Byzantine Resilient Distributed Learning in Multirobot Systems

Jiani Li, Waseem Abbas, *Member, IEEE*, Mudassir Shabbir, and Xenofon Koutsoukos, *Fellow, IEEE*

***Abstract*—Distributed machine learning algorithms are increasingly used in multirobot systems and are prone to Byzantine attacks. In this article, we consider a distributed implementation of the stochastic gradient descent (SGD) algorithm in a cooperative network, where networked agents optimize a global loss function using SGD on the local data and aggregation of the estimates of immediate neighbors. Byzantine agents can send arbitrary estimates to their neighbors, which may disrupt the convergence of normal agents to the optimum state. We show that if every normal agent combines its neighbors' estimates (states) such that the aggregated state is in the convex hull of its normal neighbors' states, then the resilient convergence is guaranteed. To assure this sufficient condition, we propose a resilient aggregation rule based on the notion of *centerpoint*, which is a generalization of the median in the higher-dimensional Euclidean space. We evaluate our results using examples of target pursuit and pattern recognition in multirobot systems. The evaluation results demonstrate that distributed learning with average, coordinate-wise median, and geometric median-based aggregation rules fail to converge to the optimum state, whereas the centerpoint-based aggregation rule is resilient in the same scenario.**

***Index Terms*—Byzantine attacks, centerpoint, multirobot systems, resilient distributed learning and optimization, resilient aggregation.**

I. Introduction

THERE is a growing trend toward collaboratively training machine learning models on distributed devices to deal with the rapid increase of data as well as privacy and security concerns. In this article, we consider the problem of distributed machine learning (DML) in a fully decentralized network [2]–[4]. In such a network, agents interact with each other without a centralized server and leverage the shared information to benefit their learning performance. Such a decentralized framework also addresses the single point of failure problem as well as scalability issues and is naturally suited for applications in multirobot systems, including spectrum sensing in cognitive networks [5], target localization and tracking [6], distributed clustering [7], and biologically inspired designs for mobile networks [8].

Although cooperation among agents helps improve the overall learning performance [2], it is also susceptible to attacks, where noncooperative or adversarial neighbors sharing wrong information can disrupt the convergence of the algorithm. Average-based information aggregation rules have been widely used in DML [2], [3], [9]. However, a single misbehaving agent can adversely impact the convergence of the average-based aggregation rules [10], [11]. Many robust aggregation rules have also been proposed to cope with outliers in data or Byzantine adversaries, including the coordinate-wise median (CM) [12]–[14], coordinate-wise trimmed mean [12], [15], geometric median (GM) [16], [17], Krum and multi-Krum [10], Bulyan, and multi-Bulyan [18], [19]. However, studies have already reported successful attacks in Byzantine systems using the rules mentioned above [20]–[22].

In this article, we study the problem of the resilient convergence of DML for multirobot systems in the presence of Byzantine adversaries. We show that in the aggregation step, if every normal agent in the network combines its neighbors' states such that the aggregated result is in the convex hull of *normal* neighbors' states, then the resilient convergence of DML is guaranteed. We observe that most of the aggregation rules used in the literature do not satisfy this condition, as illustrated in Fig. 1. The primary challenge here is that an agent cannot distinguish between its normal or Byzantine neighbors; hence, it cannot simply discard the information from Byzantine neighbors to satisfy the above condition. To address this challenge, we propose a resilient aggregation rule based on the notion of *safe region*. For a normal agent k with n_k neighbors, of which *any* f can be Byzantine agents, a safe region is the set of states that always lie in the convex hull of agent k's normal neighbors' states. We show that a normal agent can always find a state in the safe region by computing a *centerpoint* of its neighbors' states if $f \leq \lceil \frac{n_k}{d+1} \rceil - 1$, where d is the dimension of states. Thus, the centerpoint-based aggregation guarantees the resilient convergence of DML in the Byzantine system.

Our main contributions are as follows:

1) We analyze the sufficient condition to achieve Byzantine resilient convergence in distributed learning algorithms. The condition guarantees that the state obtained due to

Manuscript received 17 April 2021; revised 29 September 2021 and 14 February 2022; accepted 4 May 2022. Date of publication 13 June 2022; date of current version 6 December 2022. This article was recommended for publication by Associate Editor A. Prorok and Editor P. Robuffo Giordano upon evaluation of the reviewers' comments. *(Corresponding author: Jiani Li.)*

Jiani Li and Xenofon Koutsoukos are with the Department of Computer Science at Vanderbilt University, Nashville, TN 37235 USA (e-mail: jiani.li@vanderbilt.edu; Xenofon.Koutsoukos@vanderbilt.edu).

Waseem Abbas is with the Department of Systems Engineering, University of Texas at Dallas, Richardson, TX 75080 USA (e-mail: waseem.abbas@utdallas.edu).

Mudassir Shabbir is with the Department of Computer Science at Vanderbilt University, Nashville, TN 37235 USA, and also with the Department of Computer Science, Information Technology University, Lahore 54000, Pakistan (e-mail: mudassir.shabbir@vanderbilt.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TRO.2022.3178296.

Digital Object Identifier 10.1109/TRO.2022.3178296

A subset of the results appeared in the preliminary form in [1].

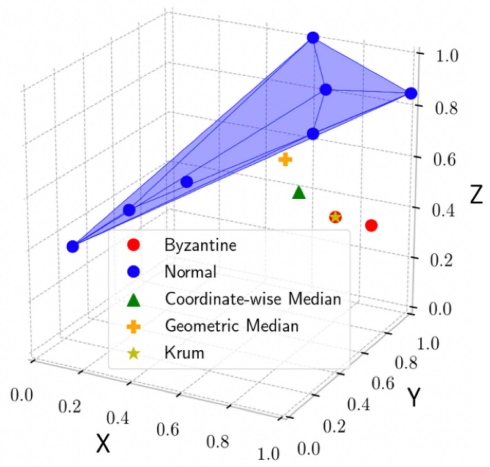


Fig. 1. Aggregating nine points including two Byzantine points using different aggregation rules: all the aggregated results fall outside of the convex hull (blue polygon) of the normal points.

the aggregation step lies inside the convex hull of the normal agents' states. When the sufficient condition is satisfied, we show that normal agents converge to the global optimum state with $O(1/i)$ convergence rate using appropriate stepsizes, where i denotes the time index.

2) We propose a centerpoint-based aggregation rule and show that it guarantees the resilient convergence of the distributed learning algorithms whenever each normal agent k in the network has $f \leq \lceil \frac{n_k}{d+1} \rceil - 1$ Byzantine neighbors.
3) We evaluate our results using the examples of target pursuit and pattern recognition in multirobot systems. We compare the proposed centerpoint-based aggregation rule with the average, CM, and GM-based rules. The simulation results show that our approach is resilient to $\lceil \frac{n_k}{d+1} \rceil - 1$ Byzantine neighbors, and the cooperation improves the average learning performance over the network than the noncooperative case, while the other approaches are not resilient in the same scenarios.

The rest of this article is organized as follows: Section II discusses the related work. Section III formulates the resilient distributed learning problem. Section IV analyzes the sufficient condition to achieve resilient convergence in distributed learning. Section V introduces the resilient aggregation rule based on the centerpoint, which satisfies the sufficient condition and further guarantees the resilient convergence in distributed learning. Section VI gives an evaluation of the results. Finally, Section VII concludes this article.

II. Related Work

Approximate Byzantine Consensus: The approximate Byzantine consensus problem initiated in [23] is widely studied in the robotics and control systems community and is very relevant to resilient distributed learning and optimization. The main objective is to ensure that all normal agents in a network satisfy the *safety* and *agreement* conditions in the presence of Byzantine agents [24], [25]. Safety condition requires normal agents to update their states such that they are always inside the convex hull of normal agents' initial states. Agreement means that eventually, all normal agents' states are very close to each other, that is, within an arbitrary $\epsilon > 0$ distance from one another. The mean subsequence reduced (MSR) algorithms [26]–[28] and the median-based algorithms [29]–[31] were proposed for the approximate Byzantine consensus over scalar states. The problem is more challenging when the state vectors are in $\mathbb{R}^d$ where $d \geq 2$. For higher-dimensional cases, the MSR technique can be applied in each dimension separately. However, this does not guarantee that the aggregated result will be in the convex hull of normal states.

For vector consensus, Tverberg partition [25], [32], [33], safe area [24], and centerpoint-based [34], [35] approaches have been proposed. The resilient vector consensus has various applications in multirobot systems for fault-tolerant rendezvous [36], formation control [37], flocking [38], secure localization [39], [40], and target pursuit [1]. Some of these works rely on coordinate-wise resilient scalar consensus, which does not necessarily achieve resilient vector consensus. Note that in the resilient consensus problem, the connectivity and robustness of the underlying network play an important role in the convergence of the iterative algorithms. In order to achieve resilient consensus, the underlying network should satisfy certain robustness conditions that in turn guarantee the redundant information needed by agents to ensure consensus in the presence of adversarial agents [27], [28]. Different from the consensus problem, in distributed learning, agents learn to converge to their target using the adaptation step in addition to the aggregation of the neighbors' states. We will show in Section IV that the normal agents converge toward the optimum state as long as the safety condition is satisfied in the aggregation step of distributed learning, regardless of the robustness of the underlying connectivity graph, which is different from the case of resilient consensus. Note that in distributed consensus, the robustness condition requires every normal agent to have sufficient many normal neighbors in order to gather enough information and diffuse its own information across the entire network and ensure the convergence point is resulted from the combination of the information from the entire network [27].

Resilient Aggregation in DML: To achieve resilient convergence in DML, one approach is to discard cooperation with possible Byzantine neighbors using the idea of trimming, similar to the MSR approach used in resilient scalar consensus. In such an approach, it is assumed that a maximum of f Byzantine agents can be present in the neighborhood of a normal agent. Algorithms are then designed for a normal agent to rank its neighbors based on some trust criteria and a normal agent discards the values from its f least trusted neighbors. Various metrics have been proposed in the literature to evaluate a agent's trustworthiness, including metrics based on the product of the weight and the loss [11], model parameters [15], gradients and their norms [41], [42], and a combination of gradient and model parameters [43]. Moreover, various majority-based aggregation rules have been proposed, similar to the median-based approach used in resilient scalar consensus. Well-known majority-based aggregation rules include the CM [12]–[14], coordinate-wise trimmed mean [12], [15], GM [16], [17], Krum and multi-Krum

[10], Bulyan and multi-Bulyan [18], [19]. Instead of ranking and filtering out the suspicious messages, these rules differ from the ranking methods in the way that they do not need the ranking step and the aggregation result directly precludes states far away from the cluster of the majority of the states by the intrinsic properties of the aggregation rules. However, studies have already reported all of the abovementioned methods are not resilient to attacks under certain conditions [20]–[22].

Resilient DML via Computation Redundancy: One can also use computation redundancy to achieve resilient convergence in DML, which typically involves coding theory and algorithmic redundancy [44]–[46]. An example of such a framework is DRACO [44] in which the parameter server uses redundant gradients received from agents to eliminate the effects of adversarial updates. Another algorithm proposed recently is the RSA [47] that introduces an ℓ_p-norm regularization term into the objective function for the resilience purpose. It eliminates the effect caused by the magnitudes of malicious messages sent by the Byzantine agents, as a result, only the number of Byzantine agents, but not the magnitude, influence the model update, making the algorithm robust to large outliers.

III. Problem Formulation

In this section, we describe the problem of distributed learning in networks in an adversarial setting. First, we introduce the following notation:

$(\cdot)^\top$ transpose of a matrix;
$[n]$ $\{1, 2, \ldots, n\}$;
$|\cdot|$ cardinality of a set;
$\|\cdot\|$ ℓ_2-norm of a vector;
$\mathbb{E}_\xi[\cdot]$ the expected value of a random variable ξ; if the context is clear, $\mathbb{E}[\cdot]$ is used.

A. Distributed Learning

Consider a connected network of n agents[1] modeled by an *undirected graph* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ represents agents and $\mathcal{E}$ represents interactions between agents. A bidirectional edge $(l, k) \in \mathcal{E}$ means that agents k and l can exchange information with each other. Since each agent also has its own information, we have $(k, k) \in \mathcal{E}, \forall k \in \mathcal{V}$. The *neighborhood* of k is the set $\mathcal{N}_k = \{l \in \mathcal{V} | (l, k) \in \mathcal{E}\}$. Each agent k has data $\{(x_k^i, y_k^i)\}_{i \in \mathcal{S}_k}$ sampled randomly from the distribution generated by the random variable ξ_k, where $x_k^i \in \mathbb{R}^d$, $y_k^i \in \mathbb{R}$, and $\mathcal{S}_k$ are the sample set. We consider a convex *prediction function* (model) $\varphi_k(x_k^i) = \theta_k^\top x_k^i$, where $\theta_k \in \mathbb{R}^d$ is the model parameter (or state). We use $\ell_k(\cdot)$ to denote a convex *loss function* associated with the prediction function for agent k, and $f_k(\cdot)$ to denote the convex (expected) *risk function* $f_k(\theta_k) = \mathbb{E}[\ell_k(\theta_k; \xi_k)]$.

The *objective* of the network of n agents is to estimate the parameter vector θ^* in a distributed and cooperative manner, that minimizes a global cost function of the following form:

$$\min_\theta \left\{ J(\theta) \triangleq \frac{1}{n} \sum_{k=1}^{n} f_k(\theta) \right\}. \tag{1}$$

[1]We use terms *agent* and *robot* interchangeably.

Stochastic gradient descent (SGD) can be used to optimize the global cost function (1) given the stochastic gradient of $J(\theta)$. Since such a value is not available, we consider a distributed solution for each agent, known as *cooperative SGD*, which takes the following two steps in synchronized rounds of communication between agents [2]

$$\hat{\theta}_{k,i} = \theta_{k,i-1} - \alpha_{k,i-1} \nabla \ell_k \left(\theta_{k,i-1}; \xi_k^{i-1}\right), \quad \text{(SGD)} \tag{2}$$

$$\theta_{k,i} = \text{Aggr}\left(\left\{\hat{\theta}_{l,i} : l \in \mathcal{N}_k\right\}\right). \quad \text{(Aggregation)} \tag{3}$$

In cooperative SGD, at each iteration i, agent k minimizes the individual risk using SGD given local data, followed by an aggregation step that aggregates neighboring estimates. Here, $\alpha_{k,i}$ is the stepsize, $\nabla \ell_k(\theta_{k,i-1}; \xi_k^{i-1})$ is the gradient using the instantaneous realization ξ_k^{i-1} of the random variable ξ_k, $\mathcal{N}_k$ is the neighborhood set of agent k, and $\text{Aggr}(\cdot)$ denotes an aggregation function. An example of aggregation functions is the convex combination of the neighbors' states, i.e.,

$$\text{Aggr}\left(\left\{\hat{\theta}_{l,i} : l \in \mathcal{N}_k\right\}\right) \triangleq \sum_{l \in \mathcal{N}_k} a_{lk}(i) \hat{\theta}_{l,i}$$

where $a_{lk}(i)$ denotes the weight assigned by agent k to l at iteration i, and satisfies the following:

$$a_{lk}(i) \geq 0, \quad \sum_{l \in \mathcal{N}_k} a_{lk}(i) = 1, \quad a_{lk}(i) = 0 \text{ if } l \notin \mathcal{N}_k. \tag{4}$$

B. Byzantine Attacks and Resilient Distributed Learning

In distributed learning, the issue of resilience against Byzantine agents has received much attention recently [12], [16], [17], [48], [49]. Since Byzantine agents can send incorrect information (state values) to their neighbors, the aggregation step is susceptible to cyber-attacks. In particular, normal agents communicating with Byzantine neighbors and updating their states using the Byzantine messages in the aggregation step may converge to a point desired by the attacker [11].

We assume two types of agents in the network, normal and Byzantine. *Normal* agents are the ones that interact with their neighbors synchronously and always update their estimates according to the prescribed update rule. *Byzantine* agents are the ones that can change their states arbitrarily and do not follow the prescribed update rule. Moreover, a Byzantine agent can transmit different values to its different neighbors. Further, we assume that the identities of normal and Byzantine agents are not changing. Since Byzantine agents that stop sending messages can be easily identified in a synchronous network, we assume Byzantine agents always send messages during communication. For a normal agent k, all agents in its neighborhood are indistinguishable, that is, k cannot identify which of its neighbors are Byzantine. Further, we use the following notation:

$\mathcal{N}$ set of normal agents;
$\mathcal{F}$ set of Byzantine agents($|\mathcal{N}| + |\mathcal{F}| = n$);
$\mathcal{N}_k$ set of neighbors of agent k;
$\mathcal{N}_k^+$ set of normal neighbors of agent k;
$\mathcal{N}_k^-$ set of Byzantine neighbors of agent k, such that $|\mathcal{N}_k^+| + |\mathcal{N}_k^-| = |\mathcal{N}_k|$;

f upper bound on the number of Byzantine neighbors of a normal agent, i.e., $|\mathcal{N}_k^-| \leq f$.

In the presence of Byzantine agents, the objective of the normal agents should be rewritten as follows:

$$\min_{\theta} \left\{ F(\theta) \triangleq \frac{1}{|\mathcal{N}|} \sum_{k \in \mathcal{N}} f_k(\theta) \right\}. \tag{5}$$

We make the following assumption about the global objective function in (5):

Assumption 1: (Strong convexity) The global objective function F is strongly convex in that there exists a constant $c > 0$ such that

$$F(y) \geq F(x) + \nabla F(x)^\top (y - x) + \frac{c}{2}\|y - x\|^2, \forall x, y.$$

Hence, F has a unique minimizer, denoted as θ^* with $F^* \triangleq F(\theta^*)$. We also assume that all agents share the same minimizer θ^*, which is a common scenario in distributed learning problems, where data of different agents arises from the same distribution, distributed inference applications, where distributions depend on a common parameter vector to be optimized, and single task networks where agents work to attain a common objective such as tracking a target [2], [50]–[53].

This article aims to address the problem of resilient distributed learning in the presence of Byzantine agents. The goal is to ensure that all the normal agents in the network using the cooperative SGD algorithm to optimize (5) achieve *resilient convergence*, formally stated below.

Definition 1: (Resilient Convergence) The network is said to achieve resilient convergence if

$$\lim_{i \to \infty} \mathbb{E}\left[\|\theta_{k,i} - \theta^*\|^2\right] = 0, \forall k \in \mathcal{N} \tag{6}$$

thereby ensuring that all normal agents converge to the globally optimum state in expectation. Here, θ^* is the minimizer of (5).

IV. Resilient Distributed Learning

In this section, we propose a sufficient condition to guarantee the resilient convergence of the cooperative SGD algorithms. We also discuss the possible outcome when this condition is not satisfied. Later in Section V, we propose an aggregation rule that satisfies the sufficient condition, which further guarantees the resilient convergence of the cooperative SGD.

A. Sufficient Condition and Convergence Analysis

Sufficient Condition. At each iteration $i \in \mathbb{N}$ and for every normal agent $k \in \mathcal{N}$, the outcome of the aggregation step $\theta_{k,i}$ is a convex combination of the estimates of the normal neighbors of k, i.e.,

$$\theta_{k,i} = \sum_{l \in \mathcal{N}_k^+} a_{lk}(i)\hat{\theta}_{l,i}, \forall i \in \mathbb{N}, k \in \mathcal{N}$$

$$\text{s.t. } a_{lk}(i) \geq 0, \forall l \in \mathcal{N}_k^+, \text{ and } \sum_{l \in \mathcal{N}_k^+} a_{lk}(i) = 1. \tag{7}$$

In other words, at each iteration i, a normal agent k aggregates its neighbors' estimates such that the output of the aggregation $\theta_{k,i}$ is in the *convex hull of normal neighbors' state estimates* regardless of the estimates from Byzantine neighbors[2].

Next, we prove the resilient convergence when the sufficient condition is satisfied (see Theorem 1). To facilitate the analysis, we list the following assumptions and lemma.

Assumption 2: (Lipschitz-continuous gradients). The global objective function F is continuously differentiable and ∇F is Lipschitz continuous with Lipschitz constant $L > 0$, i.e.,

$$\|\nabla F(x) - \nabla F(y)\| \leq L\|x - y\|, \forall x, y.$$

Assumption 3: (First and second moment limits). The objective function F and the sequence of $\theta_{k,i}$ for $k \in \mathcal{N}$ and $i \in \mathbb{N}$, obtained by implementing the cooperative SGD algorithm satisfy the following:

1) $F(\theta_{k,i}) \leq F_{\inf}$ for some scalar $F_{\inf}$.
2) There exist scalars $\mu_G \geq \mu > 0$ such that
$$\nabla F\left(\theta_{k,i}\right)^\top \mathbb{E}_{\xi_k^i}\left[\nabla \ell_k(\theta_{k,i}; \xi_k^i)\right] \geq \mu \|\nabla F\left(\theta_{k,i}\right)\|^2 \text{ and}$$
$$\|\mathbb{E}_{\xi_k^i}\left[\nabla \ell_k\left(\theta_{k,i}; \xi_k^i\right)\right]\| \leq \mu_g \|\nabla F\left(\theta_{k,i}\right)\|$$
$$, \forall k \in \mathcal{N} \text{ and } i \in \mathbb{N}.$$
3) There exist scalars $M_k \geq 0$ and $V_k \geq 0$ such that
$$\mathbb{E}_{\xi_k^i}\left[\|\nabla \ell_k(\theta_{k,i}; \xi_k^i)\|^2\right] - \|\mathbb{E}_{\xi_k^i}\left[\nabla \ell_k\left(\theta_{k,i}; \xi_k^i\right)\right]\|^2$$
$$\leq M_k + V_k \|\nabla F(\theta_{k,i})\|^2, \forall k \in \mathcal{N} \text{ and } i \in \mathbb{N}.$$

Note that Assumption 3 is based on the preliminary assumption that agents share the same minimizer θ^*.

Lemma 1: [3] Under Assumptions 1–3 and stepsize $0 < \alpha_{k,i} \leq \frac{\mu}{LG_k}$, where $G_k \triangleq V_k + \mu_g^2$, for all $k \in \mathcal{N}$, $i \in \mathbb{N}$, the iterates of SGD (step *(2)*) satisfy the following inequalities:

$$\mathbb{E}\left[F\left(\hat{\theta}_{k,i+1}\right) - F^*\right] \leq (1 - \alpha_{k,i}c\mu)\,\mathbb{E}\left[F\left(\theta_{k,i}\right) - F^*\right] + \frac{1}{2}\alpha_{k,i}^2 L M_k.$$

Denote $\Delta_{k,i} \triangleq \mathbb{E}[F(\theta_{k,i}) - F(\theta^*)]$ as the expected optimality gap. Using the cooperative SGD algorithm satisfying the sufficient condition, $\Delta_{k,i}$ can be bounded as given in Theorem 1, which guarantees its resilient convergence as given in Proposition 1.

Theorem 1: If the sufficient condition (7) and Assumptions 1–3 are satisfied, and all normal agents implement the cooperative SGD algorithm with the same diminishing stepsize sequence $\alpha_{k,i} = \alpha_i$ given by

$$\alpha_i = \frac{\beta}{\gamma + i} \text{ for some } \beta > \frac{1}{c\mu} \text{ and } \gamma > 0 \text{ s.t. } 0 < \alpha_1 \leq \frac{\mu}{L\bar{G}_k} \tag{8}$$

where $\bar{G}_k \triangleq \max_{k \in \mathcal{N}} G_k$, then for all $i \in \mathbb{N}, k \in \mathcal{N}$, the expected optimality gap satisfies

$$\Delta_{k,i} \leq \frac{\nu}{\gamma + i} \tag{9}$$

[2]The convex hull of a set of points $S = \{p_1, p_2, \ldots, p_n\}$ in $\mathbb{R}^d$ is the smallest convex set containing S. Any point p inside the convex hull of S has the property that $p = \sum_{k=1}^n \lambda_k p_k$, where $0 \leq \lambda_k \leq 1$ and $\sum_{k=1}^n \lambda_k = 1$. And no point outside of the convex hull has such representation.

[3]Proof is given in the Appendix.

where $\nu = \max\left\{\frac{\beta^2 L}{2(\beta c\mu - 1)}\max_{l\in\mathcal{N}} M_l, (\gamma+1)\Delta_{k,1}\right\}$.

Proof: Given (7) and the convexity of F, using Jensen's inequality, we have

$$F(\theta_{k,i}) \leq \sum_{l\in\mathcal{N}_k^+} a_{lk}(i) F\left(\hat{\theta}_{l,i}\right). \tag{10}$$

Given (8), $\alpha_{k,i} = \alpha_i \leq \alpha_1 \leq \frac{\mu}{L G_k} \leq \frac{\mu}{L G_k}$, for all $k \in \mathcal{N}$. Thus, following Lemma 1, it holds that

$$\Delta_{k,i+1} \leq \sum_{l\in\mathcal{N}_k} a_{lk}(i)\left[(1-\alpha_i c\mu)\Delta_{l,i} + \frac{1}{2}\alpha_i^2 L M_l\right]. \tag{11}$$

The following proof is based on the convergence proof of SGD [54, Th. 4.7]. When $i = 1$, (9) holds given the definition of ν. We now prove (9) by induction. Assume (9) holds for some $i \geq 1$, then it follows from (11) that

$$\begin{aligned}
\Delta_{k,i+1} &\leq \sum_{l\in\mathcal{N}_k} a_{lk}(i)\left[\left(1-\frac{\beta c\mu}{\hat{i}}\right)\frac{\nu}{\hat{i}} + \frac{\beta^2 L M_l}{2\hat{i}^2}\right] \\
&= \sum_{l\in\mathcal{N}_k} a_{lk}(i)\left[\left(\frac{\hat{i}-\beta c\mu}{\hat{i}^2}\right)\nu + \frac{\beta^2 L M_l}{2\hat{i}^2}\right] \\
&= \frac{\hat{i}-1}{\hat{i}^2}\nu - \sum_{l\in\mathcal{N}_k} a_{lk}(i)\underbrace{\left[\frac{\beta c\mu - 1}{\hat{i}^2}\nu - \frac{\beta^2 L M_l}{2\hat{i}^2}\right]}_{\text{nonnegative by the definition of } \nu} \\
&\leq \frac{\nu}{\hat{i}+1}
\end{aligned}$$

where $\hat{i} :\triangleq \gamma + i$, and the last inequality follows since $\hat{i}^2 \geq (\hat{i}+1)(\hat{i}-1)$, which completes our proof. □

Remark 1: It immediately follows from Theorem 1 that normal agents achieve $O(1/i)$ convergence rate since $\mathbb{E}[F(\theta_{k,i}) - F(\theta^*)] \leq \frac{\nu}{\gamma+i}$ with constants $\nu > 0$ and $\gamma > 0$.

Proposition 1: If the sufficient condition (7) and Assumptions 1–3 are satisfied, and all normal agents implement the cooperative SGD with the same diminishing stepsize sequence defined in (8), then the network achieves resilient convergence as defined in Definition 1.

Proof: Given the expected optimality gap (9) obtained in Theorem 1, and the fact that $\lim_{i\to\infty}\frac{\nu}{\gamma+i} = 0$ with constants $\nu > 0$ and $\gamma > 0$, it follows that

$$\lim_{i\to\infty}\mathbb{E}\left[F(\theta_{k,i}) - F(\theta^*)\right] = 0, \forall k \in \mathcal{N}.$$

Using the strong convexity of F, it yields that

$$\lim_{i\to\infty}\mathbb{E}\left[\|\theta_{k,i} - \theta^*\|^2\right] \leq \frac{c}{2}\lim_{i\to\infty}\mathbb{E}\left[F(\theta_{k,i}) - F(\theta^*)\right] = 0$$

$\forall k \in \mathcal{N}$, which completes the proof. □

The above discussion establishes that if the aggregation result is a convex combination of the normal neighbors' states, then the resilient convergence is guaranteed. Now, we consider a scenario in which the condition (7) is not satisfied and $\theta_{k,i}$ in an aggregation step of a normal node k lies outside the convex hull of $\hat{\theta}_{j,i}$, where $j \in \mathcal{N}_k^+$. Then, it is possible that after aggregation the distance from $\theta_{k,i}$ to θ^* is larger than the distance of any of the normal neighbors' states to θ^*. This means the aggregation step indeed makes the state of the normal agent k deviate from θ^*. As the number of iterations increases, the deviation from the target state might also increase and normal agents might converge to a wrong state. We further demonstrate this in Section VI, where the aggregation rules, such as average, CM, and GM, do not satisfy the sufficient condition, and normal agents fail to achieve resilient convergence using such rules.

B. Advantages of Cooperation

In the previous section, we proposed a sufficient condition for achieving resilient distributed learning and analyzed its convergence. However, a particular case, where agents do not cooperate with each other, i.e., the *noncooperative* case, also achieves this goal. However, the comparison between cooperative and noncooperative distributed learning and adaptation has been studied in the literature, which reveals the advantages of using cooperation in distributed learning, especially for lifelong adaptation [2], [55], [56], [53, Chap. 12]. In the following, we briefly review the results of such studies and discuss the advantages by deploying the cooperative setting.

Note that we used a time-dependent diminishing stepsize in the previous section for convergence analysis. However, constant stepsizes are often used for lifelong learning and adaption over time, in various applications such as distributed sensing, biologically systems, and target localization, where diminishing stepsizes fail to work. When using constant stepsizes, for sufficiently small stepsizes, the network converges to a point close to the optimum point but with a bounded distance. The squared Euclidean distance between the convergence point and the optimum point is referred to as the steady-state mean-square-deviation (MSD), which can be used to measure the learning performance [2].

It is studied in [56] that for sufficiently small stepsize, cooperation improves the steady-state MSD in a distributed learning network. We illustrate the case when the data is related via a linear model, i.e., $y_k^i = \theta_{k,i}^\top x_k^i + v_k^i$, where v_k^i is a noise term with $\mathbb{E}[{v_k^i}^\top v_k^i] = \sigma_{v,k}^2$. For sufficiently small uniform constant stepsize α for each agent, the network learning performance of noncooperative SGD is defined as the averaged steady-state MSD among agents and can be approximated by

$$\underset{\text{ncop, net}}{\text{MSD}} \triangleq \lim_{i\to\infty}\frac{1}{N}\sum_{k=1}^N \mathbb{E}\|\tilde{\theta}_{k,i}\|^2 \approx \frac{\alpha d}{2}\cdot\left(\frac{1}{N}\sum_{k=1}^N \sigma_{v,k}^2\right) \tag{12}$$

where $\tilde{\theta}_{k,i} \triangleq \theta^* - \theta_{k,i}$, and d is the dimension of the states. And for cooperative SGD, the network steady-state MSD is given by

$$\text{MSD}_{\text{coop,net}} \approx \frac{\alpha d}{2}\cdot\left(\sum_{k\in\mathcal{N}} p_k^2 \sigma_{v,k}^2\right) \tag{13}$$

where $A = \{a_{lk}, k = 1, 2, \ldots, N\}$ is the $N \times N$ left-stochastic combination matrix at convergence and $p = \{p_k, k =$

$1, 2, \ldots, N\}$ denotes the right eigenvector of A that is associated with the eigenvalue at one and satisfies $Ap = p, p^\top \mathbb{1} = 1, 0 < p_k < 1$.

The aggregated result can be expressed by a weighted sum of all the normal agents' estimates, i.e., $\theta_{k,i} = \sum_{l \in \mathcal{N}_k} a_{lk}(i)\hat{\theta}_{k,i}$, where $0 \leq a_{lk}(i) \leq 1$ and $\sum_{l \in \mathcal{N}_k} a_{lk}(i) = 1$. Assume the noise variance is uniform across all normal agents, i.e., $\sigma^2_{v,k} = \sigma^2_v, (k = 1, 2, \ldots, N)$, then we observe that the steady-state MSD performance of the cooperative SGD is better than the noncooperative SGD by (12) and (13) as

$$\begin{aligned} &\text{MSD}_{\text{coop,net}} - \text{MSD}_{\text{ncop,net}} \\ &= \frac{\alpha d}{2} \cdot \left(\sum_{k \in \mathcal{N}^+} p_k^2 \sigma^2_{v,k} - \frac{1}{|\mathcal{N}^+|} \sum_{k \in \mathcal{N}^+} \sigma^2_{v,k} \right) \\ &= \frac{\alpha d}{2} \cdot \left(\sum_{k \in \mathcal{N}^+} p_k^2 - 1 \right) \cdot \sigma_v^2 \leq 0 \end{aligned}$$

where $\sum_{k \in \mathcal{N}_k^+} p_k^2 \leq 1$ given $\sum_{k \in \mathcal{N}_k} p_k = 1$ and $p_k > 0$.

Based on the above discussion, cooperation in general leads to better learning performance measured by steady-state MSD when constant stepsizes are used for lifelong learning and adaptation, for instance, when the objective to be optimized by the network is time-varying. This is further demonstrated in the evaluation section. Moreover, cooperation among agents, for instance, in a diffusion network, improves the network performance by reducing the *excess risk* compared to the noncooperative setup [52],[53, Sec. 12.4]. However, cooperation could be detrimental in the presence of Byzantine agents. This article presents a solution that discards Byzantine agents' effects in a cooperative setting, thus allowing to utilize the benefits of cooperation.

V. Resilient Aggregation

The sufficient condition in Section IV requires that in the aggregation step, a normal agent computes a point that lies in the convex hull of points corresponding to *normal* neighbors' states. Computing such a point is a challenging task as a normal agent cannot distinguish between its normal and Byzantine neighbors. Our goal in this section will be to propose an efficient way to compute such a point in the aggregation step. For this, we will utilize the notion of *safe region* as defined below.

A. The Safe Region

Let S denote a set of n_k points, and S' be a subset of S containing exactly $(n_k - f)$ points. There are $\binom{n_k}{n_k - f}$ possibilities of S' and one such possibility corresponds to the set of normal points. Let $\mathcal{S}$ be the family of all $\binom{n_k}{n_k - f}$ possibilities of S'. If one could somehow show that the intersection of convex hulls of members of $\mathcal{S}$ is nonempty, then every point in this intersection is guaranteed to lie inside the convex hull of normal points. We call this intersection set a *safe region*.

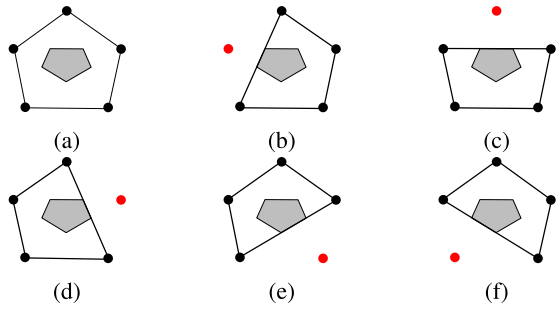


Fig. 2. Illustration of $\text{Safe}_1(S)$ (shaded region) for a set of five points in (a). In (b)–(f), black nodes are normal, red nodes are Byzantine, and the area spanned by thick lines is the convex hull of the normal nodes.

Definition 2: (Safe Region) For a set S of n_k points in $\mathbb{R}^d$, of which any f points can be Byzantine, the safe region of S is

$$\text{Safe}_f(S) = \bigcap_{S' \subset S, |S'| = n_k - f} \text{Conv}(S')$$

where $\text{Conv}(S')$ denotes the convex hull of S'.

It is immediately implied from the above definition that $\text{Safe}_f(S)$, if it exists, is always in the convex hull of the $(n_k - f)$ normal points, regardless of the selection of the f Byzantine points, as illustrated by the example in Fig. 2. There are $n_k = 5$ points and $f = 1$, which means there are five possibilities to choose a point corresponding to a Byzantine agent. The gray shaded area is the safe region $\text{Safe}_1(S)$. We note that the safe region always lies in the convex hull of four normal points regardless of the selection of the Byzantine point, as illustrated in Fig. 2(b)–(f).

The existence of the safe region depends on the number of Byzantine agents (points) f. For example, if $f > n_k/2$, then a safe region may not exist even in dimension $d = 1$, because two potential sets of normal points, (leftmost, and rightmost intervals of $n_k - f$ points) are nonoverlapping, and the intersection will be empty. For dimension $d = 2$, if there are $n_k/3$ Byzantine points, then all possibilities for $2n_k/3$ normal points might not have a common point. In other words, no matter which point we choose for aggregation, there is always a chance that it lies outside the convex hull of normal points in $\mathbb{R}^2$. Thus, the number of allowed Byzantine points can not be more than $n/3$ in a plane. The condition of the existence of a nonempty safe region has been studied in [24, Lemma 3.6, 3.10], which is given in the following.

Lemma 2: For a set S of n_k points in $\mathbb{R}^d$, of which any f points could be Byzantine, if $f < \frac{n_k}{d+1}$, then $\text{Safe}_f(S)$ is necessarily nonempty; and if $f \geq \frac{n_k}{d+1}$, then $\text{Safe}_f(S)$ might be empty.

Based on Lemma 2, the maximum number of Byzantine agents the system is resilient to is $f = \lceil \frac{n_k}{d+1} \rceil - 1$ and the safe region is nonempty in this case. For computing, a point in the safe region when $f = \lceil \frac{n_k}{d+1} \rceil - 1$, we use the notion of *centerpoint*, which we explain next.

B. Centerpoint-Based Resilient Vector Consensus

In the following, we define the notion of a *centerpoint*, and show that every centerpoint lies inside $\text{Safe}_f(S)$ for $f = \lceil \frac{n_k}{d+1} \rceil - 1$.

Definition 3: (Centerpoint) Given a set S of n_k points in $\mathbb{R}^d$ in general positions,[4] where $n_k \geq d+1$, a centerpoint p is a point, not necessarily from S, such that any closed half-space[5] of $\mathbb{R}^d$ that contains p also contains at least $\lceil \frac{n_k}{d+1} \rceil$ points from S.

Intuitively, a centerpoint lies in the "center region" of the set of points, in the sense that there are enough points of S on each side of a centerpoint. A centerpoint extends the notion of median to higher dimensions, and is an active topic of study in discrete geometry [34], [35], [57]–[59]. Note that centerpoint is not unique, in fact, there can be infinitely many centerpoints. The set of all centerpoints constitutes the convex safe region. We have studied the connection between the centerpoint and the safe region in [34], [35]. In particular, we have the following result.

Lemma 3: Let S be a set of n_k points in $\mathbb{R}^d$, $\mathcal{C}(S)$ be the corresponding centerpoint region (set of all centerpoints) and $f = \lceil \frac{n_k}{d+1} \rceil - 1$, then

$$\text{Safe}_f(S) \equiv \mathcal{C}(S).$$

An immediate consequence of Lemmas 2 and 3 is that for a set S of n_k points in $\mathbb{R}^d$, of which any $f = \lceil \frac{n_k}{d+1} \rceil - 1$ points could be Byzantine, a centerpoint of S is always inside the convex hull of the $(n_k - f)$ normal points, regardless of the selection of the f Byzantine points.

Now that the existence of a point in the safe region for optimal number of Byzantine neighbors is guaranteed, one has to actually find such a point to aggregate to. It is easy to compute a centerpoint in lower dimensions. In two dimensions, the time complexity for computing a centerpoint is $O(n_k)$ using a prune and search algorithm in [60]. The algorithm iteratively removes a fraction of points from a given point set while ensuring that the centerpoint of the remaining points is also a centerpoint of the original point set. When the number of points becomes smaller than a constant, the algorithm uses a brute force method to compute a centerpoint [60]. In three dimensions, we can use Chan's algorithm in [61] to find a centerpoint in $O(n_k^2)$ time. Chan[61] basically provides a randomized algorithm to some geometric linear program in $O(n_k^2)$ and uses this framework to find a geometric Tukey median of a given point set, which is guaranteed to be a centerpoint also. We have given an overview of these methods in our previous work [34], [35]. We note that $d \leq 3$ is the case in many practical applications in robotics. For higher dimensions, i.e., $d > 3$, the time bound to compute a centerpoint is $O(n_k^{d-1})$ [61], which is impractical for very large d. However, in such cases, algorithms exist to compute an approximate centerpoint [62]. These approximations degrade the optimal bound on the number of Byzantine agents. For instance, given a set of n_k points, of which at most $(\frac{n_k}{d^{r/r-1}})$ are Byzantine and the remaining are normal points, we can compute a point that is in the convex hull of normal points in time $O(n_k^{c \log d}(rd)^d)$, where r is any integer greater than 1, and c is some positive constant. By increasing r, the quality of approximation, and hence the bound on the number of Byzantine agents improves and approaches $\frac{n}{d}$; however, this also leads to an increase in the time complexity. Moreover, the method proposed in [63] generates approximate centerpoint in linear time complexity for any dimension. However, this will reduce the upper bound on Byzantine tolerance from $\lceil \frac{n_k}{d+1} \rceil - 1$ to $\lceil \frac{n_k}{4(d+1)^3} \rceil - 1$.

Proposition 2: If $f \leq \lceil \frac{|\mathcal{N}_k|}{d+1} \rceil - 1$, Assumptions 1–3 are satisfied, and all normal agents implement the cooperative SGD with the same diminishing stepsize sequence defined in (8) using the centerpoint-based aggregation rule in the aggregation step, i.e.,

$$Aggr\left(\left\{\hat{\theta}_{l,i} : l \in \mathcal{N}_k\right\}\right) \triangleq \mathcal{C}\left(\left\{\hat{\theta}_{l,i} : l \in \mathcal{N}_k\right\}\right)$$

then the network achieves resilient convergence as defined in Definition 1.

Proof: Since $f \leq \lceil \frac{|\mathcal{N}_k|}{d+1} \rceil - 1$, given Lemma 3 and Definition 2, it follows that

$$Aggr\left(\left\{\hat{\theta}_{l,i} : l \in \mathcal{N}_k\right\}\right) \in \text{Conv}(\left\{\hat{\theta}_{j,i} : j \in \mathcal{N}_k^+\right\})$$

which satisfies the sufficient condition in (7). Then, the resilient convergence is guaranteed given the results provided in Proposition 1. □

VI. Evaluations

In this section, we evaluate the proposed centerpoint-based aggregation rule for the cooperative SGD algorithm using target pursuit and pattern recognition as case studies.[6] We compare it with other commonly used aggregation rules including the *average* ($a_{lk} = \frac{1}{|\mathcal{N}_k|}$ for $l \in \mathcal{N}_k$), CM, and GM, as well as the non-cooperative SGD (non-coop SGD). We define the CM and GM below.

1) CM: Let $\text{med}(\cdot)$ be the one-dimensional median, then the CM $\text{Median}(\cdot)$ of vectors $\{x_k \in \mathbb{R}^d, k \in [n]\}$ is defined to be $x^{\text{Med}} \triangleq \text{Median}\{x_k : k \in [n]\}$ with the jth coordinate to be $(x^{\text{Med}})^j \triangleq \text{med}\{x_k^j : k \in [n]\}$ for each $j \in [d]$.
2) GM: The geometric median $\text{GM}(\cdot)$ of vectors $\{x_k \in \mathbb{R}^d, k \in [n]\}$ is defined to be $\text{GM}\{x_k : k \in [n]\} \triangleq \arg\min_{x \in \mathbb{R}^d} \sum_{k=1}^n \|x - x_k\|$.

We show in multiple cases that the cooperative SGD algorithm using centerpoint-based aggregation always outperforms the noncooperative SGD in achieving a better average learning performance over the network at convergence, with or without the presence of Byzantine agents. However, the other rules either fail to converge to θ^* or exhibit a worse learning performance than the noncooperative SGD, showing that such cooperation could be harmful to the overall network's performance.

[4] A set of points in $\mathbb{R}^d$ is said to be in *general positions* if no hyperplane of dimension $d-1$ or less contains more than d points.

[5] Recall that closed half-space in $\mathbb{R}^d$ is a set of the form $\{x \in \mathbb{R}^d : a^T x \geq b\}$ for some $a \in \mathbb{R}^d \setminus \{0\}$.

[6] Simulation code can be found at [Online]. Available: https://github.com/JianiLi/resilient_distributed_learning_centerpoint.

A. Target Pursuit

In this example, we consider a mobile adaptive network [8] of n agents that move collectively in pursuit of a target located at $\theta^* \in \mathbb{R}^d$ that can be either static or time-varying.

A.1. Background

Suppose the location of agent k at time i is denoted by $x_{k,i} \in \mathbb{R}^d$. The distance $d_k^o(i) \in \mathbb{R}$ between agent k and the target at time i can be expressed as

$$d_k^o(i) = {u_{k,i}^o}^\top (\theta^* - x_{k,i})$$

where $u_{k,i}^o \in \mathbb{R}^d$ denotes the unit direction vector pointing from $x_{k,i}$ to θ^*. Suppose agents have only noisy observations $\{d_k(i), u_{k,i}\}$ of the distance and the unit direction vector, i.e.,

$$d_k(i) = d_k^o(i) + \eta_k^d(i), \quad u_{k,i} = u_{k,i}^o + \eta_{k,i}^u$$

where $\eta_{k,i}^u \in \mathbb{R}^d$ and $\eta_k^d(i) \in \mathbb{R}$ denote noise terms. Let $\eta_k(i) = -{\eta_{k,i}^u}^\top (\theta^* - x_{k,i}) + \eta_k^d(i)$, $\hat{d}_k(i) = d_k(i) + u_{k,i}^\top x_{k,i}$, we have

$$\hat{d}_k(i) = u_{k,i}^\top \theta^* + \eta_k(i).$$

To optimize θ^*, consider

$$\min_\theta \left\{ F_{loc}(\theta) \triangleq \frac{1}{|\mathcal{N}|} \sum_{k \in \mathcal{N}} \mathbb{E} \|\hat{d}_k(i) - u_{k,i}^\top \theta\|^2 \right\}. \tag{14}$$

At each iteration i, agent k knows its location $x_{k,i} \in \mathbb{R}^d$ and velocity $v_{k,i} \in \mathbb{R}^d$, and it receives its neighbors' location $x_{l,i}$ for $l \in \mathcal{N}_k$. It then updates velocity according to the following update rule [8]:

$$v_{k,i+1} = \lambda \cdot h(\theta_{k,i} - x_{k,i}) + \beta v_{k,i}^g \tag{15}$$

where $\theta_{k,i}$ is the estimate of the target location by k at time i, $v_{k,i}^g$ is the velocity of the center of mass of the network, λ, β are nonnegative parameters, and

$$h(\theta_{k,i} - x_{k,i}) = \begin{cases} \theta_{k,i} - x_{k,i}, & \text{if } \|\theta_{k,i} - x_{k,i}\| \le s \\ s \cdot \frac{\theta_{k,i} - x_{k,i}}{\|\theta_{k,i} - x_{k,i}\|}, & \text{otherwise} \end{cases}$$

for some positive scaling factor s used to bound the speed in pursuing the target.

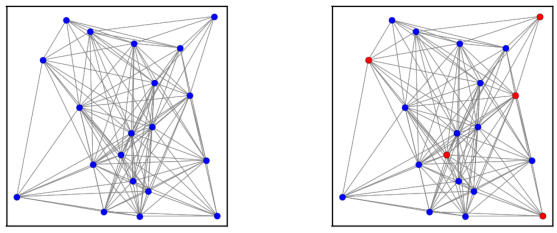
The first term in (15) relates to the objective of having the network move toward the unknown target, and the other term suggests that agents should adjust their velocities to be consistent with the average displacement vector in the neighborhood. Agents then update their location according to

$$x_{k,i+1} = x_{k,i} + \Delta t \cdot v_{k,i+1}$$

where Δt represents the time step.

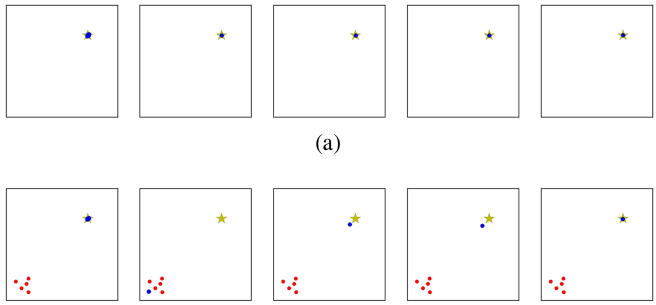
To obtain the velocity, agents need to know the estimate of the target location $\theta_{k,i}$ optimized by (14), and the velocity of the center of mass $v_{k,i}^g$, which can be optimized by

$$\min_{v^g} \left\{ F_{\text{vel}}(v^g) \triangleq \frac{1}{|\mathcal{N}|} \sum_{k \in \mathcal{N}} \mathbb{E} \|v_{k,i} - v^g\|^2 \right\}. \tag{16}$$



(a) (b)

Fig. 3. Network connectivity (blue nodes: normal agents, red nodes: Byzantine agents). (a) No attack. (b) With 5 Byzantine agents.



(a) (b)

Fig. 4. Mobile network's final deployment for static target. From left to right: noncooperative SGD, cooperative SGD with average/CM/GM/centerpoint-based aggregation. (a) No attack. (b) With 5 Byzantine agents.

A.2. Static Target

In our simulation, we consider a network of $n = 20$ agents with $d = 2$. Fig. 3 shows the initial deployment of agents with and without Byzantine attacks and their connectivity network. Agents are located in $[0, 1] \times [0, 1]$ region initially and agents connected with links are neighbors. The average neighborhood size is $\sum_{k=1}^n |\mathcal{N}_k|/n \approx 13.9$ and the underlying connectivity topology does not change throughout the simulation. The regression vector $u_{k,i}$ has uniform covariance matrix $R_{u,k} = \sigma_{u,k}^2 I_2$, $\sigma_{u,k}^2 \in [0, 1.0]$, where I_2 is the identity matrix of size 2. The noise variance of distance $\sigma_{d,k}^2 \in [1.0, 2.0], \forall k \in \mathcal{N}$. The time-varying stepsizes for updating location and velocity estimates are both $\alpha_{k,i} = \frac{2}{i+10}$ for $k \in \mathcal{N}$. Further, $\lambda = 0.5$, $\beta = 0.1$, $s = 1$, and $\Delta t = 0.2s$. The target location is denoted by $\theta^* = (5, 5)$. In the case of attack, we randomly select five agents as the Byzantine agents. For any normal agent $k \in \mathcal{N}$, it is guaranteed that the number of its Byzantine neighbors is upper bounded by $\lceil \frac{|\mathcal{N}_k|}{3} \rceil - 1$. Thus, the centerpoint-based aggregation should be resilient in such a network.

We run the noncooperative SGD and cooperative SGD with average/CM/GM/centerpoint-based aggregation rules to estimate the target location $\theta_{k,i}$ and the velocity $v_{k,i}^g$. In the case of attack, Byzantine agents continuously send (0,0) to all normal agents as their current estimates of the target location and velocity. Fig. 4 shows the final deployment of agents after 500 iterations, where the yellow star represents the target.

In the case of no attack, we find all the four aggregation rules—average, CM, GM, and centerpoint—converge to the target as shown in Fig. 4(a). However, in the presence of Byzantine

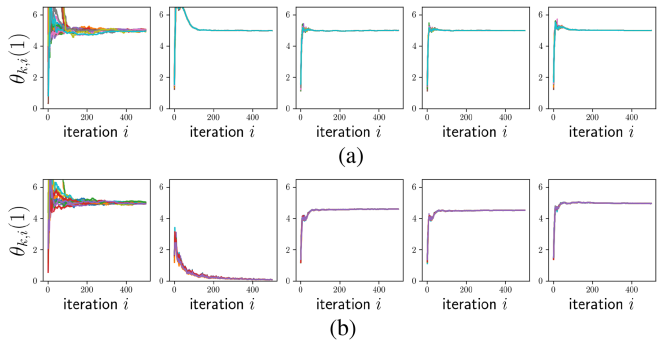


Fig. 5. Static target estimates $\theta_{k,i}$ (1^{st} dimension). From left to right: noncooperative SGD, cooperative SGD with average/CM/GM/centerpoint-based aggregation. (a) No attack. (b) With 5 Byzantine agents.

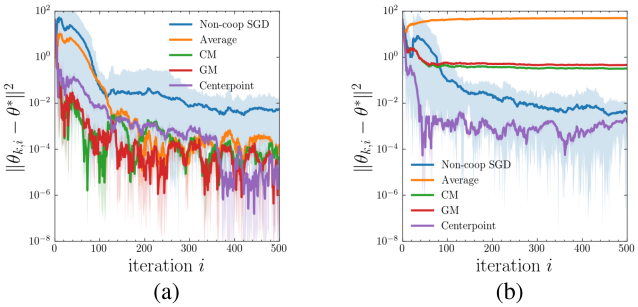


Fig. 6. Estimation accuracy $\|\theta_{k,i} - \theta^*\|^2$ for $k \in \mathcal{N}$ with different aggregation rules for static target. (a) No attack. (b) With 5 Byzantine agents.

agents, only the centerpoint-based cooperative SGD converges to the target as shown in Fig. 4(b). Fig. 5 illustrates the state estimates as a function of time, where each line represents the estimates of a normal agent k. The learning accuracy (mean and range) measured by $\|\theta_{k,i} - \theta^*\|^2$, for $k \in \mathcal{N}$ is illustrated in Fig. 6, where lines are the average values, and shaded area is the range between the minimum and the maximum values among the network. We observe that cooperative SGD with all the four aggregation rules achieve a better average learning accuracy at convergence (measured by $\sum_{k\in\mathcal{N}} \|\theta_{k,i} - \theta^*\|^2/|\mathcal{N}|$) than the noncooperative SGD under no attack, whereas only the centerpoint-based aggregation achieves a better average learning accuracy than the noncooperative SGD under attack.

A.3. Time-Varying Target

We next consider the case when the target is time-varying. Using time-varying target can make robots follow a desired trajectory, which can be used in swarm robotics. The location of the time-varying target is given by $(5 + \cos(0.01i), 5 + \sin(0.01i))$. The noise variance of distance $\sigma^2_{d,k} \in [2.0, 3.0], \forall k \in \mathcal{N}$. And we use fixed stepsize $\alpha_{k,i} = 0.05$, for $k \in \mathcal{N}, i \in \mathbb{N}$. The other setups and Byzantine attacks are the same as in Section VI-A.2.

Fig. 7 shows the final deployment of agents after 1000 iterations, where the yellow dashed circle represents the time-varying target trajectory and the yellow star represents the current target. Fig. 8 illustrates the state estimates as a function of time. And the learning accuracy measured by $\|\theta_{k,i} - \theta^*\|^2$ are illustrated

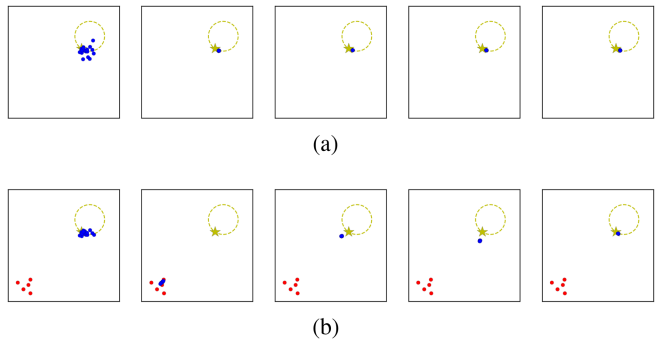


Fig. 7. Mobile network's final deployment for time-varying target. From left to right: noncooperative SGD, cooperative SGD with average/CM/GM/centerpoint-based aggregation. (a) No attack. (b) With 5 Byzantine agents.

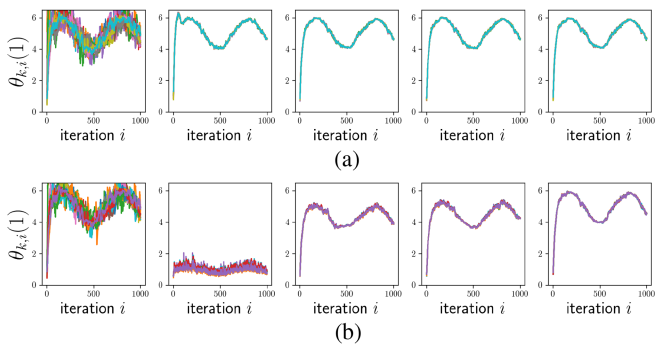


Fig. 8. Time-varying target estimates $\theta_{k,i}$ (1^{st} dimension). From left to right: noncooperative SGD, cooperative SGD with average/CM/GM/centerpoint-based aggregation. (a) No attack. (b) With 5 Byzantine agents.

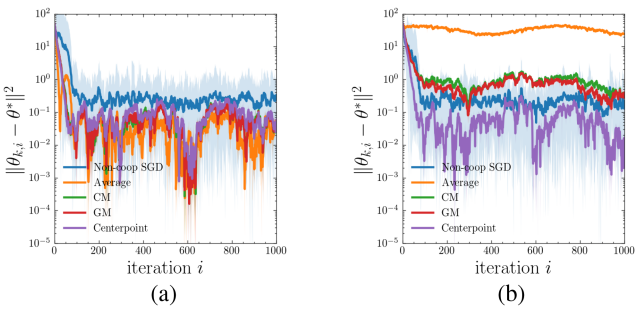


Fig. 9. Estimation accuracy $\|\theta_{k,i} - \theta^*\|^2$ for $k \in \mathcal{N}$ with different aggregation rules, for time-varying target. (a) No attack. (b) With 5 Byzantine agents.

in Fig. 9. The simulation shows similar results to the case of static target.

A.4. Experiments on Robotarium

In addition to the numerical simulations, we carried out similar experiments using real robots on Robotarium [64], a multi-robot testbed developed at the Georgia Institute of Technology. The robots are 11 cm wide, 10 cm long, and operate on a 3m $\times$ 2m area. We denote the bottom-left corner of the arena to be the original point with coordinates $(0, 0)$ and the upper-right corner to be $(3, 2)$.

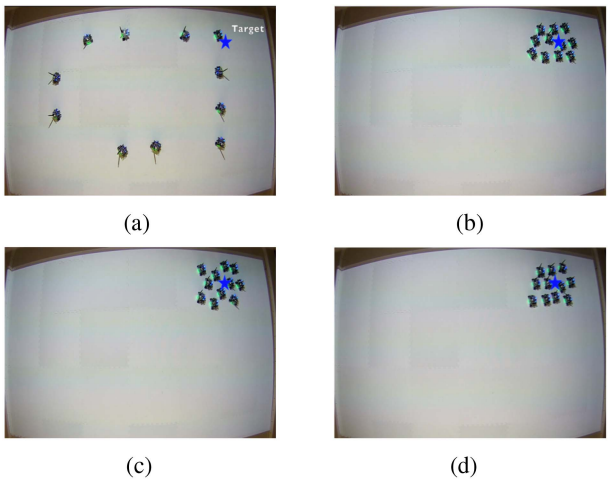


Fig. 10. Network under no attack on Robotarium. (a) Initial network. (b) Final network (CM). (c) Final network (GM). (d) Final network (Centerpoint).

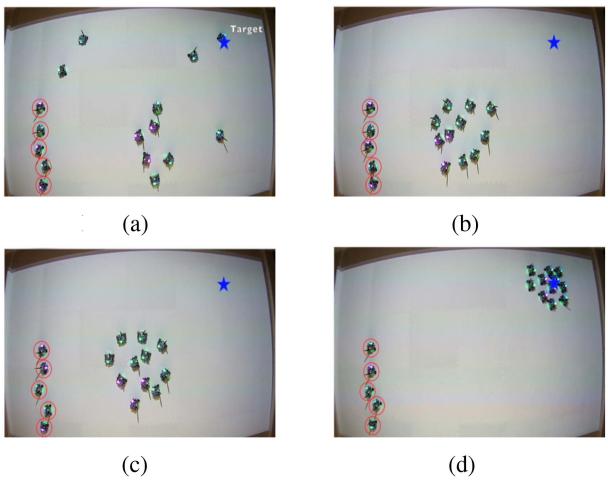


Fig. 11. Network with five Byzantine robots on Robotarium. (a) Initial network. (b) Final network (CM). (c) Final network (GM). (d) Final network (Centerpoint).

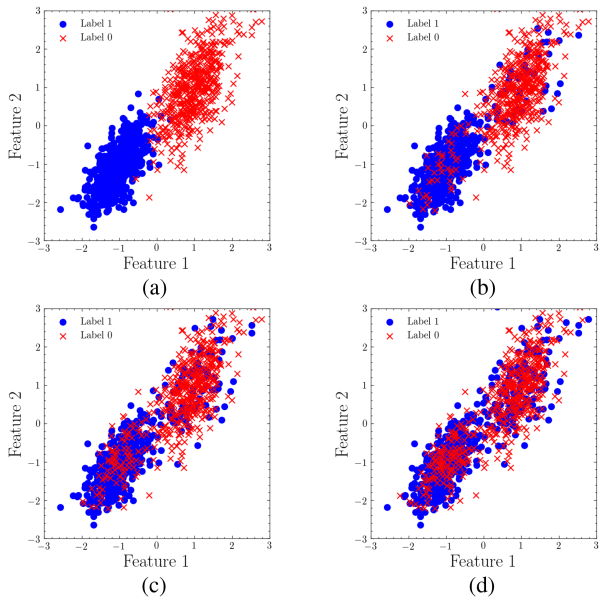


Fig. 12. (a) Real data distribution, (b)–(d) data with outliers received by normal agents. (a) Real distribution. (b) 10% Outliers. (c) 20% Outliers. (d) 30% Outliers.

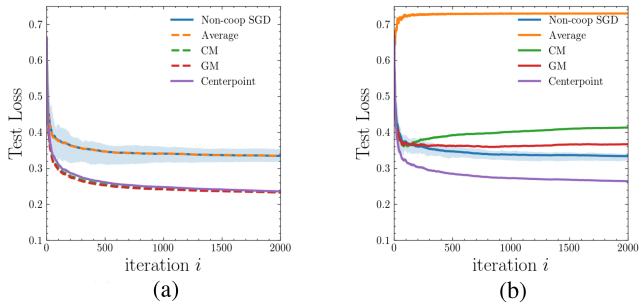


Fig. 13. Test loss on 500 test data samples from the real data distribution without outliers (Normal agents receive training data with uniform outlier rate 20%). (a) No attack. (b) With 3 Byzantine agents.

We consider a network of 11 normal robots. Parameters are selected to be $s = 1, \Delta t = 1s$. The target location is set to be $\theta^* = (2.4, 1.7)$. The regression vector $u_{k,i}$ has uniform covariance matrix $R_{u,k} = \sigma^2_{u,k} I_2$, $\sigma^2_{u,k} \in [0.1, 0.5]$. The noise variance of distance $\sigma^2_{d,k} \in [0.5, 5.0]$. Both $\sigma^2_{d,k}$ and $\sigma^2_{u,k}$ decrease linearly as the distance to the target decreases. The fixed stepsize is 0.2. In the case of attack, five more Byzantine robots are introduced making the total number of robots to be 16. We consider the network to be modeled by a complete graph, where every agent is the neighbor of every other agent. Since the centerpoint-based aggregation rule is resilient up to $\lceil \frac{16}{3} \rceil - 1 = 5$ Byzantine robots, we expect it to be resilient in the experiment.

Figs. 10 and 11 show the network deployments using CM/GM/centerpoint-based cooperative SGD under no attack and with attack, respectively. The Byzantine robots are indicated by the red circle, and the target location is highlighted by the blue star. Byzantine robots stay stationary throughout the experiment and continuously send wrong estimates of the target location $(0, 0)$ and velocity vector $(0, 0)$ to normal robots. We adopt the collision avoidance mechanism implemented by Robotarium in our experiment.

The results are similar to the simulation results. Without attacks, robots with CM/GM/centerpoint-based aggregation rules all converge to the target. However, in the presence of Byzantine agents, only robots using the centerpoint-based aggregation rule converge to the target.

B. Pattern Recognition

In the second case study, we consider the case when robots perform a pattern recognition or detection task using sensor readings. We consider a network of 10 agents modeled by a complete graph, where every agent is the neighbor of every other agent. Agents collect two-dimensional features (sensor readings) to perform binary classification. The real data distribution is given

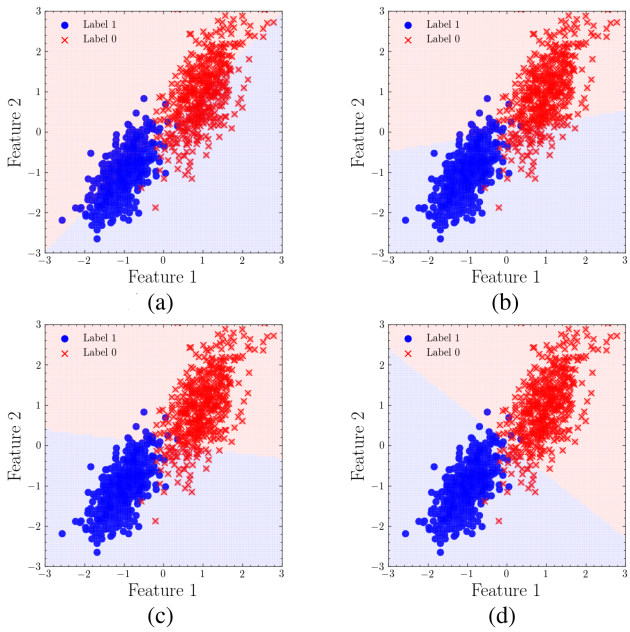


Fig. 14. Decision boundary achieved by different aggregation rules when 3 out of 10 agents are Byzantine. (a) Average. (b) CM. (c) GM. (d) Centerpoint.

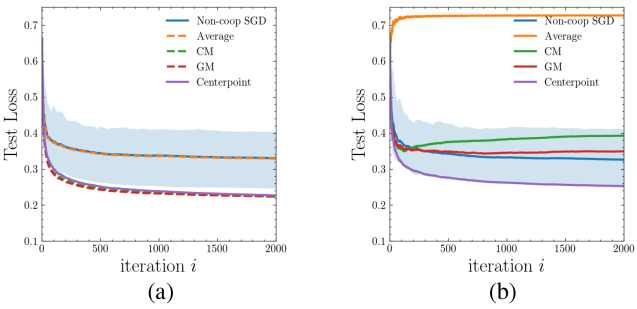


Fig. 15. Test loss on 500 test data samples from the real data distribution without outliers. (Normal agents receive training data with different outlier rates from 10% to 30%). (a) No attack. (b) With 3 Byzantine agents.

in Fig. 12(a). The label-0 data has mean $(1, 1)$ and covariance $((0.1468, 0.9233), (0.1863, 0.3456))$ and label-1 data has mean $(-1, -1)$ and covariance $((0.4170, 0.7203), (0.0001, 0.3023))$. We assume outliers in the training data such that the labels of the outliers are inverted – from 0 to 1 or from 1 to 0. The data distribution with $10\% - 30\%$ outliers is illustrated in Fig. 12 b–d. We use logistic regression to classify the data. The global cost function has the following form:

$$\min_{\theta} \Bigg\{ F(\theta) \triangleq \frac{1}{|\mathcal{N}|} \sum_{k \in \mathcal{N}} -\mathbb{E}\Big\{ y_k^i \log\left(g\left(\theta^\top x_k^i\right)\right) + \left(1 - y_k^i\right) \log\left(1 - g\left(\theta^\top x_k^i\right)\right) \Big\} \Bigg\} \tag{17}$$

where $g(z) = \frac{1}{1+e^{-z}}$. The cooperative SGD algorithm in (2) and (3) can be used to optimize the above cost function. The time-varying stepsize is $\alpha_{k,i} = \frac{1}{i+10}$ for $k \in \mathcal{N}$.

We consider two scenarios. In the first scenario, every normal agent receives data with uniform outlier rate 20%. This simulates the case in which agents receive similar data resulting in similar learning performance. We compute the test loss of normal agents over 500 data samples from the real data distribution without outliers by (17). In the case of attack, we randomly pick 3 out of 10 normal agents as Byzantine agents that continuously send $(1, -1)$ as their estimates to the other normal agents. The test loss for the noncooperative SGD, cooperation using average, CM, GM, and centerpoint is plotted in Fig. 13. Since we consider a complete graph, normal agents receive the same messages in cooperation and therefore their aggregation results are the same. As a result, normal agents share the same test loss in the cooperative cases, which is also the mean of their test losses. We find the centerpoint-based aggregation outperforms the other cooperative aggregation rules as well as the noncooperative SGD. When there is an attack, only the centerpoint-based aggregation converges with a better learning performance measured by the average test loss than the noncooperative SGD. Fig. 14 illustrates the decision boundary achieved by different aggregation rules under attack.

In the second scenario, every normal agent receives data from the real data distribution with different outlier rate of $10\% - 30\%$. This simulates the case in which agents receive different data resulting in different learning performance. Figs. 15 illustrates the learning results. We observe that the results are similar to the previous example.

VII. Conclusion

The major computational step in the the proposed approach is the computation of a point in the safe region $\text{Safe}_f(S)$. We do so by computing a *centerpoint* of a set of points S in dimension d. If we have a set of N points (i.e., $|S| = N$), then a centerpoint can be computed in $O(N)$ time in $d = 2$, and $O(N^2)$ time in $d = 3$. Since in robotic applications, the position vector is in two or three dimensions, the case of centerpoint computation in $d = 2, 3$ is of particular interest. In general, the problem of checking whether a point is a centerpoint of a given set of points or not is a co-NP-complete problem. In higher dimensions $(d \geq 4)$, the complexity of finding a centerpoint is unknown. However, there exist approximation algorithms and randomized algorithms that compute approximate centerpoints. We also note that a point in a safe region $\text{Safe}_f(S)$ can also be computed using other techniques, for instance, through linear programming [65]. The linear program uses a total of $\binom{n}{n-f}(d+1+n-f)$ constraints in $d + \binom{n}{n-f}(n-f)$ variables, which cannot be solved in polynomial time for $f = \lceil \frac{n}{d+1} \rceil - 1$ with the number of variables and constraints that are not polynomial in n. However, centerpoint-based computation of a point in $\text{Safe}_f(S)$ offers more advantages in terms of computational complexity and characterization.

In this work, we studied the resilient aggregation rules for DML algorithms. We showed that the commonly used CM and geometric median-based aggregation methods do not guarantee resilient convergence for distributed learning. We proposed a centerpoint-based aggregation rule that generalizes

the resilience property of the median into higher dimensions. The centerpoint-based aggregation rule guarantees that the distributed learning algorithms converge to the optimum state if the number of Byzantine agents in a normal agent's neighborhood is less than $\lceil \frac{n_k}{d+1} \rceil$, where n_k is the number of agents in the neighborhood, and d is the dimension of the state vector of the agents. Finally, we note that the framework and the corresponding methods and analysis can be easily generalized to federated learning. We aim to explore the tradeoff between the computational cost for resilient aggregation and the degradation in learning performance for future work.

APPENDIX A
PROOF OF LEMMA 1

Proof: Our proof is based on the convergence proof of SGD [54, Th. 4.6].

Since F has an L-Lipschitz continuous gradient, it holds that

$$F\left(\hat{\theta}_{k,i+1}\right) - F\left(\theta_{k,i}\right) \leq \nabla F\left(\theta_{k,i}\right)^{\top}\left(\hat{\theta}_{k,i+1} - \theta_{k,i}\right) + \frac{1}{2}L\|\hat{\theta}_{k,i+1} - \theta_{k,i}\|^2.$$

Given the SGD step (2), we have

$$F\left(\hat{\theta}_{k,i+1}\right) - F\left(\theta_{k,i}\right) \leq -\alpha_{k,i}\nabla F(\theta_{k,i})^{\top}\nabla \ell_k(\theta_{k,i};\xi_k^i) + \frac{1}{2}\alpha_{k,i}^2 L\|\nabla \ell_k\left(\theta_{k,i};\xi_k^i\right)\|^2. \tag{18}$$

Take the expected value of the above equation with respect to the random variable ξ_k^i. Since $\hat{\theta}_{k,i+1}$ depends on ξ_k^i, whereas $\theta_{k,i}$ does not, we obtain

$$\begin{aligned}&\mathbb{E}_{\xi_k^i}\left[F\left(\hat{\theta}_{k,i+1}\right)\right] - F(\theta_{k,i})\\ &\quad\leq -\alpha_{k,i}\nabla F(\theta_{k,i})^{\top}\mathbb{E}_{\xi_k^i}\left[\nabla \ell_k\left(\theta_{k,i};\xi_k^i\right)\right]\\ &\quad\quad+\frac{1}{2}\alpha_{k,i}^2 L\mathbb{E}_{\xi_k^i}\left[\|\nabla \ell_k\left(\theta_{k,i};\xi_k^i\right)\|^2\right].\end{aligned}$$

Given Assumption 3, it follows that

$$\begin{aligned}&\mathbb{E}_{\xi_k^i}\left[F\left(\hat{\theta}_{k,i+1}\right)\right] - F(\theta_{k,i})\\ &\leq -\mu\alpha_{k,i}\|\nabla F\left(\theta_{k,i}\right)\|^2 + \frac{1}{2}\alpha_{k,i}^2 L\mathbb{E}_{\xi_k^i}\left[\|\nabla \ell_k\left(\theta_{k,i};\xi_k^i\right)\|^2\right]\\ &\leq -\mu\alpha_{k,i}\|\nabla F(\theta_{k,i})\|^2\\ &\quad+\frac{1}{2}\alpha_{k,i}^2 L\left(M_k + (V_k + \mu_g^2)\|\nabla F(\theta_{k,i})\|^2\right)\\ &= -\left(\mu - \frac{1}{2}\alpha_{k,i}LG_k\right)\alpha_{k,i}\|\nabla F\left(\theta_{k,i}\right)\|^2 + \frac{1}{2}\alpha_{k,i}^2 LM_k\end{aligned} \tag{19}$$

where $G_k \triangleq V_k + \mu_g^2$.

Given that F is strongly convex, there exists $0 < c \leq L$ such that

$$\|\nabla F(\theta)\|^2 \geq 2c\left(F(\theta) - F^*\right) \text{ for all } \theta.$$

Also, since $\alpha_{k,i} \leq \frac{\mu}{LG_k}$, it holds that $\alpha_{k,i}LG_k \leq \mu$. Following 19, We have

$$\begin{aligned}&\mathbb{E}_{\xi_k^i}\left[F\left(\hat{\theta}_{k,i+1}\right)\right] - F(\theta_{k,i})\\ &\leq -\frac{1}{2}\alpha_{k,i}\mu\|\nabla F(\theta_{k,i})\|^2 + \frac{1}{2}\alpha_{k,i}^2 LM_k\\ &\leq -\alpha_{k,i}c\mu\left(F(\theta_{k,i}) - F^*\right) + \frac{1}{2}\alpha_{k,i}^2 LM_k.\end{aligned}$$

Subtracting F^* from both sides of (19) and taking total expectations over the joint distribution $\xi_{k,i}$ for all $k \in \mathcal{N}$, $i \in \mathbb{N}$, we have

$$\mathbb{E}\left[F\left(\hat{\theta}_{k,i+1}\right) - F^*\right] \leq (1 - \alpha_{k,i}c\mu)\mathbb{E}\left[F\left(\theta_{k,i}\right) - F^*\right] + \frac{1}{2}\alpha_{k,i}^2 LM_k$$

which completes the proof. □

REFERENCES

[1] J. Li, W. Abbas, M. Shabbir, and X. Koutsoukos, "Resilient distributed diffusion for multi-robot systems using centerpoint," in *Proc. Robot.: Sci. Syst.*, Corvalis, Oregon, USA, 2020.

[2] A. H. Sayed, S. Tu, J. Chen, X. Zhao, and Z. J. Towfic, "Diffusion strategies for adaptation and learning over networks: An examination of distributed strategies and network behavior," *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 155–171, May 2013.

[3] A. H. Sayed, "Adaptive networks," *Proc. IEEE*, vol. 102, no. 4, pp. 460–497, Apr. 2014.

[4] A. Lalitha, O. C. Kilinc, T. Javidi, and F. Koushanfar, "Peer-to-peer federated learning on graphs," 2019, *arXiv:1901.11173*.

[5] J. Plata-Chaves, N. Bogdanovic, and K. Berberidis, "Distributed diffusion-based LMS for node-specific adaptive parameter estimation," *IEEE Trans. Signal Process.*, vol. 63, no. 13, pp. 3448–3460, Jul. 2015.

[6] R. Abdolee, S. Saur, B. Champagne, and A. H. Sayed, "Diffusion LMS localization and tracking algorithm for wireless cellular networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2013, pp. 4598–4602.

[7] X. Zhao and A. H. Sayed, "Clustering via diffusion adaptation over networks," in *Proc. 3rd Int. Workshop Cogn. Inf. Process.*, 2012, pp. 1–6.

[8] S. Tu and A. H. Sayed, "Mobile adaptive networks," *IEEE J. Sel. Topics Signal Process.*, vol. 5, no. 4, pp. 649–664, Aug. 2011.

[9] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. 20th Int. Conf. Artif. Intell. Statist.*, Fort Lauderdale, FL, USA, 2017, pp. 1273–1282.

[10] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Proc. Neural Inf. Process. Syst.*, 2017, pp. 119–129.

[11] J. Li, W. Abbas, and X. Koutsoukos, "Resilient distributed diffusion in networks with adversaries," *IEEE Trans. Signal Inf. Process. over Netw.*, vol. 6, pp. 1–17, 2020.

[12] D. Yin, Y. Chen, K. Ramchandran, and P. L. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 5636–5645.

[13] X. Chen, T. Chen, H. Sun, Z. S. Wu, and M. Hong, "Distributed training with heterogeneous data: Bridging median- and mean-based algorithms," 2019, *arXiv:1906.01736*.

[14] H. Yang, X. Zhang, M. Fang, and J. Liu, "Byzantine-resilient stochastic gradient descent for distributed learning: A lipschitz-inspired coordinate-wise median approach," 2019, *arXiv: 1909.04532*.

[15] Z. Yang and W. U. Bajwa, "ByRDiE: Byzantine-resilient distributed coordinate descent for decentralized learning," *IEEE Trans. Signal Inf. Process. over Netw.*, vol. 5, no. 4, pp. 611–627, Dec. 2019.

[16] Y. Chen, L. Su, and J. Xu, "Distributed statistical machine learning in adversarial settings: Byzantine gradient descent," *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 1, no. 2, pp. 1–25, 2017.

[17] K. Pillutla, S. M. Kakade, and Z. Harchaoui, "Robust aggregation for federated learning," 2019, *arXiv:1912.13445*.

[18] E. M. E. Mhamdi, R. Guerraoui, and S. Rouault, "The hidden vulnerability of distributed learning in Byzantium," in *Proc. 35th Int. Conf. Mach. Learn.*, Stockholmsmässan, Stockholm, Sweden, 2018, pp. 3518–3527.
[19] E. El-Mhamdi and R. Guerraoui, "Fast and secure distributed learning in high dimension," 2019, *arXiv:1905.04374*.
[20] G. Baruch, M. Baruch, and Y. Goldberg, "A little is enough: Circumventing defenses for distributed learning," in *Proc. Neural Inf. Process. Syst.*, 2019, pp. 8632–8642.
[21] C. Xie, O. Koyejo, and I. Gupta, "Fall of empires: Breaking byzantine-tolerant SGD by inner product manipulation," in *Proc. 35th Conf. Uncertainty Artif. Intell.*, 2019, paper no. 83.
[22] M. Fang, X. Cao, J. Jia, and N. Z. Gong, "Local model poisoning attacks to Byzantine-robust federated learning," 2019, *arXiv:1911.11815*.
[23] D. Dolev, N. A. Lynch, S. S. Pinter, E. W. Stark, and W. E. Weihl, "Reaching approximate agreement in the presence of faults," *J. ACM*, vol. 33, no. 3, pp. 499–516, 1986.
[24] H. Mendes and M. Herlihy, "Multidimensional approximate agreement in Byzantine asynchronous systems," in *Proc. 45th Annu. ACM Symp. Theory Comput.*, 2013, pp. 391–400.
[25] N. H. Vaidya and V. K. Garg, "Byzantine vector consensus in complete graphs," in *Proc. ACM Symp. Princ. Distrib. Comput.*, 2013, pp. 65–73.
[26] R. M. Kieckhafer and M. H. Azadmanesh, "Reaching approximate agreement with mixed-mode faults," *IEEE Trans. Parallel Distrib. Syst.*, vol. 5, no. 1, pp. 53–63, Jan. 1994.
[27] H. LeBlanc, H. Zhang, X. D. Koutsoukos, and S. Sundaram, "Resilient asymptotic consensus in robust networks," *IEEE J. Sel. Areas Commun.*, vol. 31, no. 4, pp. 766–781, Apr. 2013.
[28] N. H. Vaidya, L. Tseng, and G. Liang, "Iterative approximate byzantine consensus in arbitrary directed graphs," in *Proc. ACM Symp. Princ. Distrib. Comput.*, Funchal, Madeira, Portugal, 2012, pp. 365–374.
[29] H. Zhang and S. Sundaram, "A simple median-based resilient consensus algorithm," in *Proc. 50th Annu. Allerton Conf. Commun., Control, Comput.*, 2012, pp. 1734–1741.
[30] M. Franceschelli, A. Giua, and A. Pisano, "Finite-time consensus on the median value with robustness properties," *IEEE Trans. Autom. Control*, vol. 62, no. 4, pp. 1652–1667, Apr. 2017.
[31] A. Pilloni, A. Pisano, M. Franceschelli, and E. Usai, "Robust distributed consensus on the median value for networks of heterogeneously perturbed agents," in *Proc. IEEE 55th Conf. Decis. Control*, 2016, pp. 6952–6957.
[32] N. H. Vaidya, "Iterative Byzantine vector consensus in incomplete graphs," in *Proc. Int. Conf. Distrib. Comput. Netw.*, Springer, 2014, pp. 14–28.
[33] X. Wang, S. Mou, and S. Sundaram, "A resilient convex combination for consensus-based distributed algorithms," *Numer. Algebra, Control Optim.*, vol. 9, 2019, Art. no. 269.
[34] M. Shabbir, J. Li, W. Abbas, and X. Koutsoukos, "Resilient vector consensus in multi-agent networks using centerpoints," in *Proc. Amer. Control Conf.*, 2020, pp. 4387–4392.
[35] W. Abbas, M. Shabbir, J. Li, and X. Koutsoukos, "Resilient distributed vector consensus using centerpoint," *Automatica*, vol. 136, 2022, Art. no. 110046.
[36] H. Park and S. Hutchinson, "Fault-tolerant rendezvous of multirobot systems," *IEEE Trans. Robot.*, vol. 33, no. 3, pp. 565–582, Jun. 2017.
[37] L. Guerrero-Bonilla, A. Prorok, and V. Kumar, "Formations for resilient robot teams," *IEEE Robot. Automat. Lett.*, vol. 2, no. 2, pp. 841–848, Apr. 2017.
[38] K. Saulnier, D. Saldana, A. Prorok, G. J. Pappas, and V. Kumar, "Resilient flocking for mobile robot teams," *IEEE Robot. Automat. Lett.*, vol. 2, no. 2, pp. 1039–1046, Apr. 2017.
[39] Z. Li, W. Trappe, Y. Zhang, and B. Nath, "Robust statistical methods for securing wireless localization in sensor networks," in *Proc. 4th Int. Symp. Inf. Process. Sensor Netw.*, UCLA, Los Angeles, California, USA, 2005, pp. 91–98.
[40] Y. Zeng, J. Cao, J. Hong, S. Zhang, and L. Xie, "Secure localization and location verification in wireless sensor networks: A survey," *J. Supercomput.*, vol. 64, no. 3, pp. 685–701, 2013.
[41] C. Xie, S. Koyejo, and I. Gupta, "Zeno: Distributed stochastic gradient descent with suspicion-based fault-tolerance," in *Proc. 36th Int. Conf. Mach. Learn.*, 2019, pp. 6893–6901.
[42] N. Gupta and N. H. Vaidya, "Byzantine fault tolerant distributed linear regression," 2019, *arXiv:1903.08752*.
[43] L. Su and N. H. Vaidya, "Byzantine-resilient multi-agent optimization," *IEEE Trans. Autom. Control*, vol. 66, no. 5, pp. 2227–2233, May 2021.
[44] L. Chen, H. Wang, Z. B. Charles, and D. S. Papailiopoulos, "DRACO: Byzantine-resilient distributed training via redundant gradients," in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 902–911.
[45] S. Rajput, H. Wang, Z. Charles, and D. Papailiopoulos, "DETOX: A redundancy-based framework for faster and more robust gradient aggregation," in *Proc. Neural Inf. Process. Syst.*, 2019, pp. 10320–10330.
[46] D. Data, L. Song, and S. N. Diggavi, "Data encoding methods for byzantine-resilient distributed optimization," in *Proc. IEEE Int. Symp. Inf. Theory*, 2019, pp. 2719–2723.
[47] L. Li, W. Xu, T. Chen, G. B. Giannakis, and Q. Ling, "RSA: Byzantine-robust stochastic aggregation methods for distributed learning from heterogeneous datasets," in *Proc. 33rd AAAI Conf. Artif. Intell.*, 2019, pp. 1544–1551.
[48] Z. Yang, A. Gang, and W. U. Bajwa, "Adversary-resilient distributed and decentralized statistical inference and machine learning: An overview of recent advances under the byzantine threat model," *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 146–159, May 2020.
[49] E.-M. El-Mhamdi, R. Guerraoui, A. Guirguis, and S. Rouault, "SGD: Decentralized byzantine resilience," 2019, *arXiv:1905.03853*.
[50] O. Dekel, R. Gilad-Bachrach, O. Shamir, and L. Xiao, "Optimal distributed online prediction using mini-batches," *J. Mach. Learn. Res.*, vol. 13, pp. 165–202, 2012.
[51] J. B. Predd, S. R. Kulkarni, and H. V. Poor, "Distributed learning in wireless sensor networks," *IEEE Signal Process. Mag.*, vol. 23, no. 4, pp. 56–69, Jul. 2006.
[52] Z. J. Towfic, J. Chen, and A. H. Sayed, "Excess-risk of distributed stochastic learners," *IEEE Trans. Inf. Theory*, vol. 62, no. 10, pp. 5753–5785, Oct. 2016.
[53] A. H. Sayed, "Adaptation, learning, and optimization over networks," *Found. Trends Mach. Learn.*, vol. 7, no. 4–5, pp. 311–801, 2014.
[54] L. Bottou, F. E. Curtis, and J. Nocedal, "Optimization methods for large-scale machine learning," *SIAM Rev.*, vol. 60, no. 2, pp. 223–311, 2018.
[55] X. Zhao and A. H. Sayed, "Performance limits for distributed estimation over LMS adaptive networks," *IEEE Trans. Signal Process.*, vol. 60, no. 10, pp. 5107–5124, Oct. 2012.
[56] S. Tu and A. H. Sayed, "Diffusion strategies outperform consensus strategies for distributed estimation over adaptive networks," *IEEE Trans. Signal Process.*, vol. 60, no. 12, pp. 6217–6234, Dec. 2012.
[57] J. Matoušek, *Lectures on Discrete Geometry*. Berlin, Germany: Springer, 2002.
[58] J. De Loera, X. Goaoc, F. Meunier, and N. Mustafa, "The discrete yet ubiquitous theorems of Carathéodory, Helly, Sperner, Tucker, and Tverberg," *Bull. Amer. Math. Soc.*, vol. 56, no. 3, pp. 415–511, 2019.
[59] N. H. Mustafa, S. Ray, and M. Shabbir, "k-centerpoints conjectures for pointsets in $\mathbb{R}^d$," *Int. J. Comput. Geometry Appl.*, vol. 25, no. 3, pp. 163–185, 2015.
[60] S. Jadhav and A. Mukhopadhyay, "Computing a centerpoint of a finite planar set of points in linear time," *Discrete Comput. Geometry*, vol. 12, no. 3, pp. 291–312, 1994.
[61] T. M. Chan, "An optimal randomized algorithm for maximum tukey depth," in *Proc. 15th Annu. ACM-SIAM Symp. Discrete Slgorithms*, 2004, pp. 430–436.
[62] G. L. Miller and D. R. Sheehy, "Approximate centerpoints with proofs," *Comput. Geometry*, vol. 43, no. 8, pp. 647–654, 2010.
[63] W. Mulzer and D. Werner, "Approximating Tverberg points in linear time for any fixed dimension," *Discret. Comput. Geom.*, vol. 50, no. 2, pp. 520–535, 2013.
[64] D. Pickem *et al.*, "The Robotarium: A remotely accessible swarm robotics research testbed," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2017, pp. 1699–1706.
[65] H. Mendes, M. Herlihy, N. H. Vaidya, and V. K. Garg, "Multidimensional agreement in Byzantine systems," *Distrib. Comput.*, vol. 28, no. 6, pp. 423–441, 2015.

Jiani Li received the Ph.D. degree in computer science from the Electrical Engineering and Computer Science Department, Vanderbilt University, Nashville, TN, USA, in 2021.

She is currently working as a Research Scientist with Meta. Her research focuses on resilient multiagent distributed systems and resilient design of cyber-physical systems with machine learning components.

Waseem Abbas (Member, IEEE) received the M.Sc. and Ph.D. degrees in electrical and computer engineering, from Georgia Institute of Technology, Atlanta, GA, USA, in 2013 and 2010, respectively.

He is currently an Assistant Professor with the System Engineering Department, University of Texas at Dallas, Dallas, TX, USA. Previously, he was a Research Assistant Professor with the Vanderbilt University, Nashville, TN, USA. He was a Fulbright scholar from 2009 till 2013. His research interests are in the areas of control of networked systems, resilience and robustness in networks, distributed optimization, and graph-theoretic methods in complex networks.

Mudassir Shabbir received the Ph.D. degree in computer science from Division of Computer Science, Rutgers University, NJ, USA, in 2014.

He is currently an Associate Professor with the Department of Computer Science, Information Technology University, Lahore, Pakistan and a Research Assistant Professor with Vanderbilt University, Nashville TN, USA. Previously, he has worked with Lahore University of Management Sciences, Lahore, Pakistan, Los Alamos National Labs, New Mexico, Bloomberg L.P. New York, NY, USA, and with Rutgers University, NJ, USA. He was Rutgers Honors Fellow form 2011 to 2012. He also works in graph machine learning and resilient network systems. His main area of research is algorithmic and discrete geometry, and has developed new methods for the characterization and computation of succinct representations of large datasets with applications in nonparametric statistical analysis.

Xenofon Koutsoukos (Fellow, IEEE) received his Ph.D. degree in electrical engineering from the University of Notre Dame in 2000. He is currently a Professor and the Chair of the Department of Computer Science and a Senior Research Scientist with the Institute for Software Integrated Systems (ISIS), Vanderbilt University, Nashville, TN, USA.

He was a Member of Research Staff with the Xerox Palo Alto Research Center (PARC) (2000–2002). He has authored or coauthored more than 300 journal and conference papers and he is coinventor of four US patents. His research work is in the area of cyber-physical systems with emphasis on learning-enabled systems, formal methods, distributed algorithms, security and resilience, diagnosis and fault tolerance, and adaptive resource management.

Prof. Koutsoukos was the recipient of the NSF Career Award in 2004, the Excellence in Teaching Award in 2009 from the Vanderbilt University School of Engineering, and the 2011 NASA Aeronautics Research Mission Directorate (ARMD) Associate Administrator (AA) Award in Technology and Innovation. He was named a Fellow of the IEEE for his contributions to the design of resilient cyber-physical systems.