t)$ as constants, as well as the

¹ Reference [11] works on a different formulation in which each worker may have a different local cost function.

7:6 Echo-CGC: A Communication-Efficient Byzantine DML

Byzantine gradients. This is reasonable because (i) we have the knowledge about Q and w^t given an execution, and (ii) the Byzantine gradients are arbitrary, and do not depend on the data batches. From now on, without further specification, we abbreviate the operator $\mathbb{E}_{\Xi^t}(\cdot | w^t, \mathcal{G}_{\mathcal{B}}^t)$ as \mathbb{E} .

Below we present two further assumptions of local stochastic gradient g_j^t at each fault-free worker j . Similar to [4, 8], we rely on the two following assumptions for correctness proof.

► **Assumption 4** (IID Random Batches). *For all $j \in \mathcal{H}$ and $t \in \mathbb{N}$,*

$$\mathbb{E}(g_j^t) = \nabla Q(w^t) \quad (5)$$

► **Assumption 5** (Bounded Variance). *For all $j \in \mathcal{H}$ and $t \in \mathbb{N}$,*

$$\mathbb{E}\|g_j^t - \nabla Q(w^t)\|^2 \leq \sigma^2 \|\nabla Q(w^t)\|^2 \quad (6)$$

Notation. We list the most important notations and constants used in our algorithm and analysis in the following table.

■ **Table 1** Notations and constants used in this paper.

\mathcal{H}	set of fault-free workers; $h = \mathcal{H} $
\mathcal{B}	set of faulty workers; $b = \mathcal{B} $
t	round number, $t = 0, 1, 2, \dots$
w^*	optimal solution to Q , i.e., $w^* = \operatorname{argmin}_{w \in \mathbb{R}^d} Q(w)$
w^t	parameter in round t
g_j^t	estimated gradient of j in round t
\tilde{g}_j^t	“reconstructed” gradient of j by server in round t
\hat{g}_j^t	gradient of j in round t after applying the CGC filter
η	fixed step size as in Equation (2)
L	Lipschitz constant
μ	strong convexity constant
r	deviation ratio, a key parameter in our algorithm
k^*	constant defined in Lemma 2, $k^* \approx 1.12$

3 Our Algorithm: Echo-CGC

Our algorithm is inspired by Gupta and Vaidya [11]. Specifically, we integrate their CGC filter with a novel aggregation phase. Our aggregation mechanism utilizes the broadcast property of the radio network to improve the communication complexity. In the CGC algorithm [11], each worker needs to send a d -dimensional gradient to the server, whereas in our algorithm, some workers only need to send the “echo message” which is of size $O(n)$ bits. Note that in typical machine learning applications, $d \gg n$.

We design our algorithm for the synchronous parameter server model, so the algorithm is presented in an iterative fashion. That is, each worker and the parameter server proceed in synchronous rounds, and the algorithm specifies the exact steps for each round t . Our Algorithm, Echo-CGC, is presented in Algorithm 1. The algorithm uses the notations and constants summarized in Table 1.

Algorithm Description

Initially, the parameter server randomly generates an initial parameter $w^0 \in \mathbb{R}^d$. Each round $t \geq 0$ consists of three phases: (i) computation phase, (ii) communication phase, and (iii) aggregation phase. Echo-CGC takes the following inputs: step size η , deviation ratio r , number of workers n , and maximum number of tolerable faults f . The exact requirements on the values of these inputs will become clear later. For example, n, f, r need to satisfy the bound derived in Lemma 3. More discussion will be presented in Section 4.3.

Computation Phase. In the computation phase of round t , the server broadcasts w^t to the workers. Each worker j then computes the local stochastic gradient $g_j^t = \nabla Q_j(w^t)$ using w^t and its random data batch ξ_j^t . Since we assume the parameter server is fault-free, each worker receives the identical w^t . The local gradient is stochastic, because each worker uses a random data batch to compute the local gradient g_j^t .

Communication Phase. In the communication phase, each worker needs to send the information regarding to its local gradient to the parameter server. This phase is our main novelty, and different from prior algorithms [11, 4, 6]. We utilize the property of the broadcast channel to reduce the communication complexity. As mentioned earlier, the communication phase of round t is divided into n slots t_1, \dots, t_n . Without loss of generality, we assume that each worker j is scheduled to broadcast its information in slot t_j (of round t). Note that we assume that the underlying physical or MAC layer is jamming-resistant and reliable; hence, each fault-free worker can reliably broadcast the information to all the other nodes.

Steps for Worker j . Each worker j stores a set of gradients that it overhears in round t . Denote by R_j the set of stored gradients. By assumption, R_j consists of gradients g_i^t for $i < j$, when at the beginning of slot t_j . Upon receiving a gradient g_i^t (in the form of a vector in \mathbb{R}^d), worker j stores it to R_j if g_i^t is linearly independent with all existing gradients in R_j . In the slot t_j , worker j computes the ‘‘echo gradient’’ using vectors stored in R_j . Specifically, worker j takes the following steps:

- It expresses R_j as $R_j = \{g_{i_1}^t, \dots, g_{i_{|R_j|}}^t\}$ and constructs a matrix $A_j \in \mathbb{R}^{d \times |R_j|}$ as

$$A_j^t = \begin{bmatrix} g_{i_1}^t & g_{i_2}^t & \cdots & g_{i_{|R_j|}}^t \end{bmatrix}$$

- It then computes the Moore-Penrose inverse (M-P inverse in short) of A_j^t , defined as

$$(A_j^t)^+ = ((A_j^t)^T A_j^t)^{-1} (A_j^t)^T,$$

where A^T is the transpose of matrix A . The existence of the M-P inverse is guaranteed. Intuitively this is because all columns of A_j^t are linearly independent by construction. The formal proof is presented in our full paper [25].

- Next, worker j computes a vector $x_j^t \in \mathbb{R}^{|R_j|}$ using the M-P inverse:

$$x_j^t = (A_j^t)^+ g_j^t,$$

where g_j^t is the local stochastic gradient of Q computed by j in the computation phase. Note that x_j^t is of size $O(n)$, since R_j contains at most n elements.

- Finally, it computes the ‘‘echo gradient’’ as

$$(g_j^t)^* = A_j^t x_j^t$$

Mathematically, $(g_j^t)^*$ is the projection of g_j^t onto the span of vectors in R_j , i.e., the closest vector to g_j^t in the span of R_j .

Next, worker j checks whether the following inequality holds where $(g_j^t)^*$ is the echo gradient, g_j^t the local stochastic gradient, and r the deviation ratio.

$$\|(g_j^t)^* - g_j^t\| \leq r \|g_j^t\| \quad (7)$$

Worker j performs one of the two actions depending on the result of Inequality (7).

- If Inequality (7) holds, then j sends the *echo message* $(\|g_j^t\|/(\|g_j^t\|)^*, x_j^t, I_j^t)$ to the server, where $I_j^t = \{i_1, \dots, i_{|R_j|}\}$ is a sorted list of worker IDs whose gradients are stored in R_j .
- Otherwise, worker j broadcasts the raw gradient g_j^t to server and all the other workers.

Steps for Parameter Server: The parameter server uses a vector G to store the gradients from workers. Specifically, in each round t , for each worker j , the server computes \tilde{g}_j^t and stores it as the j -th element of G . At the beginning of round t , every element $G[j]$ is initialized as an empty placeholder \perp . During the communication phase, the parameter server takes two possible actions upon receiving a message from worker j :

- If the message is a vector, then the server stores $\tilde{g}_j^t = g_j^t$ in $G[j]$.
- Otherwise, the message is a tuple (k, x, I) . The server then does the following:
 - If there exists some $i \in I$ such that $G[i] = \perp$ (i.e., the server has not received a message from worker i), then due to the reliable broadcast property, the server can safely identify j as a Byzantine worker. By convention, we let the server store $\tilde{g}_j^t = \vec{0}$, the zero vector in \mathbb{R}^d , in $G[j]$.
 - Otherwise, denote the matrix A_I as $A_I = [G[i_1], \dots, G[i_{|R_j|}]]$ where $I = \{i_1, \dots, i_{|R_j|}\}$, and the server stores \tilde{g}_j^t as $\tilde{g}_j^t = kA_I x$ in $G[j]$.

Aggregation Phase. The final phase is identical to the algorithm in [11], in which the server updates the parameter using the CGC filter. First, the server sorts the stored gradients G^t in the increasing order of their Euclidean norm and relabel the IDs so that $\|\tilde{g}_{i_1}^t\| \leq \dots \leq \|\tilde{g}_{i_n}^t\|$. Then the server applies the CGC filter as follows:

$$\hat{g}_j^t = \begin{cases} \frac{\|\tilde{g}_{i_{n-f}}^t\|}{\|\tilde{g}_j^t\|} \tilde{g}_j^t, & j \in \{i_{n-f+1}, \dots, i_n\} \\ \tilde{g}_j^t, & j \in \{i_1, \dots, i_{n-f}\} \end{cases} \quad (8)$$

Finally, the server aggregates the gradients by $g^t = \sum_{j=1}^n \hat{g}_j^t$ and updates the parameter by $w^{t+1} = w^t - \eta g^t$, where η is the fixed step size.

4 Convergence Analysis

In this section, we prove the convergence of our algorithm Echo-CGC. The proof is more complicated than the one in [11], even though both algorithms use the CGC filter. This is mainly due to two reasons: (i) we use stochastic gradient, whereas [11] uses a deterministic gradient; and (ii) echo messages only results in an approximate gradient (i.e., the echo gradient which may be deviated from the local stochastic gradient by a ratio r). Intuitively, in addition to the Byzantine tampering, we need to deal with non-determinism from stochastic gradients and noise from echo messages.

4.1 Convergence Rate Analysis

In this part, we first analyze the *convergence rate* ρ , which is a constant defined later in Equation (13). Recall a few notations that $h = |\mathcal{H}|$ and $b = |\mathcal{B}|$, where given the execution, \mathcal{H} is the set of fault-free workers and \mathcal{B} is the set of Byzantine workers. Also recall that

Algorithm 1 Algorithm Echo-CGC.

```

1: Parameters:
2:    $\eta > 0$  is the step size defined in Equation (2)
3:    $r > 0$  is the deviation ratio
4:    $n, f, r$  satisfy the resilience bounds stated in Lemma 3
5: Initialization at server:  $w^0 \leftarrow$  a random vector in  $\mathbb{R}^d$ 
6: for  $t \leftarrow 0$  to  $\infty$  do
7:   /* Computation Phase */
8:   At server: broadcast  $w^t$  to all workers;  $G \leftarrow$  a  $\perp$ -vector of length  $n$ 
9:   At worker  $j$ :
10:    receive  $w^t$  from the server
11:     $g_j^t \leftarrow \nabla Q_j(w^t)$ ;  $R_j \leftarrow \{\}$  ▷ local stochastic gradient at worker  $j$ 
12:   /* Communication Phase */
13:   for  $i \leftarrow 1$  to  $n$  do
14:     (i) At worker  $i$ :
15:     if  $|R_i| = 0$  then
16:       broadcast  $g_i^t$ 
17:     else
18:        $A \leftarrow [g]_{g \in R_j}$ ;  $A^+ \leftarrow (A^T A)^{-1} A^T$ ;  $x \leftarrow A^+ g_i^t$  ▷  $Ax$  is the echo gradient
19:       if  $\|Ax - g_i^t\| \leq r \|g_i^t\|$  then
20:          $I \leftarrow \{i' : g_{i'}^t \in R_j\}$  in an ascending order
21:         broadcast  $(\|g_i^t\| / \|Ax\|, x, I)$  ▷ echo message
22:       else
23:         broadcast  $g_i^t$  ▷ raw local gradient
24:       end if
25:     end if
26:     (ii) At worker  $j > i$ :
27:     if  $j$  receives vector  $g_i^t$  from worker  $i$  then
28:        $A \leftarrow [g]_{g \in R_j}$ ;  $A^+ \leftarrow (A^T A)^{-1} A^T$ 
29:       if  $g_i^t$  is linearly independent with  $R_j$  (i.e.,  $AA^+ g_i^t \neq g_i^t$ ) then
30:          $R_i \leftarrow R_i \cup \{g_i^t\}$ 
31:       end if
32:     end if
33:     (iii) At server:
34:     if it receives a vector  $g_j^t$  from worker  $j$  then
35:        $G[j] \leftarrow g_j^t$  ▷  $j$  transmitted a raw gradient
36:     else if it receives an echo message  $(k, x, I)$  from worker  $j$  then
37:       if  $\exists i \in I$  such that  $G[i] = \perp$  then
38:          $G[j] \leftarrow \vec{0}$  ▷  $j$  is a Byzantine worker
39:       else
40:          $A_I \leftarrow [\tilde{g}_i^t]_{i \in I}$ ,  $G[j] \leftarrow k A_I x$  ▷  $j$  transmitted an echo message
41:       end if
42:     end if
43:   end for
44:   /* Aggregation Phase (applying CGC filter from [11]) */
45:    $g^t \leftarrow \sum_{g \in G} CGC(g)$  ▷  $CGC(\cdot)$  defined in Equation (8)
46:    $w^{t+1} \leftarrow w^t - \eta \cdot g^t$  ▷  $\eta$  defined in Equation (2)
47: end for

```

7:10 Echo-CGC: A Communication-Efficient Byzantine DML

L and μ are the constants defined in the Assumption 2 and 3, respectively; σ defined in Assumption 5; and r is the deviation ratio used in Echo-CGC. To derive ρ , we need to define series of constants based on the given parameters of n, f, h, b, L, μ, r , and σ .

We first define a constant β as

$$\beta = (n - 2f) \frac{\mu - r(1 + \sigma)L}{1 + r} - b(1 + k_h \sigma)L, \quad (9)$$

where k_x is defined as

$$k_x = 1 + \frac{x - 1}{\sqrt{2x - 1}}, \quad \forall x \geq 1. \quad (10)$$

We then define a constant γ as

$$\gamma = nL^2 (h(1 + \sigma^2) + b\alpha_h), \quad (11)$$

where

$$\alpha_x = x\sigma^2 + (1 + k_h \sigma)^2, \quad \forall x \geq 1. \quad (12)$$

Finally, we define the convergence rate ρ using β and γ as follows:

$$\rho = 1 - 2\beta\eta + \gamma\eta^2. \quad (13)$$

We will prove that under some standard assumptions, the convergence rate ρ is in the interval $[0, 1)$. We first present several auxiliary lemmas. Due to page limit, most proofs are presented in the full paper [25].

► **Lemma 1.** *Let $L, \mu > 0$ be the Lipschitz constant and strong convexity constant defined in Assumption 2 and 3, respectively. Then we have $\mu \leq L$.*

► **Lemma 2.** *Denote $k^* = \sup_x \{k_x / \sqrt{x} : x \geq 1\}$. Then $k^* < \infty$, and numerically $k^* \approx 1.12$. Equivalently, $k_h \leq k^* \sqrt{h}$ for all $h \geq 1$.*

► **Lemma 3.** *Assume $n\mu - (3 + k_n \sigma)fL > 0$, then there exists $r > 0$ that satisfies equation below.*

$$r < \frac{n\mu - (3 + k_n \sigma)fL}{(n - 2f)(1 + \sigma)L + (1 + k_n \sigma)fL}. \quad (14)$$

Moreover, if $r > 0$ satisfies Equation (14), then $\beta > 0$.

Lemma 3 implies that we need to bound σ for convergence. In general, Echo-CGC is correct if $\sigma = o(\log n)$. For brevity, we make the following assumption to simplify the proof of convergence and the analysis of communication complexity. We stress that this assumption can be relaxed using basically the same analysis with a denser mathematical manipulation.

► **Assumption 6.** *Let σ be the variance bound defined in Assumption 5. We further assume that $\sigma < \frac{1}{\sqrt{n}}$.*

Under Assumption 6, we can narrow down the bound of r in Lemma 3 to loosen our assumption on fault tolerance.

► **Lemma 4.** *Assume $n\mu - (3 + k^*)fL > 0$ ($k^* \approx 1.12$), then there exists $r > 0$ satisfying Equation (15) such that $\beta > 0$.*

$$r < \frac{n\mu - (3 + k^*)fL}{(n - 2f)(1 + \sigma)L + (1 + k^*)fL}. \quad (15)$$

► **Theorem 5.** *Assume $n\mu - (3 + k^*)fL > 0$ and r is a value that satisfies Inequality (15). Then we can find an $\eta > 0$ such that $\eta < 2\beta/\gamma$, which in turn makes $\rho \in [0, 1)$.*

4.2 Proof of Convergence

Next, we prove the convergence of our algorithm. That is, Echo-CGC converges to the optimal point w^* of the cost function Q . We prove the convergence under the assumption that $n\mu - (3 + k^*)fL > 0$. Due to page limit, we present key proofs here, and the rest can be found in [25].

Recall our definition of the conditional expectation $\mathbb{E} = \mathbb{E}_{\Xi^t}(\cdot \mid w^t, G_B^t)$ introduced in Section 2.2. Before proving the main theorem, we introduce some preliminary lemmas.

► **Lemma 6.** *For all t and for all $j \in \mathcal{H}$,*

$$\mathbb{E}\|g_j^t\| \leq (1 + \sigma)\|\nabla Q(w^t)\|. \quad (16)$$

► **Lemma 7.** *Recall that \hat{g}_j^t is the gradient after applying the CGC filter. For all t and for all $j \in \{1, 2, \dots, n\}$,*

$$\mathbb{E}\|\hat{g}_j^t\| \leq (1 + k_h\sigma)\|\nabla Q(w^t)\|. \quad (17)$$

The proof of Lemma 7 is based on Lemma 6 and the following prior results: Gumbel [10] and Hartley and David [12] proved that given identical means and variances (μ, σ^2) , the upper bound of the expectation of the largest random variable among n independent random variables is $\mu + \frac{\sigma(n-1)}{\sqrt{2n-1}}$.

► **Lemma 8.** *Following the same setup, for all t and for all $j \in \{1, 2, \dots, n\}$,*

$$\mathbb{E}\|\hat{g}_j^t\|^2 \leq \alpha_h\|\nabla Q(w^t)\|^2. \quad (18)$$

The proof of Lemma 8 is based on Lemma 6 and the following result: Papadatos [18] proved that for n i.i.d. random variables $X_1 \leq X_2 \leq \dots \leq X_n$ with finite variance σ^2 , the maximum variance of X_n is bounded above by $n\sigma^2$.

Lemma 7 and Lemma 8 provide upper bounds on $\mathbb{E}\|\hat{g}_j^t\|$ and $\mathbb{E}\|\hat{g}_j^t\|^2$. These two bounds allow us to bound the impact of bogus gradients transmitted by a faulty node j . If j transmitted an extreme gradient, it would be dropped by the CGC filter; otherwise, these two bounds essentially imply that the filtered gradient \hat{g}_j^t has some nice property even if j is faulty. For fault-free gradients, Lemma 6 provides a better bound.

► **Theorem 9.** *Assume that $n\mu - (3 + k^*)fL > 0$. We can find $r > 0$ that satisfies Inequality (15) and $\eta > 0$ such that $\eta < 2\beta/\gamma$. Echo-CGC with the chosen r and η will converge to the optimal parameter w^* as $t \rightarrow \infty$.*

Proof. Our ultimate goal is to show that the sequence $\{\mathbb{E}\|w^t - w^*\|^2\}_{t=0}^\infty$ converges to 0. Recall that the aggregation rule of the algorithm is $w^{t+1} = w^t - \eta g^t$. Thus, we obtain that

$$\begin{aligned} \mathbb{E}\|w^{t+1} - w^*\|^2 &\leq \mathbb{E}\|w^t - w^* - \eta g^t\|^2 \\ &= \underbrace{\mathbb{E}\|w^t - w^*\|^2}_A - \underbrace{2\eta\mathbb{E}\langle w^t - w^*, g^t \rangle}_B + \underbrace{\eta^2\mathbb{E}\|g^t\|^2}_C. \end{aligned} \quad (19)$$

Since w^t is known, w^t can be treated as a constant, and $\mathbb{E}\|w^t - w^*\|^2 = \|w^t - w^*\|^2$.

Part C: In [25], we show that the following inequality holds.

$$\mathbb{E}\|g^t\|^2 \leq \gamma\|w^t - w^*\|^2. \quad (20)$$

Part B: By linearity of inner product,

$$\langle w^t - w^*, g^t \rangle = \sum_{j \in \mathcal{H}} \langle w^t - w^*, \hat{g}_j^t \rangle + \sum_{j \in \mathcal{B}} \langle w^t - w^*, \tilde{g}_j^t \rangle. \quad (21)$$

First, by Schwarz Inequality, $\langle w^t - w^*, \hat{g}_j^t \rangle \geq -\|w^t - w^*\| \|\hat{g}_j^t\|$; by Lemma 7 and L -Lipschitz assumption, $\mathbb{E} \|\hat{g}_j^t\| \leq (1 + k_h \sigma) L \|w^t - w^*\|$. Thus,

$$\mathbb{E} \langle w^t - w^*, \hat{g}_j^t \rangle \geq -(1 + k_h \sigma) L \|w^t - w^*\|^2, \quad \forall j \in \mathcal{B}. \quad (22)$$

Next, observe that by our algorithm, for each $j \in \mathcal{H}$, the received gradient before CGC filter \tilde{g}_j^t satisfies (i) $\|\tilde{g}_j^t\| = \|g_j^t\|$ and (ii) $\tilde{g}_j^t = a_j(g_j^t + \Delta g_j^t)$, for some constant $a_j = \|g_j^t\| / \|g_j^t + \Delta g_j^t\|$ and a vector Δg_j^t such that $\|\Delta g_j^t\| \leq r \|g_j^t\|$. This implies $a_j \geq 1/(1+r)$. Therefore,

$$\begin{aligned} \mathbb{E} \langle w^t - w^*, \tilde{g}_j^t \rangle &= \mathbb{E} \langle w^t - w^*, a_j(g_j^t + \Delta g_j^t) \rangle \\ &\geq \frac{1}{1+r} (\langle w^t - w^*, \mathbb{E} g_j^t \rangle + \mathbb{E} \langle w^t - w^*, \Delta g_j^t \rangle), \quad \forall j \in \mathcal{H}. \end{aligned} \quad (23)$$

By Assumption 4, $\mathbb{E} g_j^t = \nabla Q(w^t)$; by strong convexity,

$$\langle w^t - w^*, \nabla Q(w^t) \rangle \geq \mu \|w^t - w^*\|^2.$$

By Schwarz inequality, $\mathbb{E} \langle w^t - w^*, \Delta g_j^t \rangle \geq -\|w^t - w^*\| \mathbb{E} \|\Delta g_j^t\|$; and $\mathbb{E} \|\Delta g_j^t\| \leq r \mathbb{E} \|g_j^t\|$. By Lemma 6 and L -Lipschitz assumption, $\mathbb{E} \|g_j^t\| \leq (1 + \sigma) L \|w^t - w^*\|$. Thus,

$$\mathbb{E} \langle w^t - w^*, \Delta g_j^t \rangle \geq -r(1 + \sigma) L \|w^t - w^*\|^2.$$

Upon substituting these results into Equation (23), we obtain that

$$\mathbb{E} \langle w^t - w^*, \tilde{g}_j^t \rangle \geq \frac{\mu - r(1 + \sigma)L}{1 + r} \|w^t - w^*\|^2, \quad \forall j \in \mathcal{H}. \quad (24)$$

We partition \mathcal{H} into two parts: $\mathcal{H}_1 = \mathcal{H} \cap \{i_1, \dots, i_{n-f}\}$ and $\mathcal{H}_2 = \mathcal{H} \setminus \mathcal{H}_1$. For each $j \in \mathcal{H}_1$, the received gradient is unchanged by CGC filter, i.e., $\hat{g}_j^t = \tilde{g}_j^t$. Therefore, Equation (24) also holds for \hat{g}_j^t , for all $j \in \mathcal{H}_1$.

The case of \mathcal{H}_2 is similar. Note that for each $j \in \mathcal{H}_2$, the gradient \tilde{g}_j^t is scaled down to \hat{g}_j^t by CGC filter. In other words, there exists some constant $a'_j \geq 0$ such that $\hat{g}_j^t = a'_j \tilde{g}_j^t$. Therefore, by Equation (23),

$$\mathbb{E} \langle w^t - w^*, \hat{g}_j^t \rangle = \mathbb{E} \langle w^t - w^*, a'_j \tilde{g}_j^t \rangle = a'_j \mathbb{E} \langle w^t - w^*, \tilde{g}_j^t \rangle, \quad \forall j \in \mathcal{H}_2.$$

We can verify that if by assumption that $r > 0$ satisfies Equation (15), then $\mu - r(1 + \sigma)L > 0$; and Equation (23) implies that $\mathbb{E} \langle w^t - w^*, \tilde{g}_j^t \rangle \geq 0$. Therefore,

$$\mathbb{E} \langle w^t - w^*, \hat{g}_j^t \rangle \geq 0, \quad \forall j \in \mathcal{H}_2. \quad (25)$$

Note that $|\mathcal{H}_1| \geq h - 2f$. Upon substituting Equation (22), (24), (25) into Equation (21), we obtain that

$$\mathbb{E} \langle w^t - w^*, g^t \rangle \geq \left((n - 2f) \frac{\mu - r(1 + \sigma)L}{1 + r} - b(1 + k_h \sigma)L \right) \|w^t - w^*\|^2. \quad (26)$$

By definition of β in Equation (9), this implies $\mathbb{E} \langle w^t - w^*, g^t \rangle \geq \beta \|w^t - w^*\|^2$.

Conclusion: Upon combining part A, B and C, by definition of ρ in Equation (13),

$$\mathbb{E} \|w^{t+1} - w^*\|^2 \leq \rho \|w^t - w^*\|^2, \quad \forall t = 0, 1, 2, \dots$$

Recall the definition of the conditional expectation operator \mathbb{E} . This implies that

$$\mathbb{E} (\|w^t - w^*\|^2 \mid w^0, \mathcal{G}_{\mathcal{B}}^0, \dots, \mathcal{G}_{\mathcal{B}}^t) \leq \rho^t \|w^0 - w^*\|^2$$

By Theorem 5, $\rho \in (0, 1)$. Therefore, as $t \rightarrow \infty$, $\|w^t - w^*\|^2$ converges to 0. In other words, w^t converges to the optimal parameter w^* . This proves the theorem. \blacktriangleleft

4.3 Communication Complexity

We analyze the communication complexity of the Echo-CGC algorithm, and show that under suitable conditions, it effectively reduces communication complexity compared to prior algorithms [4, 11]. First consider a ball in \mathbb{R}^d whose center is the true gradient $\nabla Q(w^t)$:

$$B(\nabla Q(w^t), \frac{r}{2+r} \|\nabla Q(w^t)\|) = \{u \in \mathbb{R}^d : \|u - \nabla Q(w^t)\| \leq \frac{r}{2+r} \|\nabla Q(w^t)\|\}, \quad (27)$$

where $r > 0$ is the deviation ratio. For a slight abuse of notations, we abbreviate the ball as B . This should not be confused with \mathcal{B} , the set of Byzantine workers. We present only the main results, and the proofs can be found in [25].

► **Lemma 10.** *For all $u, v \in B$, $\|u - v\| \leq r\|u\|$ (and $\|u - v\| \leq r\|v\|$).*

Given Lemma 10, we compute the probability that an arbitrary gradient g_j^t is in the ball B . By Markov's Inequality,

$$\begin{aligned} \Pr(g_j^t \in B) &= \Pr\left(\|g_j^t - \nabla Q(w^t)\|^2 \leq \frac{r^2}{(2+r)^2} \|\nabla Q(w^t)\|^2\right) \\ &\geq 1 - \frac{\mathbb{E}\|g_j^t - \nabla Q(w^t)\|^2}{\frac{r^2}{(2+r)^2} \|\nabla Q(w^t)\|^2}. \end{aligned} \quad (28)$$

By Assumption 5, $\mathbb{E}\|g_j^t - \nabla Q(w^t)\|^2 \leq \sigma^2 \|\nabla Q(w^t)\|^2$, so we conclude that $\Pr(g_j^t \in B) \geq p$, where p is the lower bound defined as $p = 1 - (1 + 2/r)^2 \sigma^2$.

Denote $n_B = |\{j : g_j^t \in B\}|$ and n^* as the number of workers that send the ‘‘echo message’’ in a round. By Lemma 10, $n^* \geq n_B - 1$. Since each event $\{g_j^t \in B\}$ is independent and has a fixed probability, n^* follows a Binomial distribution with success probability $\Pr(g_j^t \in B)$ which is bounded below by p . Therefore,

$$\mathbb{E}n^* \geq \mathbb{E}n_B - 1 \geq np - 1.$$

For $n \gg 1$, we assume that $1/n \approx 0$. Also in practice, $d \gg n$, so the message complexity of each echo message (in $O(n)$ bits) is negligible compared to raw gradients (in $O(d)$ bits). Hence, the ratio of bit complexity of our algorithm and prior algorithms (e.g., [4, 11]) can be approximately bounded above as follows:

$$\begin{aligned} \frac{\text{bit complexity of Echo-CGC}}{\text{bit complexity of prior algorithms}} &= \frac{n^*O(n) + (n - n^*)O(d)}{nO(d)} \\ &\leq \frac{(np - 1)O(n) + [n - (np - 1)]O(d)}{nO(d)} \\ &\approx 1 - p. \end{aligned}$$

We denote the upper bound of ratio of reduced complexity to complexity of prior algorithms as $C = 1 - p = (1 + 2/r)^2 \sigma^2$.

Analysis. By Equation (3) and Lemma 2, C can be expressed as

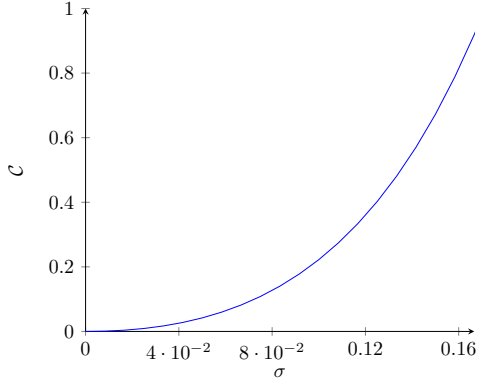
$$C \leq \sigma^2 \left(1 + 2 \cdot \frac{(1 - 2x)(1 + \sigma) + (1 + \sigma k^* \sqrt{n})x}{\mu/L - (3 + \sigma k^* \sqrt{n})x}\right)^2, \quad (29)$$

where $x = f/n$ is the fault-tolerance factor.

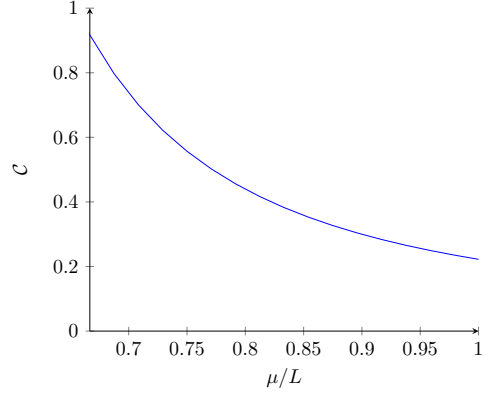
As Equation (29) shows, the ratio C is related to four non-trivial variables: (i) bound of variance $\sigma \geq 0$; (ii) resilience $x = f/n$ satisfying the assumption in Lemma 3, i.e.,

$$\mu/L - (3 + \sigma k^* \sqrt{n})x > 0;$$

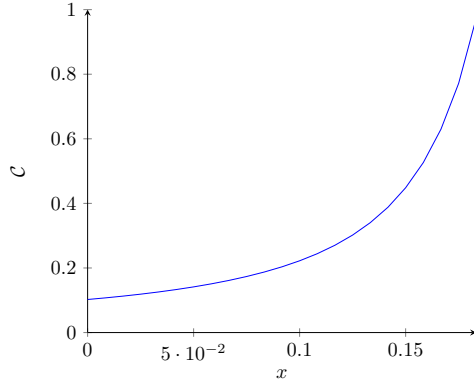
(iii) constant L/μ , which is determined by the cost function Q and satisfies $0 < L/\mu < 1$ by Lemma 1; and (iv) number of workers $n > 0$.



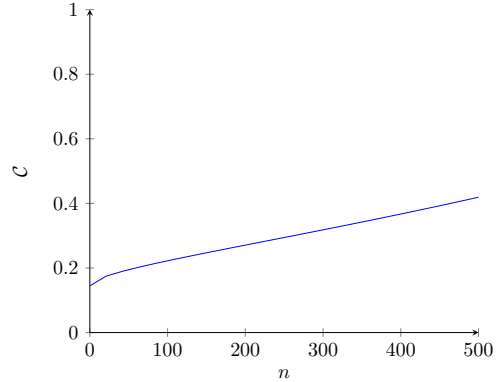
(a) C as a function of σ , for fixed $\mu/L = 1$, $x = 0.1$, and $n = 100$.



(b) C as a function of μ/L , for fixed $\sigma = 0.1$, $x = 0.1$, and $n = 100$.



(c) C as a function of x , for fixed $\sigma = 0.1$, $\mu/L = 1$, and $n = 100$.



(d) C as a function of n , for fixed $\sigma = 0.1$, $\mu/L = 1$, and $x = 0.1$.

We first plot the relation between one factor and C while fixing the other three factors. First, we present the most significant fact, σ . We fix $\mu/L = 1$, $x = 0.1$, and $n = 100$. As Figure 1a shows, C increases in an almost quadratic speed with σ because of the σ^2 term in Equation (29). Therefore, our algorithm is guaranteed to have lower communication complexity when the variance of gradients is relatively low, especially when $\sigma \leq 0.1$. In practice, this is the scenario when the data set consists mainly of similar data instances.

Then, we plot C against μ/L with fixed $\sigma = 0.1$, $x = 0.1$, and $n = 100$. As Figure 1b shows, C decreases as μ/L becomes closer to 1. As $\mu/L > 0.75$, $C < 0.5$, meaning that $[0.75, 1]$ is the range of μ/L where our algorithm is guaranteed to perform significantly better.

Next, we plot C against x with fixed $\sigma = 0.1$, μ/L , and $n = 100$. As Figure 1c shows, there is a trade-off between C and fault resilience x . As x approaches the max resilience defined in Lemma 3, i.e., $x_{\max} = \frac{\mu/L}{(3 + \sigma k^* \sqrt{n})}$, the theoretical upper bound C blows up. Moreover, as $x < 0.15$, $C < 0.4$; and thus $[0, 0.15]$ is a proper range of x .

Finally, we plot C against n with fixed $\sigma = 0.1$, $\mu/L = 1$, and $x = 0.1$. As Figure 1d shows, C increases almost linearly with respect to n with a relatively flat slope. In other words, n is *not* a significant factor of C ; and the performance of our algorithm is stable in a wide range of n .

In conclusion, our algorithm is guaranteed to require lower communication complexity when: (i) σ is low, i.e., data instances are similar and (ii) μ/L is close to 1. Also, there is a trade-off between resilience and efficiency. As a concrete example, when $\sigma = 0.1$, $x = 0.2$, $\mu/L = 1$, and $n = 100$, $C \approx 0.25$, meaning that our algorithm is guaranteed to save at least 75% of communication cost.

5 Summary

In this paper, we present our Byzantine-tolerant DML algorithm that incurs lower communication complexity in a single-hop radio network (under suitable conditions). Our algorithm is inspired by the CGC filter [11], but we need to devise new proofs to handle the randomness and noise introduced in our mechanism.

There are two interesting open problems: (i) multi-hop radio network; and (ii) different mechanism for constructing echo messages, e.g., usage of angles rather than distance ratio.

References

- 1 Dan Alistarh, Seth Gilbert, Rachid Guerraoui, Zarko Milosevic, and Calvin Newport. Securing every bit: Authenticated broadcast in radio networks. In *Proceedings of the Twenty-Second Annual ACM Symposium on Parallelism in Algorithms and Architectures*, SPAA '10, page 50–59, New York, NY, USA, 2010. Association for Computing Machinery. doi:10.1145/1810479.1810489.
- 2 Baruch Awerbuch, Andrea Richa, and Christian Scheideler. A jamming-resistant mac protocol for single-hop wireless networks. In *Proceedings of the Twenty-Seventh ACM Symposium on Principles of Distributed Computing*, PODC '08, page 45–54, New York, NY, USA, 2008. Association for Computing Machinery. doi:10.1145/1400751.1400759.
- 3 Vartika Bhandari and Nitin H. Vaidya. On reliable broadcast in a radio network. In *Proceedings of the Twenty-Fourth Annual ACM Symposium on Principles of Distributed Computing*, PODC '05, page 138–147, New York, NY, USA, 2005. Association for Computing Machinery. doi:10.1145/1073814.1073841.
- 4 Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. Machine learning with adversaries: Byzantine tolerant gradient descent. In *NIPS'17*, page 118–128, Red Hook, NY, USA, 2017. Curran Associates Inc.
- 5 Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, USA, 2004.
- 6 Yudong Chen, Lili Su, and Jiaming Xu. Distributed statistical machine learning in adversarial settings: Byzantine gradient descent. *Proc. ACM Meas. Anal. Comput. Syst.*, 1(2), December 2017. doi:10.1145/3154503.
- 7 Georgios Damaskinos, El Mahdi El Mhamdi, Rachid Guerraoui, Rhicheek Patra, and Mahsa Taziki. Asynchronous Byzantine machine learning (the case of SGD). In Jennifer Dy and Andreas Krause, editors, *Proceedings of Machine Learning Research*, volume 80, pages 1145–1154, Stockholm, Sweden, 2018. PMLR. URL: <http://proceedings.mlr.press/v80/damaskinos18a.html>.
- 8 El-Mahdi El-Mhamdi, Rachid Guerraoui, Arsany Guirguis, Lê Nguyễn Hoàng, and Sébastien Rouault. Genuinely distributed byzantine machine learning. In *Proceedings of the 39th Symposium on Principles of Distributed Computing*, PODC '20, page 355–364, New York, NY, USA, 2020. Association for Computing Machinery. doi:10.1145/3382734.3405695.
- 9 J. Fan and J. Lv. A selective overview of variable selection in high dimensional feature space. *Statistica Sinica*, pages 101–148, January 2010.

- 10 E. J. Gumbel. The maxima of the mean largest value and of the range. *The Annals of Mathematical Statistics*, 25(1):76–84, 1954. URL: <http://www.jstor.org/stable/2236513>.
- 11 Nirupam Gupta and Nitin H. Vaidya. Fault-tolerance in distributed optimization: The case of redundancy. In *Proceedings of the 39th Symposium on Principles of Distributed Computing*, PODC '20, page 365–374, New York, NY, USA, 2020. Association for Computing Machinery. doi:10.1145/3382734.3405748.
- 12 H. O. Hartley and H. A. David. Universal bounds for mean range and extreme observation. *The Annals of Mathematical Statistics*, 25(1):85–99, 1954. URL: <http://www.jstor.org/stable/2236514>.
- 13 Seyyedali Hosseinalipour, Christopher G. Brinton, Vaneet Aggarwal, Huaiyu Dai, and Mung Chiang. From federated learning to fog learning: Towards large-scale distributed machine learning in heterogeneous wireless networks, 2020. arXiv:2006.03594.
- 14 Chiu-Yuen Koo. Broadcast in radio networks tolerating byzantine adversarial behavior. In *Proceedings of the Twenty-Third Annual ACM Symposium on Principles of Distributed Computing*, PODC '04, page 275–282, New York, NY, USA, 2004. Association for Computing Machinery. doi:10.1145/1011767.1011807.
- 15 Mu Li, David G. Andersen, Alexander Smola, and Kai Yu. Communication efficient distributed machine learning with the parameter server. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'14, page 19–27, Cambridge, MA, USA, 2014. MIT Press.
- 16 Ruben Mayer and Hans-Arno Jacobsen. Scalable deep learning on distributed infrastructures: Challenges, techniques, and tools. *ACM Comput. Surv.*, 53(1), February 2020. doi:10.1145/3363554.
- 17 V. Navda, A. Bohra, S. Ganguly, and D. Rubenstein. Using channel hopping to increase 802.11 resilience to jamming attacks. In *IEEE INFOCOM 2007 - 26th IEEE International Conference on Computer Communications*, pages 2526–2530, 2007.
- 18 Nickos Papadatos. Maximum variance of order statistics. *Annals of the Institute of Statistical Mathematics*, 47:185–193, February 1995. doi:10.1007/BF00773423.
- 19 Lili Su and Nitin H. Vaidya. Fault-tolerant multi-agent optimization: Optimal iterative distributed algorithms. In George Giakkoupis, editor, *Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing, PODC 2016, Chicago, IL, USA, July 25-28, 2016*, pages 425–434. ACM, 2016. doi:10.1145/2933057.2933105.
- 20 Lili Su and Nitin H. Vaidya. Non-bayesian learning in the presence of byzantine agents. In *Distributed Computing - 30th International Symposium, DISC 2016, Paris, France, September 27-29, 2016. Proceedings*, pages 414–427, 2016. doi:10.1007/978-3-662-53426-7_30.
- 21 Y. Sun, M. Peng, Y. Zhou, Y. Huang, and S. Mao. Application of machine learning in wireless networks: Key techniques and open issues. *IEEE Communications Surveys Tutorials*, 21(4):3072–3108, 2019.
- 22 Zeyi Tao and Qun Li. esgd: Communication efficient distributed deep learning on the edge. In *USENIX Workshop on Hot Topics in Edge Computing (HotEdge 18)*, Boston, MA, July 2018. USENIX Association. URL: <https://www.usenix.org/conference/hotedge18/presentation/tao>.
- 23 Joost Verbraeken, Matthijs Wolting, Jonathan Katzy, Jeroen Kloppenburg, Tim Verbelen, and Jan S. Rellermeyer. A survey on distributed machine learning. *ACM Comput. Surv.*, 53(2), March 2020. doi:10.1145/3377454.
- 24 Cong Xie, Sanmi Koyejo, and Indranil Gupta. Zeno: Distributed stochastic gradient descent with suspicion-based fault-tolerance. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of Machine Learning Research*, volume 97, pages 6893–6901, Long Beach, California, USA, 2019. PMLR. URL: <http://proceedings.mlr.press/v97/xie19b.html>.
- 25 Qinzi Zhang and Lewis Tseng. Echo-CGC: A communication-efficient byzantine-tolerant distributed machine learning algorithm in single-hop radio network, 2020. arXiv:2011.07447.