A CASE OF DISTRIBUTED OPTIMIZATION IN ADVERSARIAL ENVIRONMENT

Nikhil Ravi, Anna Scaglione, Angelia Nedić

School of Electrical Computer and Energy Engineering
Arizona State University, Tempe, USA
{Nikhil.Ravi, Anna.Scaglione, Angelia.Nedich}@asu.edu

ABSTRACT

In this paper, we consider the problem of solving a distributed (consensus-based) optimization problem in a network that contains regular and malicious nodes (agents). The regular nodes are performing a distributed iterative algorithm to solve their associated optimization problem, while the malicious nodes inject false data with a goal to steer the iterates to a point that serves their own interest. The problem consists of detecting and isolating the malicious agents, thus allowing the regular nodes to solve their optimization problem. We propose a method to dwarf data injection attacks on distributed optimization algorithms, which is based on the idea that the malicious nodes (individually or in collaboration) tend to give themselves away when broadcasting messages with the intention to drive the consensus value away from the optimal point for the regular nodes in the network. In particular, we provide a new *gradient-based metric* to detect the neighbors that are likely to be malicious. We also provide some simulation results demonstrating the performance of the proposed approach.

Index Terms— distributed optimization, adversarial nodes, byzantine fault tolerance

1. INTRODUCTION

Many problems in machine learning and social sciences can be generalized as an optimization problem of minimizing the finite sum of local functions:

$$\min_{x\in\mathbb{R}^d} \sum_{i=1}^{N} \ell_i(x), \tag{1}$$

where $N \geq 1$ is the number of nodes (agents) and $\ell_i : \mathbb{R}^d \to \mathbb{R}$ is a convex objective function of node i. With the emergence of big data paradigm and the large sensor networks, solving problem (1) in a centralized manner can be computationally prohibitive, thus giving rise to the distributed optimization framework. In this framework, every node in a network performs local computations and locally exchanges its estimates with the neighbors, whereby at each time step, every node updates using its own estimates and a weighted combination of its neighbors' estimates. In this way, the problem in (1) is solved using a consensus-type distributed optimization (DO) method. Starting with [1], various DO algorithms have been proposed to solve problems of the type presented in (1) for convex and non-convex problems see [2, 3, 4, 5, 6].

In this paper, we consider the case where each node broadcasts its estimate to all its neighbors at each time-step. This exchange of data is vulnerable to noisy transmission media, non-synchronous clocks, external attacks and insider attacks. These irregularities may drive the algorithm iterates to a point in the neighborhood of the optimal point. The attacks are usually modelled as Byzantine attacks due to the unpredictability in the behavior of the malicious nodes. We model the adversarial environment in a similar fashion to the Byzantine generals' problem [7, 8, 9, 10]. While external attacks may be mitigated by employing encryption and authentication, *robust* distributed algorithms have been developed to overcome insider attacks [11, 12, 13]. Two algorithms, one based on coordinate-wise median and another based on coordinate-wise trimmed mean, have been developed in [14] which are Byzantine-robust. Another popular approach to attacker suppression involves regularizing the objective function of the optimization problem (1). In [11] it has been shown that optimizing the objective cost function regularized by a total variation term has appealing robustness property to unreliable agents injecting false values in the network. A drawback in such an approach is that it is difficult to find the right regularizer for a given attack scenario. Moreover, it is especially difficult to design a general regularizer that is resilient to any type of attacks for any network topology.

The detection and localization of malicious nodes in the network have been extensively studied. In [15] it has been proposed to detect the malicious nodes based on the discrepancies in convergence times. Another popular approach is the use of detection theory and hypothesis testing to test for statistical anomalies in the presence of an attacker [16]. A system theoretic approach is presented in [17]. However, these methods tend to focus on a specific attack scenario, and require multiple iterations of the algorithm to detect and localize the malicious nodes.

This work was supported in part by the National Science Foundation CCF-BSF 1714672 grant.

In this paper, we consider a network of agents that aim to solve problem (1) by implementing the Fast Row-stochastic Optimization with uncoordinated STep-sizes (FROST) algorithm proposed in [18]. FROST requires a row-stochastic weights that are locally assigned to the incoming information, it does not require coordination of the stepsize selection among the nodes, and it works in networks with directed communication links. We consider an approach that leverages the self-healing properties of distributed optimization where nodes mimic the iterates of FROST but dynamically sever ties with neighbors that are more likely to be attackers rather than regular nodes. Every regular node accumulates evidence about the trends of *averaged gradient variations* (to be defined precisely) and uses this evidence to decide on the neighbors that needs to be excluded from its neighbor set. The paper is organized as follows. In the forthcoming Section 2, we describe the network model and the consensus-based optimization algorithm FROST from [18]. In Section 3, we introduce the malicious agents behavior and provide an insight into the gradient-based signatures they leave behind. In Section 4, we propose a detection strategy based on a local gradient variance and present some simulation results that demonstrate the proposed approach. We conclude in Section 5.

2. NETWORK MODEL AND PROBLEM FORMULATION

We shall model the agent system by a digraph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} := \{1, 2, \ldots, N\}$ represents the N agents in the system and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ represents the set of directed links. If $(i, j) \in \mathcal{E}$, then node j can receive information from node i; node i is said to be an in-neighbor of node j. We define the *in*-neighborhood of node j by $\mathcal{N}_{[j]}^- := \{i \in \mathcal{V} : (i, j) \in \mathcal{E}\} \cup \{j\}$.

The set of nodes, $\mathcal{V}$, is divided into two subsets; $\mathcal{V}_r$ – the set of regular nodes (RNs) with $N_r := |\mathcal{V}_r|$ and $\mathcal{V}_m$ – the set of malicious nodes (MNs) with $N_m := |\mathcal{V}_m|$ whose existence (if any) the RNs strive to detect and sever the incoming links from them. The communication network is captured by a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \mathcal{V}_r \cup \mathcal{V}_m$ and $\mathcal{E} = \mathcal{E}_r \cup \mathcal{E}_m$; $\mathcal{E}_r$ is the set of links that connect RNs, and $\mathcal{E}_m$ are directed links that emanate from a malicious node and end at a regular node, $\mathcal{E}_m = \{(i, j) \in \mathcal{E} : i \in \mathcal{V}_m,\ j \in \mathcal{V}_r\}$. Thus, the network has an adjacency matrix $\boldsymbol{A}$ and an incidence matrix $\boldsymbol{E}$ associated to $\mathcal{G}_r = (\mathcal{V}_r, \mathcal{E}_r)$ and an adjacency matrix $\boldsymbol{B}$ and incidence matrix $\boldsymbol{M}$ that are associated with the bipartite directed graph $\mathcal{G}_m = (\mathcal{V}, \mathcal{E}_m)$ from the nodes in $\mathcal{V}_m$ to the nodes in $\mathcal{V}_r$.

Consider now a scenario where the regular nodes of the network are trying to solve the optimization problem:

$$\min_{\boldsymbol{x}} \ \ell(\boldsymbol{x}) = \sum_{i \in \mathcal{V}_r} \ell_i(\boldsymbol{x}_i), \ \ s.t. \ (\boldsymbol{E} \otimes \boldsymbol{I}_d)\boldsymbol{x} = \boldsymbol{0}, \quad (2)$$

where $\ell_i : \mathbb{R}^d \to \mathbb{R}$ is the private cost function known only to agent i, and $\boldsymbol{x}$ is a vector obtained by stacking the vectors $\boldsymbol{x}_i$, $i = 1, \ldots, N$. The equality constraint enforces consensus in $\boldsymbol{x}_i, \forall i \in [N]$, at convergence. Distributed optimization algorithms designed for digraphs typically require column-stochastic weight matrix to accommodate the push-sum consensus protocol [19, 20] in combination with the standard gradient method. Further improvements in the convergence rates of such algorithms are presented in DEXTRA [21] and ADD-OPT/Push-DIGing [22], while a linear-rate algorithm is presented in [23]. However, these algorithms require each node to know its out-degrees (the number of its out-neighbors) which is not a realistic assumption in many applications. As noted in Section 1, the FROST algorithm developed in [18] overcomes this constraint.

In the FROST algorithm, at every time step, an agent i maintains three state variable: $\boldsymbol{x}_i[t], \boldsymbol{z}_i[t] \in \mathbb{R}^d$ and $\boldsymbol{y}_i[t] \in \mathbb{R}^N$ and broadcasts their states to their out-neighbors and receive their in-neighbors' states. The update rule at time $t+1$ is as follows:

$$\boldsymbol{x}_i[t+1] = \sum_{j \in \mathcal{N}_{[i]}^-} w_{ij} \boldsymbol{x}_j[t] - \gamma_i \boldsymbol{z}_i[t] \quad (3a)$$

$$\boldsymbol{y}_i[t+1] = \sum_{j \in \mathcal{N}_{[i]}^-} w_{ij} \boldsymbol{y}_j[t] \quad (3b)$$

$$\boldsymbol{z}_i[t+1] = \sum_{j \in \mathcal{N}_{[i]}^-} w_{ij} \boldsymbol{z}_j[t] + \frac{\boldsymbol{g}_i(\boldsymbol{x}_i[t+1])}{[\boldsymbol{y}_i[t+1]]_i} - \frac{\boldsymbol{g}_i(\boldsymbol{x}_i[t])}{[\boldsymbol{y}_i[t]]_i}, \quad (3c)$$

where $\boldsymbol{g}_i(\boldsymbol{x}_i[\cdot])$ is the gradient of $\ell_i(\boldsymbol{x}_i[\cdot])$ evaluated at $\boldsymbol{x}_i[t]$, γ_i is the step-size of agent i, and $[\boldsymbol{W}]_{ij} = w_{ij}, \forall i, j \in [N]$ are the weights associated with the linear updates with $w_{ij} \geq 0$ for all $j \in \mathcal{N}_{[i]}^-$, $w_{ij} = 0$ for all $j \notin \mathcal{N}_{[i]}^-$, and $\sum_j w_{ij} = 1$ for all $i \in [N]$. Next, consider the following conditions:

1. The underlying communication graph is directed and strongly-connected.
2. Each local function, ℓ_i, is strongly-convex with Lipschitz-continuous gradient, i.e., for any i and $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^d$,
   (a) there exists a positive constant α such that
   $$\ell_i(\boldsymbol{x}_1) - \ell_i(\boldsymbol{x}_2) \leq \boldsymbol{g}_i(\boldsymbol{x}_1)^{\mathrm{T}}(\boldsymbol{x}_1 - \boldsymbol{x}_2) - \frac{\alpha}{2}\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_2^2.$$
   (b) there exists a positive constant β such that
   $$\|\boldsymbol{g}_i(\boldsymbol{x}_1) - \boldsymbol{g}_i(\boldsymbol{x}_2)\|_2 \leq \beta \|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_2.$$
3. Each agent in the network has and knows its unique identifier, e.g., $1, \ldots, N$,

When the preceding assumptions are satisfied, the iterates produced by the FROST method (3) converge to the optimal (consensual) point $\boldsymbol{x}^*$ of the problem (2) in the case when all the nodes are regular, i.e., when $\mathcal{V}_r = \mathcal{V}$. However, when some nodes do not follow the update rule (3) due to various reasons such as noisy transmission media or due to the

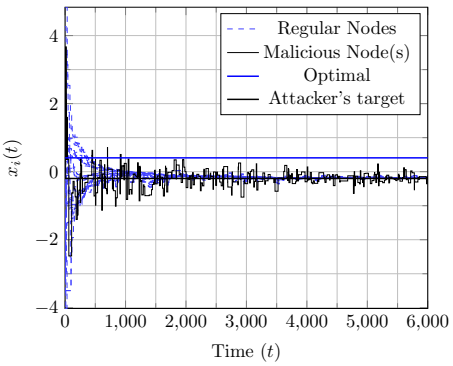


Fig. 1. Consensus without attacker detection under a constant attack scenario.

presence of malicious nodes, the FROST algorithm may converge to a point away from the optimal consensual point or not converge at all. It is, therefore, important to devise detection strategies to dynamically cut links that emanate from malicious nodes, i.e., the links in the set $\mathcal{E}_m$.

Since the process of suppression is imperfect, the network will not reach asymptotically the optimum outcome. Assuming that the algorithm converges, it will most likely converge to a non-optimal point $\boldsymbol{x}[\infty] \neq \boldsymbol{x}^*$, and we can characterize the regret of the solution $\boldsymbol{x}[\infty]$ as follows:

$$\frac{1}{N_r} \sum_{i \in \mathcal{V}_r} |\ell_i(\boldsymbol{x}_i[\infty]) - \ell_i(\boldsymbol{x}^*)|. \tag{4}$$

3. MODELING AND CHARACTERIZATION OF MALICIOUS NODES' BEHAVIOR

In this section, we shall discuss the various avenues of attacks a MN may incorporate. Without a plausible attacker suppression algorithm in place, the iterations produced by RNs will converge, albeit to a point where the attackers intend to drive them to. This can be seen in Figure 1, where the convergence occurs to a non-optimal point under a constant attack scenario, where the malicious nodes persistently send the same value to their neighbors. The successful detection of the malicious nodes depends on finding the aberrations in the statistical trends of the information received by the RNs. When the adversary follows the algorithm but chooses to manipulate its objective function, assuming that the algorithm converges, let the limit point be $\boldsymbol{x}^a$. When the system is not under an attack, the algorithm will converge to the optimal point $\boldsymbol{x}^*$. That is,

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}} \sum_{i \in \mathcal{V}_r} \ell_i(\boldsymbol{x}_i), \;\; s.t. \;\; (\boldsymbol{E} \otimes \boldsymbol{I}_d)\boldsymbol{x} = \boldsymbol{0},$$
$$\boldsymbol{x}^a = \arg\min_{\boldsymbol{x}} \sum_{i \in \mathcal{V}} \ell_i(\boldsymbol{x}_i), \;\; s.t. \;\; (\boldsymbol{E} \otimes \boldsymbol{I}_d)\boldsymbol{x} = \boldsymbol{0}.$$

At the point $\boldsymbol{x}^a$, since it is optimal point for the latter problem, we have

$$\sum_{i \in \mathcal{V}} \boldsymbol{g}_i(\boldsymbol{x}^a) = \boldsymbol{0},$$

implying that

$$\sum_{i \in \mathcal{V}_m} \boldsymbol{g}_i(\boldsymbol{x}^a) = -\sum_{i \in \mathcal{V}_r} \boldsymbol{g}_i(\boldsymbol{x}^a).$$

If suppose, $\boldsymbol{x}^a = \boldsymbol{x}^* + \boldsymbol{\epsilon}$, using the Taylor expansion of $\boldsymbol{g}_i(\boldsymbol{x}^a)$ about $\boldsymbol{x}^*$, we have

$$\sum_{i \in \mathcal{V}_m} \boldsymbol{g}_i(\boldsymbol{x}^a) \approx -\sum_{i \in \mathcal{V}_r} \left[\boldsymbol{g}_i(\boldsymbol{x}^*) + \boldsymbol{h}_i^{\mathrm{T}}(\boldsymbol{x}^*)\boldsymbol{\epsilon}\right] = -\sum_{i \in \mathcal{V}_r} \boldsymbol{h}_i^{\mathrm{T}}(\boldsymbol{x}^*)\boldsymbol{\epsilon}, \tag{5}$$

where $\boldsymbol{h}_i(\boldsymbol{x}_i[\cdot])$ denotes the Hessian of $\ell_i(\boldsymbol{x}_i[\cdot])$ evaluated at $\boldsymbol{x}_i[t]$, and we have used the fact that $\sum_{i \in \mathcal{V}_r} \boldsymbol{g}_i(\boldsymbol{x}^*) = 0$. Let $\boldsymbol{H} := \sum_{i \in \mathcal{V}_r} \boldsymbol{h}_i^{\mathrm{T}}(\boldsymbol{x}^*)$. Taking the 2-norm squared on both sides of the preceding relation, we obtain

$$\|\sum_{i \in \mathcal{V}_m} \boldsymbol{g}_i(\boldsymbol{x}^a)\|_2^2 \approx \|\boldsymbol{H}\boldsymbol{\epsilon}\|_2^2 = \boldsymbol{\epsilon}^{\mathrm{T}}\boldsymbol{H}^{\mathrm{T}}\boldsymbol{H}\boldsymbol{\epsilon} \geq \|\boldsymbol{\epsilon}\|_2^2 \lambda_{\min}\left(\boldsymbol{H}^{\mathrm{T}}\boldsymbol{H}\right).$$

Thus,

$$\|\boldsymbol{\epsilon}\|_2^2 \leq \frac{\|\sum_{i \in \mathcal{V}_m} \boldsymbol{g}_i(\boldsymbol{x}^a)\|_2^2}{\lambda_{\min}\left(\boldsymbol{H}^{\mathrm{T}}\boldsymbol{H}\right)} \leq \frac{N_m(\sum_{i \in \mathcal{V}_m} \|\boldsymbol{g}_i(\boldsymbol{x}^a)\|_2^2)}{\lambda_{\min}\left(\boldsymbol{H}^{\mathrm{T}}\boldsymbol{H}\right)}$$
$$\leq \frac{N_m^2 \max_{i \in \mathcal{V}_m} \|\boldsymbol{g}_i(\boldsymbol{x}^a)\|_2^2}{N_r^2 \lambda_{\min}\left(\frac{\boldsymbol{H}^{\mathrm{T}}\boldsymbol{H}}{N_r^2}\right)}. \tag{6}$$

From the preceding inequality, we see that the strength of the attack, which the $\boldsymbol{\epsilon}$ represents, essentially depends directly on the total number of attackers and the gradients of the attackers. If an attacker aims to lead the RNs astray by a large ϵ, it comes at the cost of giving themselves away through their larger gradient values. Also, it should be noted that to launch a substantial attack, the ratio of the number of attackers and the number of regular agents needs to be high.

4. ESTIMATION OF NEIGHBORS' GRADIENTS AND DETECTION STRATEGY

In the preceding section, we saw that the attackers' action can result in larger gradient values. Thus, we hypothesize that significant insight can be gained if we can approximate the gradient of our neighbor and track it over time relative to the mean of the gradients of the remaining neighbors. Consider

$$\dot{\boldsymbol{x}}_j[t] := \boldsymbol{x}_j[t] - \boldsymbol{x}_j[t-1] \cong \gamma_j \boldsymbol{z}_j[t-1]. \tag{7}$$

Let us use $\dot{\boldsymbol{x}}_j, \forall j \in \mathcal{N}_{[i]}^-$, as an approximation to the gradients of the neighbors of agent i. Consider the following metric

$$S_{ij}[t] = \|\dot{\boldsymbol{x}}_j[t] - \frac{1}{|\mathcal{N}_{[i]}^-| - 1} \sum_{m \in \mathcal{N}_{[i]}^- \setminus \{j\}} \dot{\boldsymbol{x}}_m[t]\|_2, \tag{8}$$

which compares each neighbor's gradient estimate to the average gradient in the rest of the neighborhood. The running average of the above metric is given by

$$\widetilde{S}_{ij}[t] = \frac{1}{t}\sum_{\tau=0}^{t} S_{ij}[\tau]. \tag{9}$$

Agents keep track of $\widetilde{S}_{ij}$ over time and, at a sufficiently large time $T \gg 0$, they sever ties with the agent in their neighborhood with the largest value of $\widetilde{S}_{ij}[T]$. Then, each agent adjusts its weights w_{ij}'s according to the conditions given in Section 2 (see below (3)) and resumes the updates with the reduced neighbor set.

To illustrate our approach, let us consider a parameter estimation problem where the observations $\boldsymbol{\kappa}_i$ are given by $\boldsymbol{\kappa}_i = \boldsymbol{x}_i + \boldsymbol{w}_i$, where $\boldsymbol{w}_i$ represents the measurement error. Assume that agent j is an attacker and its attack consists of modifying its local observation as $\boldsymbol{\kappa}_j^m = \boldsymbol{\kappa}_j + \boldsymbol{\beta}_j$. Then the local linear least square loss function may be written as:

$$\ell_j(\boldsymbol{x}_j) = \|\boldsymbol{x}_j - (\boldsymbol{\kappa}_j + \boldsymbol{\beta_j})\|_2^2, \forall j \in \mathcal{V}_m. \tag{10}$$

Considering node j as an attacker and substituting for the gradient of the loss function of node j in equation (8), we have

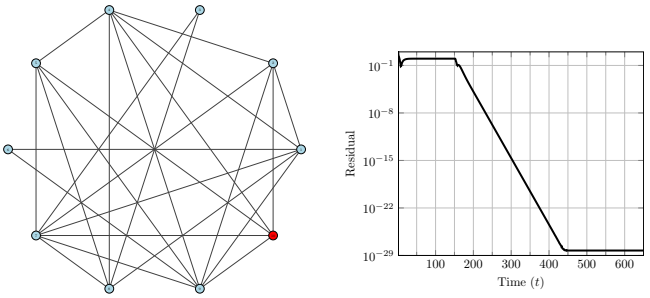
$$\begin{aligned} S_{ij}^m[t] &\approxeq \|2(\boldsymbol{x}_j[t] - (\boldsymbol{\kappa}_j + \boldsymbol{\beta_j})) - \frac{\sum_{k\in\mathcal{N}_{[i]}^-\setminus\{j\}} 2(\boldsymbol{x}_k[t] - \boldsymbol{\kappa}_k)}{|\mathcal{N}_{[i]}^-| - 1}\|_2 \\ &\leq \|2(\boldsymbol{x}_j[t] - \boldsymbol{\kappa}_j) - \frac{\sum_{k\in\mathcal{N}_{[i]}^-\setminus\{j\}} 2(\boldsymbol{x}_k[t] - \boldsymbol{\kappa}_k)}{|\mathcal{N}_{[i]}^-| - 1}\|_2 + 2\|\boldsymbol{\beta_j}\|_2 \\ &= S_{ij}^r[t] + 2\|\boldsymbol{\beta_j}\|_2. \end{aligned} \tag{11}$$

Now, consider the running average of the metric over time

$$\widetilde{S}_{ij}^m[t] = \frac{1}{t}\sum_{\tau=0}^{t} S_{ij}^m[\tau] \leq \frac{1}{t}\sum_{\tau=0}^{t} S_{ij}^r[t] + 2\|\boldsymbol{\beta_j}\|_2 = \widetilde{S}_{ij}^r[t] + 2\|\boldsymbol{\beta_j}\|_2.$$

Thus, there is a clear gap in the magnitude of the metric when a neighbor exhibits Byzantine tendencies ($\widetilde{S}_{ij}^m[t]$). Thus using the metric presented in equation (8), we can identify those neighbors that exhibit unusual behavior.

Consider the Erdős–Rényi network with $N = 10$ nodes and $p = 2.5\log(N)/N$ shown in Figure 2a. In this network, agent 10 (in color red) is a malicious node following the algorithm but feeding it false measurement data as described in equation (10). Let us suppose that the regular nodes keep a track of the metric for some time $T \gg 0$ sufficiently large such that the metric in equation (9) has gathered sufficient information about the behavior of the agents in the network. At time $t = T$, the regular nodes sever ties with their neighbor with the highest magnitude of $\widetilde{S}_{ij}[T]$. In Figure 3, we wait until $T = 150$ time steps and each agent severs ties with the outlier in its neighborhood. Thereafter, we see the self-healing properties of distributed consensus algorithm come into play and the regular nodes converge at the optimal consensus point. In Figure 2, we plot the residual, $r(t) = \frac{1}{N_r}\sum_{i\in\mathcal{V}_r} |\ell_i(\boldsymbol{x}_i(t)) - \ell_i(\boldsymbol{x}^*)|$, of the optimization problem on the y-axis over time on the x-axis.



(a) (b)

Fig. 2. (a) The Erdős–Rényi random network under consideration. (b) Residual of the optimization problem.

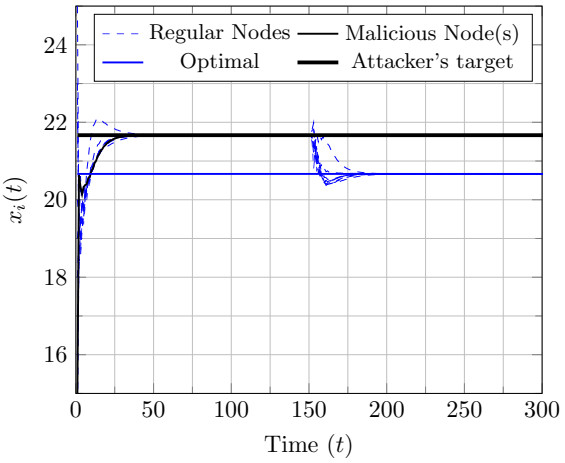


Fig. 3. Agents' iterates converge to the optimal point after attacker suppression.

5. CONCLUSION AND FUTURE WORK

We have proposed a detection strategy to counter Byzantine attacks in a system of agents trying to solve optimization problem in a distributed fashion. The strategy is independent of the network topology and the attack strategy, and makes use of a tracking a trend in gradient variance locally. Also, we have presented an upper bound for the strength of the attack which brought out a direct relationship between the attack strength and the gradient values for the attackers. Using this relationship, we hypothesized that it would be possible to detect the attackers if each agent kept a metric as a function of its neighbors' gradients and severs ties with the outliers, at a sufficiently large time T. Future work includes providing an estimate of the value of T. Work is also in progress to analyze the case of stealth attacks.

6. REFERENCES

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

[2] A. Nedić, A. Olshevsky, and M. G. Rabbat, "Network topology and communication-computation tradeoffs in decentralized optimization," *Proceedings of the IEEE*, vol. 106, no. 5, pp. 953–976, 2018.

[3] T. Tatarenko and B. Touri, "Non-convex distributed optimization," *IEEE Transactions on Automatic Control*, vol. 62, no. 8, pp. 3744–3757, 2017.

[4] P. Di Lorenzo and G. Scutari, "Next: In-network nonconvex optimization," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 2, no. 2, pp. 120–136, 2016.

[5] Y. Sun, G. Scutari, and D. Palomar, "Distributed nonconvex multiagent optimization over time-varying networks," in *Signals, Systems and Computers, 2016 50th Asilomar Conference on*. IEEE, 2016, pp. 788–794.

[6] Y. Tian, Y. Sun, B. Du, and G. Scutari, "Asysonata: Achieving geometric convergence for distributed asynchronous optimization," *arXiv preprint arXiv:1803.10359*, 2018.

[7] M. Pease, R. Shostak, and L. Lamport, "Reaching agreement in the presence of faults," *Journal of the ACM (JACM)*, vol. 27, no. 2, pp. 228–234, 1980.

[8] L. Lamport, R. Shostak, and M. Pease, "The byzantine generals problem," *ACM Transactions on Programming Languages and Systems (TOPLAS)*, vol. 4, no. 3, pp. 382–401, 1982.

[9] D. Dolev, "The byzantine generals strike again," *Journal of algorithms*, vol. 3, no. 1, pp. 14–30, 1982.

[10] D. Dolev, N. A. Lynch, S. S. Pinter, E. W. Stark, and W. E. Weihl, "Reaching approximate agreement in the presence of faults," *Journal of the ACM (JACM)*, vol. 33, no. 3, pp. 499–516, 1986.

[11] W. Ben-Ameur, P. Bianchi, and J. Jakubowicz, "Robust distributed consensus using total variation," *IEEE Transactions on Automatic Control*, vol. 61, no. 6, pp. 1550–1564, 2016.

[12] H. J. LeBlanc, H. Zhang, X. Koutsoukos, and S. Sundaram, "Resilient asymptotic consensus in robust networks," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 4, pp. 766–781, 2013.

[13] S. Sundaram and B. Gharesifard, "Distributed optimization under adversarial nodes," *IEEE Transactions on Automatic Control*, 2018.

[14] D. Yin, Y. Chen, K. Ramchandran, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," *arXiv preprint arXiv:1803.01498*, 2018.

[15] Q. Yan, M. Li, T. Jiang, W. Lou, and Y. T. Hou, "Vulnerability and protection for distributed consensus-based spectrum sensing in cognitive radio networks," in *INFOCOM, 2012 Proceedings IEEE*. IEEE, 2012, pp. 900–908.

[16] R. Gentz, S. X. Wu, H.-T. Wai, A. Scaglione, and A. Leshem, "Data injection attacks in randomized gossiping," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 2, no. 4, pp. 523–538, 2016.

[17] F. Pasqualetti, A. Bicchi, and F. Bullo, "Consensus computation in unreliable networks: A system theoretic approach," *IEEE Transactions on Automatic Control*, vol. 57, no. 1, pp. 90–104, 2012.

[18] R. Xin, C. Xi, and U. A. Khan, "FROST – Fast row-stochastic optimization with uncoordinated step-sizes," *EURASIP Journal on Advances in Signal Processing*, vol. 2019, no. 1, p. 1, 2019.

[19] D. Kempe, A. Dobra, and J. Gehrke, "Gossip-based computation of aggregate information," in *Foundations of Computer Science, 2003. Proceedings. 44th Annual IEEE Symposium on*. IEEE, 2003, pp. 482–491.

[20] F. Bénézit, V. Blondel, P. Thiran, J. Tsitsiklis, and M. Vetterli, "Weighted gossip: Distributed averaging using non-doubly stochastic matrices," in *Information theory proceedings (isit), 2010 ieee international symposium on*. IEEE, 2010, pp. 1753–1757.

[21] C. Xi and U. A. Khan, "Dextra: A fast algorithm for optimization over directed graphs," *IEEE Transactions on Automatic Control*, vol. 62, no. 10, pp. 4980–4993, 2017.

[22] C. Xi, R. Xin, and U. A. Khan, "Add-opt: Accelerated distributed directed optimization," *IEEE Transactions on Automatic Control*, vol. 63, no. 5, pp. 1329–1339, 2018.

[23] C. Xi and U. A. Khan, "Distributed subgradient projection algorithm over directed graphs," *IEEE Transactions on Automatic Control*, vol. 62, no. 8, pp. 3986–3992, 2017.