Efficient Byzantine-Resilient Decentralized Learning: Sparse M-Estimation with Robust Aggregation Techniques

Chris Junchi Li

Department of Electrical Engineering and Computer Sciences$^{\diamond}$
University of California, Berkeley

October 2, 2024

Abstract

This paper addresses the challenge of robust sparse M-estimation in decentralized networks under Byzantine attacks. In a decentralized learning framework, some nodes may behave arbitrarily due to system failures, cyber-attacks, or malicious interference, compromising the overall model accuracy. To mitigate these threats, we propose a Byzantine-resilient algorithm, which combines gradient tracking with a proximal algorithm, and utilizes robust aggregation rules such as Median, Trimmed Mean, and Krum. These techniques ensure that the global gradient is accurately estimated despite the presence of Byzantine nodes, and enforce sparsity in high-dimensional settings. We theoretically demonstrate that our method achieves linear convergence under mild assumptions, and we validate its finite-sample performance through extensive numerical experiments. Our approach provides a resilient and efficient solution to decentralized learning problems in hostile environments.

Keywords: Byzantine robustness, decentralized learning, sparse M-estimation, gradient tracking, robust aggregation, proximal algorithm.

1 Introduction

The increasing complexity of modern data systems has driven the adoption of decentralized learning frameworks, where data and computations are distributed across a network of nodes without relying on a central server. This decentralized architecture offers significant advantages, such as enhanced privacy, reduced communication costs, and improved fault tolerance. However, these systems are vulnerable to adversarial behaviors, particularly Byzantine attacks, where some nodes behave arbitrarily or maliciously due to system failures or intentional interference. Byzantine nodes can severely compromise the learning process, posing significant challenges to maintaining robustness and accuracy.

In high-dimensional settings, where only a few relevant variables are present, these challenges are further amplified. Sparse M-estimation is a popular method for variable selection and model estimation in such contexts, but achieving robust sparse M-estimation in decentralized networks exposed to Byzantine faults remains an open problem. Existing approaches often fail to address both the high-dimensional nature of the data and the unpredictability of Byzantine nodes, making this an urgent area of research.

In this paper, we propose a novel Byzantine-resilient decentralized sparse M-estimation algorithm. Our method combines gradient tracking with proximal algorithms and employs robust aggregation techniques—such as Median, Trimmed Mean, and Krum—to mitigate the impact of Byzantine nodes. The use of L_1 regularization ensures sparsity in high-dimensional settings, and

both theoretical and empirical results demonstrate that the algorithm achieves linear convergence under mild conditions, making it a practical solution for decentralized learning in adversarial environments.

Centralized and decentralized learning with Byzantine attacks With the advancement of science and technology, data are stored independently in many different nodes/devices/sites. For example, the Observational Health Data Sciences and Informatics consortium has over 82 clinical databases around the world; Walmart stores generate data from different locations around the world; the clinical information of patients is stored at different hospitals and so on.

Due to the limitation of storage, computing capability and privacy issues, the traditional methods by directly pooling the entire data in one central node/machine/computer are not practical. To handle these issues, several federated learning methods have drawn attention. In a centralized distributed system, data are partitioned across multiple nodes and there is always a central node to combine the information from other nodes. [McMahan et al., 2017] proposed FederatedAveraging algorithm by aggregating local estimators in the central node and then communicating with local nodes, which has been extended to robust optimization [Blanchard et al., 2017, Fang et al., 2020], heterogeneous data [Li et al., 2020], multi-task learning [Smith et al., 2017], differential privacy [Cheu et al., 2019, Wei et al., 2020] and many other problems. Unfortunately, the centralized learning is fragile. If the central node is conquered, the attacker can arbitrarily communicate with every node [Bellet et al., 2018], which may lead to privacy problems or further attacks. Recently, [Colin et al., 2016] introduced decentralized learning, i.e., there is no central node to train the global model, and it is more general and practical than the centralized learning methods. For example, [Nedic, 2020] proposed decentralized gradient descent; [Richards & Rebeschini, 2019] and [Richards et al., 2020] considered homogeneous distributions, while [Yuan et al., 2016, Lian et al., 2017] and [Richards et al., 2020] focused on the doubly stochastic matrix networks. [Wu et al., 2023a] proposed the network gradient descent method under some weaker assumptions and then proved the numerical convergence with a statistical rate.

It should be pointed out that the majority of existing centralized/decentralized learning methods are based on fully normal nodes. However, in learning control [Zhuang et al., 2022, Zhou et al., 2022, Wang et al., 2023], fault diagnosis [Tao et al., 2023] and many other related fields, due to data corruption, device malfunctioning, system breakdowns, cyber attacks, malicious attacks and some other unexpected behaviors (characterized as Byzantine attacks), there is a small fraction of nodes that behave arbitrarily [Lamport et al., 1982]. These abnormal nodes, called Byzantine nodes, arbitrarily deviate from their intended behaviors, while their number and identities are unknown.

Main results In this paper, a Byzantine-robust method is proposed to solve sparse M-estimation over decentralized networks in Section 3. In order to propose a novel algorithm for the general loss function, we apply a technique called *square loss transformation* [Chen et al., 2020, Tu et al., 2023] in Section 4. To the best of our knowledge, this problem has not previously been investigated. The main contributions of this paper are summarized as follows.

(i) The *square loss Byzantine-robust network* (SLBRN) algorithm is constructed to obtain sparse M-estimation with Byzantine attacks over decentralized networks. The SLBRN algorithm combines the gradient tracking [Di Lorenzo & Scutari, 2016, Xu et al., 2017] and proximal algorithm [Rockafellar, 1976, Parikh & Boyd, 2014] to ensure convergence. In particular, the gradient tracking is applied to efficiently estimate the global gradient and the proximal algorithm is used to achieve the fast convergence rates and sparsity by adding strictly convex

and quadratic regularization penalties. All these techniques are combined to account for the Byzantine nodes and improve the decentralized sparse estimation simultaneously. Moreover, it uses several Byzantine-robust aggregation rules to resist against Byzantine attacks. Theoretically, it is shown that our proposed estimators converge with linear rates under some mild conditions, which can be achieved in most general situations.

(ii) For the general M-estimation, we propose the *square loss transformation Byzantine-robust network* (SLBRN) algorithm through applying the square loss transformation to approximate the original loss function and then aggregating the gradient information. Instead of solving the original M-estimation problem, this transformation leads to a standard LASSO type problem, which can be optimized by R packages.

(iii) Empirically, the finite-sample performance is investigated by conducting various experiments considering different regression models, Byzantine attacks, error types, number of Byzantine nodes, local sample size and so on. The numerical results together with a real data application corroborate the theoretical findings and show that our proposed estimators perform well in practice.

Contributions The main contributions of this paper are as follows:

- We propose a novel Byzantine-resilient decentralized sparse M-estimation algorithm that combines gradient tracking, proximal methods, and robust aggregation rules to ensure accurate gradient estimation and sparsity in high-dimensional settings.
- We provide a rigorous theoretical analysis that demonstrates linear convergence of the algorithm under mild conditions, even in the presence of Byzantine nodes.
- Extensive numerical experiments, including a real-world application to the Communities and Crime dataset, validate the robustness and computational efficiency of our method in various settings, particularly under Byzantine attack scenarios.

Organization The remainder of this paper is organized as follows. In Section 2, we review the related work on Byzantine-resilient decentralized learning. Section 3 presents our proposed algorithm, along with a detailed theoretical analysis of its convergence rates. In Section 4, we apply the square loss transformation to obtain Byzantine-robust M-estimation in practice. Section 5 concludes the paper with a discussion of future research directions.

Notation Throughout this paper, the following notations are necessary to be introduced. For a set A, define its cardinality as $\mathrm{Card}(A)$. For any vector $\boldsymbol{v} = (v_1, \ldots, v_p)^\top \in \mathbb{R}^p$, define $\|\boldsymbol{v}\|_l = \left(\sum_{k=1}^p v_k^l\right)^{1/l}$ as its L_l-norm for $1 \leq l \leq \infty$ and $[\boldsymbol{v}]_k$ as its k-th coordinate for $1 \leq k \leq p$. Define the absolute value of $\boldsymbol{v}$ as coordinate-wise absolute value, i.e., $|\boldsymbol{v}| = (|v_1|, \ldots, |v_p|)^\top$. For two vectors $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^p$, define inner product $\langle \boldsymbol{u}, \boldsymbol{v} \rangle = \boldsymbol{u}^\top \boldsymbol{v}$. For a function $f(\boldsymbol{v})$, define $\nabla f(\boldsymbol{v})$ as its first derivative with respect to $\boldsymbol{v}$.

2 Related work

In literature, several Byzantine-robust methods are proposed to hedge against Byzantine attacks and the key idea is to find a robust statistic to aggregate the information. However, most of them

ignored high-dimensionality and could not obtain sparse estimators.

2.1 Byzantine-robust centralized learning

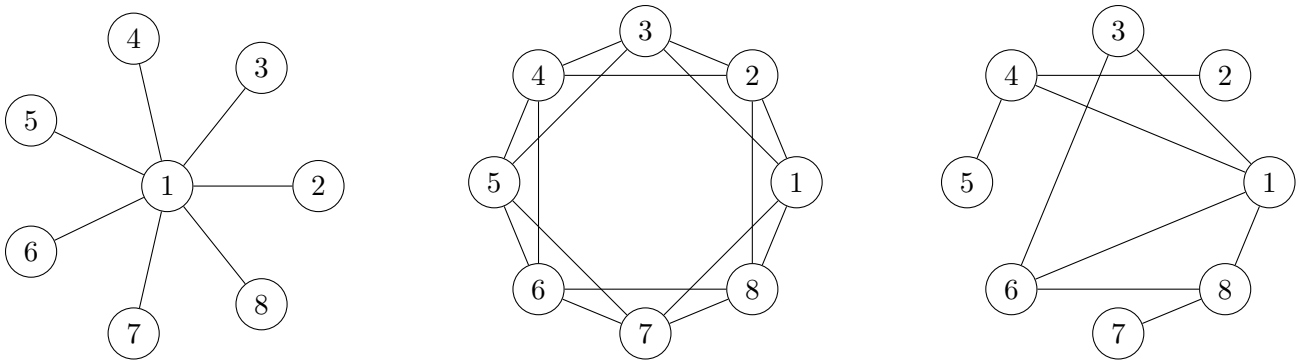
For the centralized settings, the median [Xie et al., 2018, Yin et al., 2018], trimmed mean [Yin et al., 2018], Krum [Blanchard et al., 2017], RFA [Pillutla et al., 2022], centered clipping [Karimireddy et al., 2021] based methods and their variants are commonly used in robust statistics [Blanchard et al., 2017, Xie et al., 2018, Tu et al., 2023, Zhou et al., 2023, Zhu et al., 2023, Baruch et al., 2019, Lecué & Lerasle, 2020, Fang et al., 2022]. Nevertheless, these methods relied on prior knowledge of several network-wide parameters, including the subset of Byzantine nodes within the networks, and also required the maximum cumulative mixing weight associated with the Byzantine nodes to be impractically small [Fang et al., 2022]. To solve high-dimensional optimization problems over a network of nodes, [Sun et al., 2022] proposed a novel distributed algorithm called NetLASSO with linear convergence and statistical guarantees. Unfortunately, the NetLASSO focuses on linear regression and assumes all nodes can work normally, while it fails in the presence of the Byzantine attacks.

In practice, the central node may become the bottleneck of the system: (i) its failure would make the entire system fail; (ii) its limited bandwidth sets an upper boundary on the communication cost (i.e., the number of nodes). To cope with these issues, a natural idea is to eliminate the central node and consider a decentralized network, wherein each node is connected to a subset of the other nodes. It is worthwhile pointing out that the algorithms designed for the centralized distributed learning fail in the decentralized networks, which inspires a surge of studies producing some novel algorithms suitable for decentralized networks; see [Sayed, 2014, Nedić et al., 2018, Nedic & Ozdaglar, 2009] and the references therein.

2.2 Byzantine-robust decentralized learning

Recently, several Byzantine-robust algorithms based on the decentralized networks have been proposed. For example, [Guo et al., 2021] applied stochastic gradient descent and defined a two-stage method to filter out the potential Byzantine attacks. [Xu & Huang, 2022] constructed a similarity-based reweighting method for collaborative learning. [Elkordy et al., 2022] proposed a fast and computationally efficient Byzantine-resilient approach. [He et al., 2022] studied a self-centered clipping algorithm to aggregate the received local models. Based on the distributed gradient descent and its stochastic variants, [Yang & Bajwa, 2019, Su & Vaidya, 2020, Fang et al., 2022, Hou et al., 2022, Sun et al., 2022, Ye et al., 2024] provided the Byzantine-resilient decentralized optimization and stochastic optimization with some robust aggregation rules. [Kuwaranancharoen & Sundaram, 2023] provided a general Byzantine-resilient algorithmic framework and established the linear convergence. [Peng et al., 2023] applied the stochastic subgradient method to allow possible outliers and established its sublinear convergence. [Hu et al., 2023] investigated a unified analysis based on proximal stochastic optimization with variance reduction. [Ye et al., 2024] considered privacy preservation and Byzantine-robustness together.

However, most of these existing studies have some deficiencies in practice. For example, (i) the convergence rates were not achieved even if they imposed restrictive assumptions on networks [Yang & Bajwa, 2019, Chen et al., 2021, Fang et al., 2022, Hou et al., 2022, Wu et al., 2023b, Ye et al., 2024]; (ii) it is hard to check the stringent assumptions on target loss function directly [Peng et al., 2021, Elkordy et al., 2022, He et al., 2022, Hu et al., 2023, Sun et al., 2022]; see Section 3.4 for more details; (iii) they ignored the high-dimensionality issue, which could lead to overfitting and large variance of the estimator [Guo et al., 2021, Konan et al., 2022, Xu & Huang, 2022,



(a) centralized graph (b) ring graph (c) random graph

Figure 1: Examples of some commonly used network structures.

Peng et al., 2023, Ye et al., 2024].

In summary, most of these existing studies considered the Byzantine-robust centralized/decentralized learning problems, while most of them imposed restrictive assumptions on the data distribution patterns, network structures, target loss functions and nonsparse estimators. This paper aims at overcoming these difficulties and raising a Byzantine-robust decentralized algorithm with provably linear convergence for high-dimensional covariates.

3 Sparse Byzantine-robust M-estimation

3.1 Problem formulation

We first introduce a decentralized network of m nodes by modeling it as a general undirected graph $\mathcal{G} := (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, \ldots, m\}$ is the set of m nodes, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges: $(i, j) \in \mathcal{E}$ if and only if there exists a communication link between the nodes i and j. Some commonly used graphs for the network structures are given in the following Fig. 1 with $m = 8$. Denote $\mathcal{N}_j = \{i \in \mathcal{V} : (i, j) \in \mathcal{E}\} \cup \{j\}$ as the neighbors of the node j with its cardinality $m_j := \text{Card}(\mathcal{N}_j)$. Without loss of generality, assume each node j collects n samples $\{Y_{ij}, \boldsymbol{X}_{ij}\}_{i=1}^n$. For any sample $\boldsymbol{Z} := (\boldsymbol{Y}, \boldsymbol{X})$ from the sample space, the true parameter satisfies

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta} \in \boldsymbol{\Theta}}{\arg\min} \ \{F(\boldsymbol{\theta}) := \mathbb{E}[f(\boldsymbol{\theta}; \boldsymbol{Z})]\} \tag{1}$$

where $f(\boldsymbol{\theta}; \boldsymbol{Z})$ is a general loss function of sample $\boldsymbol{Z}, \boldsymbol{\theta}^* \in \boldsymbol{\Theta} \subseteq \mathbb{R}^p$ is the target parameter with sparsity, the parameter space $\boldsymbol{\Theta}$ is convex and compact with diameter D, i.e., $\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|_2 \leq D$ for any $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \in \boldsymbol{\Theta}$. Since we focus on high-dimensional setting, a standard way to estimate $\boldsymbol{\theta}^*$ in model (1) is to minimize the empirical loss function with LASSO penalty [Tibshirani, 1996, Zhao & Yu, 2006, Hastie et al., 2015] as follows,

$$\widehat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta} \in \boldsymbol{\Theta}}{\arg\min} \ \frac{1}{m} \sum_{j=1}^{m} L_j(\boldsymbol{\theta}) + \lambda_1 \|\boldsymbol{\theta}\|_1 \tag{2}$$

where $L_j(\boldsymbol{\theta}) := \sum_{i=1}^n f(\boldsymbol{\theta}; \boldsymbol{Z}_{ij})/n$ is the loss of the node j and λ_1 is the regularization parameter. However, it is impractical to directly solve the optimization problem (2) due to the communication cost or privacy problems under the network setting.

Throughout the remainder of this paper, denote $\mathcal{N} \subseteq \mathcal{V}$ and $\mathcal{B} \subseteq \mathcal{V}$ as the sets of nonfaulty and Byzantine nodes with $\mathcal{V} = \mathcal{N} \cup \mathcal{B}$ and $b := \mathrm{Card}(\mathcal{B})$. For any node j, define the sets of Byzantine neighbors as $\mathcal{B}_j = \{i : i \in \mathcal{N}_j \cap \mathcal{B}\}$ with $b_j := \mathrm{Card}(\mathcal{B}_j)$ and consider that an α_j fraction of $\mathcal{N}_j$ are Byzantine nodes, i.e., $\alpha_j = b_j/m_j$. As in [Yin et al., 2018], we consider the nodes in the network communicate with each other using some predefined protocol. Unfortunately, the Byzantine nodes can violate this protocol and send arbitrary information to their neighbors; in particular, they may have complete knowledge of the system.

In order to get a Byzantine-robust algorithm, the following three aggregation rules are usually applied, while other choices could also be taken into consideration. For convenience, denote the inputs of these aggregation rules as $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_S \in \mathbb{R}^p$ and $S \in \mathbb{N}^*$.

(1) **Median.** It uses the coordinate-wise median as the aggregation rule, i.e., the k-th coordinate takes the form: median $\{[\boldsymbol{\theta}_1]_k, [\boldsymbol{\theta}_2]_k, \ldots, [\boldsymbol{\theta}_S]_k\}$, where median is the usual (one-dimensional) median function.

(2) **Trimmed mean.** It uses the coordinate-wise trimmed mean as the aggregation rule. For a pre-given fraction $\beta \in [0, 1/2)$, the k-th coordinate of the coordinate-wise β-trimmed mean takes the form: $\sum_{x \in U_k} x/[(1-2\beta)S]$, where U_k is a subset of $\{[\boldsymbol{\theta}_1]_k, [\boldsymbol{\theta}_2]_k, \ldots, [\boldsymbol{\theta}_S]_k\}$ obtained by removing the largest and smallest β fraction of its elements. It should be pointed out that **Trimmed mean** requires each node to have at least $2\beta S + 1$ neighbors and the elements from different neighbors may survive the screening at different coordinates.

(3) **Krum.** For $i \neq j$, let $i \xrightarrow{k} j$ denote that $[\boldsymbol{\theta}_j]_k$ belongs to $S - b_i - 2$ closest value to $[\boldsymbol{\theta}_i]_k$. The k-th coordinate of the coordinatewise Krum is $[\boldsymbol{\theta}_{i_k}]_k$, where $i_k = \arg\min_{i \in \{1,2,\ldots,S\}} \sum_{i \to j} \left([\boldsymbol{\theta}_i]_k - [\boldsymbol{\theta}_j]_k\right)^2$ such that the **Krum** requires node i to have at least $b_i + 3$ neighbors.

Among these aggregation methods, the **Trimmed mean** and **Krum** rules need prior information on the number of nodes, while the **Median** rule neither requires explicit knowledge of $\{b_j\}_{j \in \mathcal{V}}$ nor imposes an explicit constraint on the minimum number of neighbors of each node.

3.2 SLBRN method

Our aim is to decentralize the gradient descent method and obtain a sparse M-estimation in the presence of the Byzantine attacks and high-dimensional covariates. The challenge is how to accurately estimate the global gradient at each local node using the information of neighbors while preserving the linear convergence. Motivated by [Sun et al., 2022], we propose to apply the gradient tracking and proximal algorithm with a pre-determined aggregation rule $\mathcal{A}$. To be specific, at the t-th iteration, each node j computes $\boldsymbol{\theta}_j^t = \mathcal{A}(\{\boldsymbol{\theta}_i^{t-\frac{1}{2}}\}_{i \in \mathcal{N}_j})$ from all neighbors $i \in \mathcal{N}_j$ and then calculates

$$\boldsymbol{g}_j^t = G_j^{t-1} + \nabla L_j(\boldsymbol{\theta}_j^t) - \nabla L_j(\boldsymbol{\theta}_j^{t-1})$$

where the gradient tracking is applied to efficiently estimate the global gradient and $\boldsymbol{G}_j^t$ is obtained by aggregating $\{\boldsymbol{g}_i^t\}_{i \in \mathcal{N}_j}$ to play the role of the global gradient $\sum_{j=1}^m L_j(\boldsymbol{\theta})/m$. Finally, each node j minimizes the estimated loss function $(\boldsymbol{G}_j^t)^\top \boldsymbol{\theta}_j$ by adding strictly convex and quadratic regularization penalties as follows:

$$\boldsymbol{\theta}_j^{t+\frac{1}{2}} = \underset{\boldsymbol{\theta}_j \in \boldsymbol{\Theta}}{\arg\min} \left\{ (\boldsymbol{G}_j^t)^\top \boldsymbol{\theta}_j + \lambda \left\|\boldsymbol{\theta}_j\right\|_1 + \frac{\gamma}{2} \left\|\boldsymbol{\theta}_j - \boldsymbol{\theta}_j^t\right\|_2^2 \right\}$$

Algorithm 1 Square Loss Byzantine-Robust Network (SLBRN)

Input: Data $\boldsymbol{Z}_N$, $\boldsymbol{\theta}_j^0 = \boldsymbol{\theta}_j^{\frac{1}{2}} = \boldsymbol{\theta}^0$ with a pre-determined initial vector $\boldsymbol{\theta}^0$ and $\boldsymbol{G}_j^0 = \nabla L_j(\boldsymbol{\theta}_j^0)$ for $j = 1, \ldots, m$, penalty parameters λ and γ, aggregation rule $\mathcal{A}$, the number of iterations T

1: **for** $t = 1, \ldots, T$ **do**

2: **Step 1: Aggregation**

3: Each node $j \notin \mathcal{B}$ receives $\boldsymbol{\theta}_i^{t-\frac{1}{2}}$ from all neighbors $i \in \mathcal{N}_j$ and computes

$$\boldsymbol{\theta}_j^t = \mathcal{A}(\{\boldsymbol{\theta}_i^{t-\frac{1}{2}}\}_{i \in \mathcal{N}_j})$$

where $\boldsymbol{\theta}_i^{t-\frac{1}{2}}$ can be arbitrary if $i \in \mathcal{B}_j$

4: Each node $j \notin B$ computes

$$\boldsymbol{g}_j^t = G_j^{t-1} + \nabla L_j(\boldsymbol{\theta}_j^t) - \nabla L_j(\boldsymbol{\theta}_j^{t-1}) \tag{3}$$

5: Each node $j \notin \mathcal{B}$ receives $\boldsymbol{g}_i^t$ from all neighbors $i \in \mathcal{N}_j$ and computes

$$\boldsymbol{G}_j^t = \mathcal{A}(\{\boldsymbol{g}_i^t\}_{i \in \mathcal{N}_j})$$

where $\boldsymbol{g}_i^t$ can be arbitrary if $i \in \mathcal{B}_j$

6: **Step 2: Optimization**

7: Each node $j \notin \mathcal{B}$ solves

$$\boldsymbol{\theta}_j^{t+\frac{1}{2}} = \underset{\boldsymbol{\theta}_j \in \boldsymbol{\Theta}}{\arg\min} \left\{ (\boldsymbol{G}_j^t)^\top \boldsymbol{\theta}_j + \lambda \left\| \boldsymbol{\theta}_j \right\|_1 + \frac{\gamma}{2} \left\| \boldsymbol{\theta}_j - \boldsymbol{\theta}_j^t \right\|_2^2 \right\}$$

8: **end for**

Output: $\boldsymbol{\theta}_j^T$ $(j \notin \mathcal{B})$

where the L_1-penalty $\lambda\|\boldsymbol{\theta}_j\|_1$ will lead to the sparse estimator and $\gamma\|\boldsymbol{\theta}_j - \boldsymbol{\theta}_j^t\|_2^2/2$ is introduced by the proximal algorithm to speed the convergence rates. The resulting algorithm is formally introduced in Algorithm 1.

In Algorithm 1, we need to specify the initial vector $\boldsymbol{\theta}^0$, penalty parameters λ and γ, aggregation rule $\mathcal{A}$ and number of iterations T at the beginning. In Section A, extensive experimental results under different experimental settings are given to show the good performance of the proposed algorithm.

3.3 Complexity analysis

The complexity of the Algorithm 1 consists of the following four parts: (i) aggregating neighbors' parameters by some specific rule, (ii) computing gradients with local sample, (iii) aggregating neighbors' gradients by some specific rule, and (iv) optimizing penalized loss functions.

It is worth mentioning that the complexity of computing gradients and aggregating neighbors' gradients/parameters relies on the loss functions and aggregation rules, respectively, while the complexity of optimizing step is $O(d)$. For example, for square loss function, the complexity of computing gradients/parameters is $O\left(nd^2\right)$. Let $m_{\max} = \max_{j \in \mathcal{N}} m_j$, the complexities of **Median**, **Trimmed mean**, **Krum** are $O\left(m_{\max} d\right), O\left(m_{\max} d\right)$ and $O\left(m_{\max} d^2\right)$, respectively.

Therefore, the complexity of the Algorithm 1 can be achieved with a determined aggregation rule. For instance, for square loss function and **Median** aggregation rule, the complexity of our proposed Algorithm 1 is $O\left(Td\left(nd+m_{\max}\right)\right)$ given the number of iterations T. In terms of complexity, our proposed algorithm is comparable to these popular Byzantine-robust decentralized methods [Yang & Bajwa, 2019, Fang et al., 2022, Peng et al., 2023, Sun et al., 2022] introduced in Section 2.2.

3.4 Convergence analysis

In this section, the main theoretical results about convergence are presented. Before showing the theorems about convergence analysis, the definition of absolute skewness and some assumptions are needed.

Definition 1 (Absolute skewness)**.** *For a random variable $x \in \mathbb{R}$, define its absolute skewness as $\xi(x) := \mathbb{E}\left[|x-\mathbb{E}[x]|^3\right] / \mathrm{Var}^{3/2}(x)$ with $\mathrm{Var}(x) = \mathbb{E}\left[(x-\mathbb{E}[x])^2\right]$. For a random vector $x = (x_1,\ldots,x_p)^\top \in \mathbb{R}^p$, define the vector of absolute skewness of each coordinate of $\boldsymbol{x}$ as its absolute skewness, i.e., $\xi(x) := (\xi(x_1),\ldots,\xi(x_p))^\top$.*

The absolute skewness is used to quantify the asymmetry or lack of symmetry in a dataset. We make the following assumptions that are common in the analysis of decentralized optimization to provide theoretical guarantees for our proposed Algorithm 1. Throughout we assume that loss functions $f(\boldsymbol{\theta};\boldsymbol{Z})$ and $F(\boldsymbol{\theta})$ are smoothed.

Assumption 1 (Smoothness of f and F)**.** *For any sample $\boldsymbol{Z}$, the function $f(\boldsymbol{\theta};\boldsymbol{Z})$ is L-smooth and its partial derivative of $f(\boldsymbol{\theta};\boldsymbol{Z})$ with respect to the k-th coordinate, denoted by $\partial_k f(\boldsymbol{\theta};\boldsymbol{Z})$, is L_k-Lipschitz for each $k \in \{1,2,\ldots,p\}$. Moreover, the population loss function $F(\boldsymbol{\theta})$ is L_F-smooth and λ_F-strongly convex. That is, $\forall \boldsymbol{\theta}_1,\boldsymbol{\theta}_2 \in \mathbb{R}^p$ (i) $\|\nabla f(\boldsymbol{\theta}_1;\boldsymbol{Z}) - \nabla f(\boldsymbol{\theta}_2;\boldsymbol{Z})\|_2 \le L\|\boldsymbol{\theta}_1-\boldsymbol{\theta}_2\|_2$ (ii) $\|\partial_k f(\boldsymbol{\theta}_1;\boldsymbol{Z}) - \partial_k f(\boldsymbol{\theta}_2;\boldsymbol{Z})\| \le L_k\|\boldsymbol{\theta}_1-\boldsymbol{\theta}_2\|_2$ (iii) $\|\nabla F(\boldsymbol{\theta}_1) - \nabla F(\boldsymbol{\theta}_2)\|_2 \le L_F\|\boldsymbol{\theta}_1-\boldsymbol{\theta}_2\|_2$ and $F(\boldsymbol{\theta}_1) - F(\boldsymbol{\theta}_2) - \langle \nabla F(\boldsymbol{\theta}_2), \boldsymbol{\theta}_1-\boldsymbol{\theta}_2\rangle \ge \frac{\lambda_F}{2}\|\boldsymbol{\theta}_1-\boldsymbol{\theta}_2\|_2$*

In addition, when $F(\boldsymbol{\theta})$ is convex, we assume $\boldsymbol{\theta}^*$ is not only the minimizer of $F(\boldsymbol{\theta})$ in $\boldsymbol{\Theta}$, but also the minimizer of $F(\boldsymbol{\theta})$ in $\mathbb{R}^p$.

Assumption 2 (Minimizer in $\boldsymbol{\Theta}$)**.** *$F(\boldsymbol{\theta})$ is convex, both $\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}\in\Theta} F(\boldsymbol{\theta})$ and $\nabla F(\boldsymbol{\theta}^*) = \mathbf{0}$ hold.*

Then we impose the assumptions that the gradient of loss function $f(\boldsymbol{\theta};\boldsymbol{Z})$ has bounded variance, skewness and is sub-exponential.

Assumption 3 (Bounded variance and skewness of gradient)**.** *For any $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, $\mathrm{Var}(\nabla f(\boldsymbol{\theta};\boldsymbol{Z})) \le V^2$, $\|\xi(\nabla f(\boldsymbol{\theta};\boldsymbol{Z}))\|_\infty \le S$ for some $V>0$ and $S>0$.*

Assumption 4 (Sub-exponential gradient)**.** *For any $k \in \{1,2,\ldots,p\}$ and $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, $[\nabla f(\boldsymbol{\theta};\boldsymbol{Z})]_k$ is v-sub-exponential. That is, $\mathbb{E}\left[\exp\left\{\kappa([\nabla f(\boldsymbol{\theta};\boldsymbol{Z})]_k - \mathbb{E}[\nabla f(\boldsymbol{\theta};\boldsymbol{Z})]_k)\right\}\right] \le \exp\left(v^2\kappa^2/2\right)$ for any $|\kappa| < 1/v$.*

Assumption 3 and Assumption 4 are standard and satisfied in many learning problems. For example, Propositions 1 and 2 in [Yin et al., 2018] considered that the gradient of the loss function $f(\boldsymbol{\theta};\boldsymbol{Z})$ had bounded variance, each coordinate of $\nabla f(\boldsymbol{\theta};\boldsymbol{Z})$ had bounded absolute skewness and was sub-exponential, respectively, for a linear regression problem. The following Assumption 5 gives some mild restrictions on the fractions of Byzantine neighbors and was typically used in [Yin et al., 2018] and [Zhou et al., 2023].

Assumption 5. *Let* $\widehat{L} := \left(\sum_{k=1}^{p} L_k^2\right)^{1/2}$ *with* $\left\{L_k^2\right\}_{k=1}^{p}$ *given in Assumption 1, the fraction* α_j *of Byzantine nodes in* $\mathcal{N}_j$ *satisfies*

$$\alpha_j + \sqrt{\frac{p\log(1+nm_j\widehat{L}D)}{m_j\left(1-\alpha_j\right)}} + 0.4748\frac{S}{\sqrt{n}} \leq \frac{1}{2} - \delta \tag{4}$$

for any $j \in \{1, 2, \ldots, m\}$ *and some* $\delta > 0$.

Define $\rho := L_F/\left(L_F + \lambda_F\right) \in (0,1)$. Based on these assumptions, the following theorems establish the linear convergence of two variants of the proposed framework with some pre-determined aggregation rules.

Theorem 1 (Convergence with **Median** aggregation)**.** *Consider the aggregation rule in Algorithm 1 is* ***Median****. Under Assumption 1, Assumption 2 and Assumption 3 and 5 , let* $\gamma = L_F$ *and* $\lambda = C_1\sqrt{p/n}$ *with some positive constant* C_1*, after* T *iterations, with probability at least* $1 - \frac{4p}{(1+nm\widehat{L}D)^p}$*, we have*

$$\left\|\boldsymbol{\theta}_j^T - \boldsymbol{\theta}^*\right\|_2 \leq \rho^T\left\|\boldsymbol{\theta}_j^0 - \boldsymbol{\theta}^*\right\|_2 + 2L_F^{-1}p\left(\Delta_j + \chi\right) \tag{5}$$

where $\chi := O_p(Dpn^{-1/2})$, $\Delta_j := O_p\left(C_\delta V\left((\alpha_j+S)n^{-1/2} + \sqrt{p\log(nm_j\widehat{L}D)/(nm_j)}\right)\right)$, *and* $C_\delta := \sqrt{2\pi}\exp\left(\frac{1}{2}\left(\Phi^{-1}(1-\delta)\right)^2\right)$ *with* $\Phi^{-1}(\cdot)$ *being the inverse of the cumulative distribution function of the standard Gaussian distribution* $\Phi(\cdot)$.

Remark. The detailed proof of Theorem 1 is given in Appendix D.1. Theorem 1 shows the linear convergence of $\boldsymbol{\theta}_j$ for $j \in \mathcal{V}$ by using **Median** aggregation. Specially, the factor C_δ is a function of δ, e.g., $C_\delta \approx 4$ if $\delta = 1/6$. Notice that $\log(1-x) \leq -x$ for $x \in (0,1)$, Theorem 1 guarantees that after $T \geq \frac{L_F+\lambda_F}{\lambda_F}\log\left(\frac{\lambda_F}{2\Delta_j}\|\boldsymbol{\theta}_j^0 - \boldsymbol{\theta}^*\|_2\right)$ iterations, with high probability we can obtain a solution on node j as $\widehat{\boldsymbol{\theta}}_j = \boldsymbol{\theta}_j^T$ with an error $\|\widehat{\boldsymbol{\theta}}_j - \boldsymbol{\theta}^*\|_2 \leq 2L_F^{-1}\left(2\Delta_j + \chi\right)$.

Theorem 2 (Convergence with **Trimmed mean** aggregation)**.** *Consider the aggregation rule in Algorithm 1 is* ***Trimmed mean*** *with some pre-given fractions* $\beta_j, j \in \mathcal{V}$*. Suppose that Assumption 1, Assumption 2 and Assumption 4 hold, and for any* $j \in \mathcal{V}, \alpha_j \leq \beta_j \leq 1/2 - \widetilde{\delta}$ *for some* $\widetilde{\delta} > 0$*. Let* $\gamma = L_F$ *and* $\lambda = C_2\sqrt{p/n}(C_2)$ *is a positive constant), after* T *iterations, then with probability at least* $1 - \frac{4p}{(1+nm\widehat{L}D)^p}$ *we have*

$$\left\|\boldsymbol{\theta}_j^T - \boldsymbol{\theta}^*\right\|_2 \leq \rho^T\left\|\boldsymbol{\theta}_j^0 - \boldsymbol{\theta}^*\right\|_2 + 2L_F^{-1}(\widetilde{\Delta}_j + \chi) \tag{6}$$

where χ *is defined as in the Theorem 1 and* $\widetilde{\Delta}_j := O_p\left(\frac{vp}{\widetilde{\delta}}n^{-1/2}\left(\beta + \frac{1}{\sqrt{m_j}}\right)\sqrt{\log(nm_j\widehat{L}D)}\right)$.

Remark. The detailed proof of Theorem 2 is given in Appendix D.2. Theorem 2 shows the linear convergence of $\boldsymbol{\theta}_j$ for $j \in \mathcal{V}$ with **Trimmed mean** aggregation. Moreover, Theorem 2 guarantees that after $T \geq \frac{L_F+\lambda_F}{\lambda_F}\log\left(\frac{\lambda_F}{2\widetilde{\Delta}_j}\|\boldsymbol{\theta}_j^0 - \boldsymbol{\theta}^*\|_2\right)$ iterations, with high probability we can obtain a solution on node j as $\widetilde{\boldsymbol{\theta}}_j = \boldsymbol{\theta}_j^T$ satisfying $\|\widetilde{\boldsymbol{\theta}}_j - \boldsymbol{\theta}^*\|_2 \leq 2L_F^{-1}\left(2\widetilde{\Delta}_j + \chi\right)$.

4 Square loss transformation

4.1 Problem formulation

When the true parameter $\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \mathbb{E}\left[M(Y - \boldsymbol{X}^\top\boldsymbol{\theta})\right]$ and $M(\cdot)$ is a pre-determined convex loss function, i.e., $f(\boldsymbol{\theta}; \boldsymbol{Z}) = M(Y - X^\top\boldsymbol{\theta})$, to make the estimation algorithm more efficient, the square loss transformation approach is applied since least square loss can be easily verified to satisfy the assumptions in Section 3.4. The choices of function $M(\cdot)$ include

(i) Huber regression loss with $M(x) = \left(\tau|x| - \tau^2/2\right)(|x| > \tau) + \left(x^2/2\right)\mathbb{1}(|x| \le \tau)$ for a given robustification parameter $\tau > 0$;

(ii) expectile regression loss with $M(x) = x^2(\tau\mathbb{1}(x > 0)+(1-\tau)\mathbb{1}(x \le 0))$ for some pre-determined expectile level $\tau \in (0, 1)$;

(iii) L^q regression loss with $M(x) = |x|^q$ for a known $q \in (1, 2)$ and so on.

Similarly, the penalized estimator for M-estimation can be obtained as follows:

$$\widehat{\boldsymbol{\theta}}_M = \arg\min_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \left\{ \frac{1}{2nm}\sum_{j=1}^{m}\sum_{i=1}^{n} M(Y_{ij} - \boldsymbol{X}_{ij}^\top\boldsymbol{\theta}) + \lambda_M\|\boldsymbol{\theta}\|_1 \right\} \tag{7}$$

where λ_M is the regularization parameter for the original M-estimation. Denote the sub-gradient of $M(x)$ as $M'(x)$ and $m(x) = \mathbb{E}\left[M'(x + \epsilon)\right]$, where $M'(x)$ can be non-continuous and the expectation is taken over the random error term ϵ. Assume that $m(x)$ is differentiable with respect to x and an estimated $\widehat{H}_j^t(0)$ in (10) is used to approximate $m'(0)$.

4.2 SLTBRN method

Based on Algorithm 1, a novel algorithm based on square loss transformation to solve optimization (7) is established. Before computing the gradient (3) at the t-th iteration, define the pseudo-response $\widetilde{Y}_{ij}^t$ for node $j \notin \mathcal{B}$ [Chen et al., 2020, Tu et al., 2023],

$$\widetilde{Y}_{ij}^t := X_{ij}^\top\boldsymbol{\theta}_j^t + (\widehat{H}_j^t(0))^{-1}M'(Y_{ij} - X_{ij}^\top\boldsymbol{\theta}_j^t) \qquad i = 1\ldots, n \tag{8}$$

Therefore, the original M-estimation (7) can be approximated by the following problem

$$\arg\min_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \frac{1}{m}\sum_{j=1}^{m}\mathcal{L}_j^t(\boldsymbol{\theta}) + \lambda_2\|\boldsymbol{\theta}\|_1 \tag{9}$$

where $\mathcal{L}_j^t(\boldsymbol{\theta}) = \sum_{i=1}^{n}(\widetilde{Y}_{ij}^t - X_{ij}^\top j_j^t)^2/(2n)$ and λ_2 is the regularization parameter. To sum up, in place of the original response Y, the pseudo-response $\widetilde{Y}$ is constructed and thus the optimization problem (7) is transformed into the quadratic optimization (9). As the standard LASSO problem can be solved by many effective methods, the square loss transformation performs well in decreasing computation time compared with directly solving (9) even in the non-distributed setting. Given the same initial values as in Algorithm 1, the resulting algorithm for M-estimation is shown in Algorithm 2 in detail. Various experimental results are provided in Section A to show the robust performance of Algorithm 2. It is worthwhile pointing out that computing pseudo-response does not

Algorithm 2 Square Loss Transformation Byzantine-Robust Network (SLTBRN)

Input: Data $\boldsymbol{Z}_N, \boldsymbol{\theta}_j^0 = \boldsymbol{\theta}_j^{\frac{1}{2}} = \boldsymbol{\theta}^0$ with a pre-determined initial vector $\boldsymbol{\theta}^0$ and $\boldsymbol{G}_j^0 = \nabla \mathcal{L}_j^0(\boldsymbol{\theta}_j^0)$ for $j = 1, \ldots, m$, penalty parameters λ and γ, aggregation rule $\mathcal{A}$, the number of iterations T

1: **for** $t = 1, \ldots, T$ **do**

2: **Step 1: Aggregation**

3: Each node $j \notin \mathcal{B}$ receives $\boldsymbol{\theta}_i^{t-\frac{1}{2}}$ from all neighbors $i \in \mathcal{N}_j$ and computes

$$\boldsymbol{\theta}_j^t = \mathcal{A}(\{\boldsymbol{\theta}_i^{t-\frac{1}{2}}\}_{i \in \mathcal{N}_j})$$

where $\boldsymbol{\theta}_i^{t-\frac{1}{2}}$ can be arbitrary if $i \in \mathcal{B}_j$

4: Each node $j \notin \mathcal{B}$ computes pseudo-response

$$\widetilde{Y}_{ij}^t = X_{ij}^\top \boldsymbol{\theta}_j^t + (\widehat{H}_j^t(0))^{-1} M'(Y_{ij} - X_{ij}^\top \boldsymbol{\theta}_j^t) \qquad i = 1, \ldots, n$$

5: Based on pseudo-response, each node $j \notin \mathcal{B}$ computes

$$\boldsymbol{g}_j^t = \boldsymbol{G}_j^{t-1} + \nabla \mathcal{L}_j^t(\boldsymbol{\theta}_j^t) - \nabla \mathcal{L}_j^t(\boldsymbol{\theta}_j^{t-1})$$

6: Each node j receives $\boldsymbol{g}_i^t$ from all neighbors i and computes

$$\boldsymbol{G}_j^t = \mathcal{A}(\{\boldsymbol{g}_i^t\}_{i \in \mathcal{N}_j})$$

where $\boldsymbol{g}_i^t$ can be arbitrary if $i \in \mathcal{B}_j$

7: **Step 2: Optimization**

8: Each node $j \notin \mathcal{B}$ solves

$$\boldsymbol{\theta}_j^{t+\frac{1}{2}} = \underset{\boldsymbol{\theta}_j \in \boldsymbol{\Theta}}{\arg\min} \ (\boldsymbol{G}_j^t)^\top \boldsymbol{\theta}_j + \lambda \left\|\boldsymbol{\theta}_j\right\|_1 + \frac{\gamma}{2} \left\|\boldsymbol{\theta}_j - \boldsymbol{\theta}_j^t\right\|_2^2$$

9: **end for**

Output: $\boldsymbol{\theta}_j^T$ $(j \notin \mathcal{B})$

contribute to the order of complexity compared with $O\left(Td\left(nd + m_{\max}\right)\right)$ such that the Algorithms 1 and 2 have the same complexity.

For the smooth and non-smooth loss functions, different ways should be applied to estimate $m'(0)$, respectively. If the loss function $M(x)$ is smooth, $\widehat{H}_j^t(0) = \sum_{i=1}^n M''(Y_{ij}^t - X_{ij}^\top \boldsymbol{\theta}_j^t)/n$, where $M''(x)$ is the second order derivative of $M(x)$. Otherwise, consider the nonsmooth function $M(x)$. Under this circumstance, $M'(x)$ could be discontinuous with finitely many distinct discontinuous points $x_1, x_2, \ldots, x_K$, but is differentiable outside of these points. In addition, $m'(0)$ for node j at t-th iteration can be approximated by using a given kernel function $\mathcal{K}(\cdot)$ and a bandwidth h_j as

$$\widehat{H}_j^t(0) = \frac{1}{n} \sum_{i=1}^n M''(Y_{ij}^t - X_{ij}^\top \boldsymbol{\theta}_j^t) + \sum_{k=1}^K \frac{\mu_k}{n h_j} \sum_{i=1}^n \mathcal{K}\left((Y_{ij}^t - X_{ij}^\top \boldsymbol{\theta}_j^t - x_k)/h_j\right) \tag{10}$$

where $\mu_k = \lim_{x \to x_k^+} M'(x) - \lim_{x \to x_k^-} M'(x)$.

Remark. For the robust Huber loss function, $M'(x) = \tau\,\mathrm{sign}(x)\mathbb{1}(x > \tau) + x\mathbb{1}(x \le \tau)$ and $M''(x) = \mathbb{1}(x \le \tau)$. It is worth pointing out that $m'(0) = \mathbb{P}(|\epsilon| \le \tau)$ and applying (10) yields $\widehat{H}_j^t(0) = n^{-1}\sum_{i=1}^n \mathbb{1}(|Y_{ij}^t - X_{ij}^\top \boldsymbol{\theta}_j^t| \le \tau)$. For the expectile regression loss function, $M'(x) = 2x(\tau\mathbb{1}(x > 0) + (1-\tau)\mathbb{1}(x \le 0))$ and $M''(x) = 2(\tau\mathbb{1}(x > 0) + (1-\tau)\mathbb{1}(x \le 0))$. Therefore, by (10) $m'(0)$ can be estimated by $\widehat{H}_j^t(0) = 2n^{-1}\sum_{i=1}^n \Big(\tau\mathbb{1}(|Y_{ij}^t - X_{ij}^\top \boldsymbol{\theta}_j^t| > 0) + (1-\tau)\mathbb{1}(|Y_{ij}^t - X_{ij}^\top \boldsymbol{\theta}_j^t| \le 0)\Big)$.

5 Conclusion

In this paper, we propose two Byzantine-robust algorithms to solve sparse M-estimation over decentralized networks, where some nodes may behave arbitrarily due to system failures, cyber attacks, or malicious interference. These decentralized networks are modeled as undirected graphs without a central node, making them particularly vulnerable to Byzantine attacks. Our approach leverages robust aggregation rules such as Median, Trimmed Mean, and Krum, combined with gradient tracking and proximal algorithms, to ensure both robustness and sparsity in high-dimensional settings.

Theoretically, we demonstrate that the proposed algorithms are provably resilient to Byzantine attacks and achieve linear convergence under mild conditions. Extensive numerical experiments, including an application to the Communities and Crime dataset, validate the finite-sample performance of our methods across various settings. Additionally, our algorithms are designed to accommodate general smooth and non-smooth loss functions, with one leveraging the square loss transformation to improve efficiency in non-smooth cases.

While the proposed algorithms are not highly sensitive to the penalty parameters λ and γ, careful tuning via cross-validation is recommended. Future work could explore the use of non-convex penalties, such as SCAD or MCP, to eliminate bias and improve convergence rates. Moreover, addressing the high communication costs in large-scale decentralized networks using gradient compression techniques and extending the framework to handle asynchronous networks would be valuable directions for further research.

References

[Baruch et al., 2019] Baruch, G., Baruch, M., & Goldberg, Y. (2019). A little is enough: Circumventing defenses for distributed learning. *Advances in Neural Information Processing Systems*, 32, 8632–8642.

[Bellet et al., 2018] Bellet, A., Guerraoui, R., Taziki, M., & Tommasi, M. (2018). Personalized and private peer-to-peer machine learning. In *International Conference on Artificial Intelligence and Statistics*, volume 84 (pp. 473–481).: PMLR.

[Blanchard et al., 2017] Blanchard, P., El Mhamdi, E. M., Guerraoui, R., & Stainer, J. (2017). Machine learning with adversaries: Byzantine tolerant gradient descent. *Advances in Neural Information Processing Systems*, 30, 119–129.

[Bubeck, 2015] Bubeck, S. (2015). Convex optimization: Algorithms and complexity. *Foundations and Trends® in Machine Learning*, 8(3-4), 231–357.

[Chen et al., 2021] Chen, J.-H., Chen, M.-R., Zeng, G.-Q., & Weng, J.-S. (2021). BDFL: A Byzantine-fault-tolerance decentralized federated learning method for autonomous vehicle. *IEEE Transactions on Vehicular Technology*, 70(9), 8639–8652.

[Chen et al., 2020] Chen, X., Liu, W., Mao, X., & Yang, Z. (2020). Distributed high-dimensional regression under a quantile loss function. *Journal of Machine Learning Research*, 21(182), 1–43.

[Cheu et al., 2019] Cheu, A., Smith, A., Ullman, J., Zeber, D., & Zhilyaev, M. (2019). Distributed differential privacy via shuffling. In *Advances in Cryptology–EUROCRYPT 2019: 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19–23, 2019, Proceedings, Part I 38* (pp. 375–403).: Springer.

[Colin et al., 2016] Colin, I., Bellet, A., Salmon, J., & Clémençon, S. (2016). Gossip dual averaging for decentralized optimization of pairwise functions. In *International Conference on Machine Learning*, volume 48 (pp. 1388–1396).: PMLR.

[Di Lorenzo & Scutari, 2016] Di Lorenzo, P. & Scutari, G. (2016). Next: In-network nonconvex optimization. *IEEE Transactions on Signal and Information Processing over Networks*, 2(2), 120–136.

[Elkordy et al., 2022] Elkordy, A. R., Prakash, S., & Avestimehr, S. (2022). Basil: A fast and Byzantine-resilient approach for decentralized training. *IEEE Journal on Selected Areas in Communications*, 40(9), 2694–2716.

[Fang et al., 2022] Fang, C., Yang, Z., & Bajwa, W. U. (2022). BRIDGE: Byzantine-resilient decentralized gradient descent. *IEEE Transactions on Signal and Information Processing over Networks*, 8, 610–626.

[Fang et al., 2020] Fang, M., Cao, X., Jia, J., & Gong, N. (2020). Local model poisoning attacks to Byzantine-robust federated learning. In *29th USENIX security symposium (USENIX Security 20)* (pp. 1605–1622).

[Ghosh et al., 2021] Ghosh, A., Maity, R. K., Kadhe, S., Mazumdar, A., & Ramchandran, K. (2021). Communication-efficient and Byzantine-robust distributed learning with error feedback. *IEEE Journal on Selected Areas in Information Theory*, 2(3), 942–953.

[Guo et al., 2021] Guo, S., Zhang, T., Yu, H., Xie, X., Ma, L., Xiang, T., & Liu, Y. (2021). Byzantine-resilient decentralized stochastic gradient descent. *IEEE Transactions on Circuits and Systems for Video Technology*, 32(6), 4096–4106.

[Hastie et al., 2015] Hastie, T., Tibshirani, R., & Wainwright, M. (2015). *Statistical Learning with Sparsity: The Lasso and Generalizations*. Monographs on Statistics and Applied Probability. Chapman & Hall/CRC, Cambridge.

[He et al., 2022] He, L., Karimireddy, S. P., & Jaggi, M. (2022). Byzantine-robust decentralized learning via ClippedGossip. *arXiv preprint arXiv:2202.01545*.

[Hou et al., 2022] Hou, J., Wang, F., Wei, C., Huang, H., Hu, Y., & Gui, N. (2022). Credibility assessment based Byzantine-resilient decentralized learning. *IEEE Transactions on Dependable and Secure Computing*, (pp. 1–12).

[Hu et al., 2023] Hu, J., Chen, G., Li, H., Guo, X., & Huang, T. (2023). Prox-DBRO-VR: A unified analysis on decentralized Byzantine-resilient composite stochastic optimization with variance reduction and non-asymptotic convergence rates. *arXiv preprint arXiv:2305.08051*.

[Karimireddy et al., 2021] Karimireddy, S. P., He, L., & Jaggi, M. (2021). Learning from history for Byzantine robust optimization. In *International Conference on Machine Learning*, volume 139 (pp. 5311–5319).: PMLR.

[Konan et al., 2022] Konan, S., Seraj, E., & Gombolay, M. (2022). Iterated reasoning with mutual information in cooperative and Byzantine decentralized teaming. *arXiv preprint arXiv:2201.08484*.

[Kuwaranancharoen & Sundaram, 2023] Kuwaranancharoen, K. & Sundaram, S. (2023). On the geometric convergence of Byzantine-resilient distributed optimization algorithms. *arXiv preprint arXiv:2305.10810*.

[Lamport et al., 1982] Lamport, L., Shostak, R., & Pease, M. (1982). The Byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, 4(3), 382–401.

[Lecué & Lerasle, 2020] Lecué, G. & Lerasle, M. (2020). Robust machine learning by median-of-means: Theory and practice. *The Annals of Statistics*, 48(2), 906–931.

[Li et al., 2020] Li, T., Sahu, A. K., Zaheer, M., Sanjabi, M., Talwalkar, A., & Smith, V. (2020). Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2, 429–450.

[Lian et al., 2017] Lian, X., Zhang, C., Zhang, H., Hsieh, C.-J., Zhang, W., & Liu, J. (2017). Can decentralized algorithms outperform centralized algorithms? A case study for decentralized parallel stochastic gradient descent. *Advances in Neural Information Processing Systems*, 30, 5336–5346.

[McMahan et al., 2017] McMahan, B., Moore, E., Ramage, D., Hampson, S., & y Arcas, B. A. (2017). Communication-efficient learning of deep networks from decentralized data. In *International Conference on Artificial Intelligence and Statistics*, volume 54 (pp. 1273–1282).: PMLR.

[Nedic, 2020] Nedic, A. (2020). Distributed gradient methods for convex machine learning problems in networks: Distributed optimization. *IEEE Signal Processing Magazine*, 37(3), 92–101.

[Nedić et al., 2018] Nedić, A., Olshevsky, A., & Rabbat, M. G. (2018). Network topology and communication-computation tradeoffs in decentralized optimization. *Proceedings of the IEEE*, 106(5), 953–976.

[Nedic & Ozdaglar, 2009] Nedic, A. & Ozdaglar, A. (2009). Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1), 48–61.

[Parikh & Boyd, 2014] Parikh, N. & Boyd, S. (2014). Proximal algorithms. *Foundations and Trends® in Optimization*, 1(3), 127–239.

[Peng et al., 2021] Peng, J., Li, W., & Ling, Q. (2021). Byzantine-robust decentralized stochastic optimization over static and time-varying networks. *Signal Processing*, 183, 108020.

[Peng et al., 2023] Peng, J., Li, W., & Ling, Q. (2023). Byzantine-robust decentralized stochastic optimization with stochastic gradient noise-independent learning error. *arXiv preprint arXiv:2308.05292*.

[Pillutla et al., 2022] Pillutla, K., Kakade, S. M., & Harchaoui, Z. (2022). Robust aggregation for federated learning. *IEEE Transactions on Signal Processing*, 70, 1142–1154.

[Richards & Rebeschini, 2019] Richards, D. & Rebeschini, P. (2019). Optimal statistical rates for decentralised non-parametric regression with linear speed-up. *Advances in Neural Information Processing Systems*, 32, 1216–1227.

[Richards et al., 2020] Richards, D., Rebeschini, P., & Rosasco, L. (2020). Decentralised learning with random features and distributed gradient descent. In *International conference on machine learning* (pp. 8105–8115).: PMLR.

[Rockafellar, 1976] Rockafellar, R. T. (1976). Monotone operators and the proximal point algorithm. *SIAM Journal on Control and Optimization*, 14(5), 877–898.

[Savazzi et al., 2020] Savazzi, S., Nicoli, M., & Rampa, V. (2020). Federated learning with cooperating devices: A consensus approach for massive IoT networks. *IEEE Internet of Things Journal*, 7(5), 4641–4654.

[Sayed, 2014] Sayed, A. H. (2014). Adaptation, learning, and optimization over networks. *Foundations and Trends® in Machine Learning*, 7(4-5), 311–801.

[Smith et al., 2017] Smith, V., Chiang, C.-K., Sanjabi, M., & Talwalkar, A. S. (2017). Federated multi-task learning. *Advances in Neural Information Processing Systems*, 30, 4424–4434.

[Su & Vaidya, 2020] Su, L. & Vaidya, N. H. (2020). Byzantine-resilient multiagent optimization. *IEEE Transactions on Automatic Control*, 66(5), 2227–2233.

[Sun et al., 2020] Sun, Q., Zhou, W.-X., & Fan, J. (2020). Adaptive huber regression. *Journal of the American Statistical Association*, 115(529), 254–265.

[Sun et al., 2022] Sun, Y., Maros, M., Scutari, G., & Cheng, G. (2022). High-dimensional inference over networks: linear convergence and statistical guarantees. *arXiv preprint arXiv:2201.08507*.

[Tao et al., 2023] Tao, H., Qiu, J., Chen, Y., Stojanovic, V., & Cheng, L. (2023). Unsupervised cross-domain rolling bearing fault diagnosis based on time-frequency information fusion. *Journal of the Franklin Institute*, 360(2), 1454–1477.

[Tibshirani, 1996] Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 58(1), 267–288.

[Tu et al., 2023] Tu, J., Liu, W., & Mao, X. (2023). Byzantine-robust distributed sparse learning for M-estimation. *Machine Learning*, 112(10), 3773–3804.

[Wang et al., 2023] Wang, R., Zhuang, Z., Tao, H., Paszke, W., & Stojanovic, V. (2023). Q-learning based fault estimation and fault tolerant iterative learning control for MIMO systems. *ISA Transactions*, 142, 123–135.

[Wei et al., 2020] Wei, K., Li, J., Ding, M., Ma, C., Yang, H. H., Farokhi, F., Jin, S., Quek, T. Q., & Poor, H. V. (2020). Federated learning with differential privacy: Algorithms and performance analysis. *IEEE Transactions on Information Forensics and Security*, 15, 3454–3469.

[Wu et al., 2023a] Wu, S., Huang, D., & Wang, H. (2023a). Network gradient descent algorithm for decentralized federated learning. *Journal of Business & Economic Statistics*, 41(3), 806–818.

[Wu et al., 2023b] Wu, Z., Chen, T., & Ling, Q. (2023b). Byzantine-resilient decentralized stochastic optimization with robust aggregation rules. *IEEE Transactions on Signal Processing*, 71, 3179–3195.

[Xie et al., 2018] Xie, C., Koyejo, O., & Gupta, I. (2018). Generalized Byzantine-tolerant SGD. *arXiv preprint arXiv:1802.10116*.

[Xie et al., 2020] Xie, C., Koyejo, O., & Gupta, I. (2020). Fall of empires: Breaking Byzantine-tolerant SGD by inner product manipulation. In *Uncertainty in Artificial Intelligence*, volume 115 (pp. 261–270).: PMLR.

[Xu & Huang, 2022] Xu, J. & Huang, S.-L. (2022). Byzantine-resilient decentralized collaborative learning. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 5253–5257).: IEEE.

[Xu et al., 2017] Xu, J., Zhu, S., Soh, Y. C., & Xie, L. (2017). Convergence of asynchronous distributed gradient methods over stochastic networks. *IEEE Transactions on Automatic Control*, 63(2), 434–448.

[Yang & Bajwa, 2019] Yang, Z. & Bajwa, W. U. (2019). ByRDiE: Byzantine-resilient distributed coordinate descent for decentralized learning. *IEEE Transactions on Signal and Information Processing over Networks*, 5(4), 611–627.

[Ye et al., 2024] Ye, H., Zhu, H., & Ling, Q. (2024). On the tradeoff between privacy preservation and Byzantine-robustness in decentralized learning. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 9336–9340).: IEEE.

[Yin et al., 2018] Yin, D., Chen, Y., Kannan, R., & Bartlett, P. (2018). Byzantine-robust distributed learning: Towards optimal statistical rates. In *International Conference on Machine Learning*, volume 80 (pp. 5650–5659).: PMLR.

[Yuan et al., 2016] Yuan, K., Ling, Q., & Yin, W. (2016). On the convergence of decentralized gradient descent. *SIAM Journal on Optimization*, 26(3), 1835–1854.

[Zhao & Yu, 2006] Zhao, P. & Yu, B. (2006). On model selection consistency of lasso. *Journal of Machine Learning Research*, 7, 2541–2563.

[Zhou et al., 2022] Zhou, C., Tao, H., Chen, Y., Stojanovic, V., & Paszke, W. (2022). Robust point-to-point iterative learning control for constrained systems: A minimum energy approach. *International Journal of Robust and Nonlinear Control*, 32(18), 10139–10161.

[Zhou et al., 2023] Zhou, X., Chang, L., Xu, P., & Lv, S. (2023). Communication-efficient and Byzantine-robust distributed learning with statistical guarantee. *Pattern Recognition*, 137, 109312.

[Zhu et al., 2023] Zhu, B., Wang, L., Pang, Q., Wang, S., Jiao, J., Song, D., & Jordan, M. I. (2023). Byzantine-robust federated learning with optimal statistical rates. In *International Conference on Artificial Intelligence and Statistics* (pp. 3151–3178).: PMLR.

[Zhuang et al., 2022] Zhuang, Z., Tao, H., Chen, Y., Stojanovic, V., & Paszke, W. (2022). An optimal iterative learning control approach for linear systems with nonuniform trial lengths under input constraints. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 53(6), 3461–3473.

A Numerical experiments

In this section, a series of numerical experiments are presented to show the finite-sample performance of the proposed method. All the simulated data are distributed across different nodes and then connected by a network. To be specific, the communication networks among these nodes are formed based on a random Erdős-Rényi graph $G(m, q)$, where m is the number of nodes and $q \in (0, 1]$ is the probability of each edge.

According to [Ghosh et al., 2021, Karimireddy et al., 2021, Hou et al., 2022] and [Xie et al., 2020], the following four different kinds of Byzantine attacks are applied: add normal distributed entries (AN), bit flipping (BF), gaussian attack (GA), fall of empires (FE).

(1) **AN:** the Byzantine nodes add i.i.d. $N_p(\mathbf{0}, 10I_p)$ entries to the statistics before broadcasting them.

(2) **BF:** the Byzantine nodes multiply the statistics by -1 before broadcasting them.

(3) **GA:** the Byzantine nodes broadcast forged statistics sampled from a coordinate-wise standard Gaussian distribution.

(4) **FE:** Byzantine parameters can be tailored to have similar variance and magnitude as the correct gradient, which makes them hard to be distinguished. In this paper, the forged parameter submitted by the Byzantine node is $\widehat{T} = (1 - \delta)T_{\text{ave}}$, where T_{ave} is the average value of all nonfaulty parameters and $\delta = 1.1$.

Moreover, the Byzantine nodes are assumed omniscient, i.e., they could aggregate the information from all nodes at all time. All experimental results are attained by 100 replications. Notice that the four Byzantine attacks above only have effects on these statistics which would be broadcasted, thus each node $j \in \mathcal{V}$ can obtain its estimation for $\boldsymbol{\theta}^*$. In the decentralized learning fields, the mean square error (MSE) [Sun et al., 2022, Sun et al., 2022] and loss [Lian et al., 2017, Savazzi et al., 2020] are the most commonly used evaluation criteria. Since the true parameter $\boldsymbol{\theta}^*$ is known in our numerical experiments, we propose to use the MSE for better illustration of the performance of the proposed estimators. To be specific, the MSE is computed among the m nodes by averaging as $\sum_{j=1}^{m} \left\|\boldsymbol{\theta}_j^t - \boldsymbol{\theta}^*\right\|_2^2 / m$.

A.1 Linear regression model

The data are generated from a linear regression model:

$$Y_{ij} = \boldsymbol{X}_{ij}^\top \boldsymbol{\theta}^* + \epsilon_{ij}$$

where $\boldsymbol{X}_{ij}^\top = \left(1, X_{ij1}, X_{ij2}, \ldots, X_{ij(p-1)}\right)^\top \in \mathbb{R}^p$, $\left(X_{ij1}, X_{ij2}, \ldots, X_{ij(p-1)}\right)^\top$ independently comes from $N_{p-1}(\mathbf{0}, I_{p-1})$, ϵ_{ij} is the error term and $\boldsymbol{\theta}^*$ is the unknown target parameter. Set the dimension $p = 200$, number of nonzero coefficients $s = \lceil \sqrt{p} \rceil$, initial vector $\boldsymbol{\theta}_0 = \mathbf{0}$, number of iterations $T = 100$, local sample size $n = 100$, number of nodes $m = 50$, connectivity probability $q = 0.5$, the number of Byzantine nodes $b = 5$, and the error term $\epsilon \sim N(0, 1)$ independently. The first s elements of the true value $\boldsymbol{\theta}^*$ are independently generated from $N(0, 1)$. In order to see the sensitivities of these parameters, the following Section A.1.1, Section A.1.2, Section A.1.3 and Section A.1.4 show the results of the effect of error type, the number of Byzantine nodes, the connectivity probability and local sample size, respectively. Under each set of values for these parameters, we compare the following five estimators.

(1) Centralized: a central node collects all the samples and solves the penalized linear regression problem, which is used as a gold standard.

(2) NetLASSO: directly use the NetLASSO algorithm of [Sun et al., 2022], which is used as a benchmark.

(3) SLBRN-M: the proposed method in Algorithm 1 with **Median** aggregation.

(4) SLBRN-T: the proposed method in Algorithm 1 with **Trimmed mean** aggregation.

(5) SLBRN-K: the proposed method in Algorithm 1 with **Krum** aggregation.

Since the penalty parameters λ and γ have effects on the convergence and performance in practice, they are determined by the cross-validation approach. Alternatively, to facilitate comparison, we also can generate an independent set of test data of size nm and choose the tuning parameters via grid search that achieve the smallest prediction error.

A.1.1 Effect of error type

To investigate the performance of the Centralized, NetLASSO, SLBRN-K, SLBRN-M and SLBRN-T methods, the following six different errors are considered:

$$\begin{array}{lll} \text{(I) } \epsilon \sim N(0,1) & \text{(II) } \epsilon \sim t(3) & \text{(III) } \epsilon \sim \chi^2(1) - 1 \\ \text{(IV) } \ln(\epsilon + e^{0.5}) \sim N(0,1) & \text{(V) } \epsilon \sim \text{Exp}(1) - 1 & \text{(VI) } \epsilon \sim (0.5 + 0.5|X_1|)\, N(0,1) \end{array}$$

The averaged log(MSE) values are reported in Fig. 2 and Fig. 3 (one error type per column in Fig. 2 and Fig. 3) and a few conclusions can be drawn as follows. For any fixed Byzantine attack and error type, all methods except NetLASSO can attain their corresponding stable performance after 30 iterations. In terms of MSEs, it can be seen that the Centralized method based on the entire data always has the smallest MSEs; the proposed three methods have slightly larger MSEs than those of the Centralized method and their MSEs are close to each other. However, NetLASSO performs badly and diverges to the largest MSEs which are far larger than those of our proposed methods. Thus, in general, our proposed methods perform well in the presence of the Byzantine attacks, while NetLASSO does not work in practice.

A.1.2 Effect of the number of Byzantine nodes (b)

We vary the number of Byzantine nodes from $b = 0, 2, 5, 8$ and the experimental results are shown in Fig. 4. Some conclusions can be drawn as follows. (1) For any fixed Byzantine attack and b value, all these methods perform similarly to Section A.1.1. (2) When b increases, the MSEs of all methods increase except the Centralized method based on the entire data in one node. The MSEs of the proposed methods vary little as b increases. This shows that the proposed estimators are insensitive to the number of Byzantine nodes.

A.1.3 Effect of the connectivity probability of random graph (q)

We vary $q = 0.3, 0.5, 0.7$ to see how the connectivity probability influences the results. The experimental results are given in Fig. 5 and the following conclusions can be obtained. For any fixed q, the conclusions of the MSE results among these methods are similar as in Section A.1.1. As q increases, i.e., there are more edges in the graph, the MSEs of the Centralized method keep the

same, the MSEs of the SLBRN-K method decrease slightly, while other methods have stable MSE results and are not sensitive to q.

A.1.4 Effect of local sample size (n)

To see the impact of local sample size n on these methods, we consider varying $n = 50, 100, 200$ and the experimental results are shown in Fig. 6. For any fixed n, the conclusions about the MSE results are similar to those in Section A.1.1. As n increases, i.e., the total sample size also increases, the MSEs of all methods decrease, but the downward trend of the NetLASSO method is negligible.

A.2 General M-estimation

In this section, we consider Huber and expectile regression models such that the Centralized and NetLASSO methods in Section A.1 do not work. To see the feasibility of our proposed estimator after using the square loss transformation and aggregation, the methods in Section A.1 are revised with transformation. The finite-sample performance of five estimators as follows are compared.

(1) Centralized: a central node collects all the samples and solves the penalized M-estimation with transformation at each iteration, which is used as a gold standard.

(2) NetLASSO: revise the NetLASSO algorithm of [Sun et al., 2022] with transformation at each iteration, which is used as a benchmark.

(3) SLTBRN-M: the proposed method in Algorithm 2 with **Median** aggregation.

(4) SLTBRN-T: the proposed method in Algorithm 2 with **Trimmed mean** aggregation.

(5) SLTBRN-K: the proposed method in Algorithm 2 with **Krum** aggregation.

For the robust Huber loss function, the robustification parameter δ is chosen as in [Sun et al., 2020] and we consider the six different errors as in Section A.1.1. For the expectile regression loss function, the same error terms are considered except their τ th expectile equaling 0 with two expectile levels $\tau = 0.3, 0.7$. The experimental results are shown in Fig. 7, Fig. 8, Fig. 9, Fig. 10, Fig. 11, Fig. 12 and the similar conclusions as Section A.1.1 can be drawn. For example, for any fixed regression model, Byzantine attack and error type, the NetLASSO diverges while our proposed methods converge and perform similarly to the Centralized method. In conclusion, our proposed SLTBRN algorithm shows stable performance even if the target problem is general M-estimation.

B Real data

Motivated examples Our study is motivated by the Communities and Crime Data Set, which contains 1993 observations with 101 predictive attributes altogether collected by 9 communities (New England, Mid-Atlantic, East North Central, West North Central, South Atlantic, East South Central, West South Central, Mountain and Pacific, Fig. 13). Due to the differences in culture, history, geography and economy, these 9 nodes are not fully connected with each other and it is not easy to identify a central node. The logarithm of total number of violent crimes per 100 K population is treated as the response and we want to study the relationship between the response and 101 covariates. In addition, [Sun et al., 2022] investigated wireless networks with low-power

devices communicating only with nodes in their physical proximity. The existence of the Byzantine nodes in decentralized networks with high-dimensional covariates, and their impacts on the estimation motivate us to find an efficient and robust approach.

In this section, our proposed algorithm is applied to Communities and Crime Data Set from the UCI Machine Learning Repository (`http://archive.ics.uci.edu/ml/datasets/communities+and+crime+unnormalized`). After removing the predictive variables and communities with missing values, we finally get a dataset which consists of 101 predictive attributes ($p = 101$) and 1993 observations, which are distributed in $m = 9$ nodes with the numbers of observations as $n_j = 258, 358, 217, 87, 262, 122, 239, 98, 352$, respectively, for $j = 1, \ldots, 9$. The logarithm of total number of violent crimes per 100 K population is treated as the response (Y_{ij}).

After standardizing the covariates and response, the following linear regression is considered:

$$Y_{ij} = \boldsymbol{X}_{ij}^{\top}\boldsymbol{\theta} + \epsilon_{ij}$$

where $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_{101})^{\top}$ is the target parameter and $X_{ij} = (X_{ij1}, X_{ij2}, \ldots, X_{ij101})^{\top}$ is the i-th sample of node j. The data of each node are randomly partitioned into training data and testing data, where training data account for 80%. As for the communication networks among nodes, they are generated when our proposed methods can work by using a random graph $G(9, 0.5)$. Moreover, set $b = 1$ and consider the four kinds of Byzantine attacks mentioned in Section A. Two penalty parameters λ and γ are given by the cross-validation approach. The training data and testing data are used to estimate $\boldsymbol{\theta}$ and test the performance, respectively. Let A_j be the indicator set of the training data of node j. To be specific, the prediction error is set as the measure of evaluation and the mean prediction square error (MPSE) or square loss for the t-th iteration is computed as

$$\mathrm{MPSE}(t) = \frac{1}{\sum_{j=1}^{9} \mathrm{Card}(A_j)} \sum_{j=1}^{9} \sum_{i \in A_j} (\widehat{Y}_{ij}^{t} - Y_{ij})^2$$

where $\widehat{Y}_{ij}^{t} = X_{ij}^{\top}\boldsymbol{\theta}_j^{t}, \{Y_{ij}, X_{ij}\}$ belongs to the training data of node j. Similar with Section A, our proposed methods are compared with the Centralized and NetLASSO methods. The results are given in Fig. 14 and the following conclusions can be obtained similar to Section A .

(1) **Convergence:** For any fixed Byzantine attack, all methods except NetLASSO can attain their corresponding stable performance after 40 iterations. It is worthwhile pointing out that the log (MPSEs) of NetLASSO even diverge to a value much larger than 4, thus we do not show the results of NetLASSO in the figure.

(2) **MPSE:** For any fixed Byzantine attack, it can be seen that the Centralized method always has the smallest MPSEs again. The proposed three methods have slightly larger MPSEs than those of the Centralized method. Nevertheless, NetLASSO performs badly and diverges to the largest MPSEs which are far larger than those of our proposed methods. Therefore, our proposed methods handle the Byzantine attack problems well while NetLASSO fails.

C Technical lemmas

Lemma 1. *Suppose that Assumption 1 holds, we have $\lambda_F \leq L_F$.*

Proof. By Assumption 1, we have $F(\theta)$ is L_F-smooth and λ_F-strongly convex such that

$$F(\theta_1) - F(\theta_2) - \langle \nabla F(\theta_2), \theta_1 - \theta_2 \rangle \geq \frac{\lambda_F}{2} \|\theta_1 - \theta_2\|_2 \tag{11}$$

and

$$F(\theta_2) - F(\theta_1) - \langle \nabla F(\theta_1), \theta_2 - \theta_1 \rangle \geq \frac{\lambda_F}{2} \|\theta_2 - \theta_1\|_2 \tag{12}$$

Combining (11) and (12), it leads to

$$\begin{aligned} \lambda_F \|\theta_1 - \theta_2\|_2 &\leq \langle \nabla F(\theta_1) - \nabla F(\theta_2), \theta_1 - \theta_2 \rangle \leq \|\nabla F(\theta_1) - \nabla F(\theta_2)\|_2 \cdot \|\theta_1 - \theta_2\|_2 \\ &\leq L_F \|\theta_1 - \theta_2\|_2 \end{aligned}$$

where the last inequality holds since $F(\theta)$ is L_F-smooth. Thus we have $\lambda_F \leq L_F$ which completes the proof. □

Lemma 2. *Define the gradient broadcasted to node j as*

$$\boldsymbol{g}_i(\boldsymbol{\theta}) := \begin{cases} \nabla L_i(\boldsymbol{\theta}), & i \in \mathcal{N}_j \backslash \mathcal{B}_j \\ *, & i \in \mathcal{B}_j \end{cases}$$

where $*$ can be arbitrary. Define the coordinate-wise median of $\boldsymbol{g}_i(\boldsymbol{\theta})$ broadcasted to node j as $\boldsymbol{G}_j(\boldsymbol{\theta}) := \boldsymbol{Median}\{\boldsymbol{g}_i(\boldsymbol{\theta}) : i \in \mathcal{N}_j\}$. Suppose that Assumption 1, Assumption 2 and Assumption 3 hold, and Assumption 5 hold for some $\delta > 0$. Then with probability at least $1 - \frac{4p}{(1+nm\widehat{L}D)^p}$ we have

$$\|\boldsymbol{G}_j(\boldsymbol{\theta}) - \nabla F(\boldsymbol{\theta})\|_2 \leq \frac{2\sqrt{2}}{nm_j} + \sqrt{\frac{2}{n}} C_\delta V \left(\alpha_j + \sqrt{\frac{p \log(1 + nm_j \widehat{L} D)}{m_j(1-\alpha)}} + 0.4748 \frac{S}{\sqrt{n}} \right) \tag{13}$$

for all $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, where C_δ is defined as in the Theorem 1.

Proof. This lemma can be established directly by applying Claim 2 in [Yin et al., 2018]. □

Lemma 3. *Define the gradient broadcasted to node j as $\boldsymbol{g}_i(\boldsymbol{\theta})$, $i \in \mathcal{N}_j$ the same as Lemma Lemma 2. Further define the coordinate-wise trimmed mean of $\boldsymbol{g}_i(\boldsymbol{\theta})$ broadcasted to node j as $\widetilde{\boldsymbol{G}}_j(\boldsymbol{\theta}) :=$ Trimmed $\text{mean}_{\beta_j}\{\boldsymbol{g}_i(\boldsymbol{\theta}) : i \in \mathcal{N}_j\}$, where β_j is a pre-given fraction for node j. Suppose that Assumption 1 and Assumption 4 are satisfied, and that for any $j \in \mathcal{V}, \alpha_j \leq \beta_j \leq 1/2 - \widetilde{\delta}$ for some $\widetilde{\delta}$. Then with probability at least $1 - \frac{2p(m_j+1)}{\left(1+nm_j\widetilde{L}D\right)^p}$ we have*

$$\left\| \widetilde{\boldsymbol{G}}_j(\boldsymbol{\theta}) - \nabla F(\boldsymbol{\theta}) \right\|_2 \leq \frac{v}{\epsilon} \left(\frac{3\sqrt{2}\beta_j p}{\sqrt{n}} + \frac{2p}{\sqrt{nm_j}} \right) \sqrt{\log(1 + nm_j \widehat{L} D) + \frac{1}{p} \log m} + O\left(\frac{\beta_j}{n} + \frac{1}{nm_j} \right)$$

for all $\boldsymbol{\theta} \in \boldsymbol{\Theta}$.

Proof. This lemma can be established directly by applying Claim 5 in [Yin et al., 2018]. □

D Proof of theorems

D.1 Proof of Theorem 1

Proof. We condition on the event (see Appendix D.1 in [Yin et al., 2018] for more details) that the bound in (13) is satisfied for all $\boldsymbol{\theta} \in \boldsymbol{\Theta}$. Some simple calculations based on Algorithm 1 yield

$$\boldsymbol{\theta}_j^{t+1} = \Pi_{\boldsymbol{\Theta}} \left(S_\lambda \left(\boldsymbol{\theta}_j^t - \frac{1}{\gamma} \boldsymbol{G}_j^t \right) \right)$$

where $S_\lambda(\boldsymbol{\theta})$ is the soft-thresholding operator,

$$[S_\lambda(\boldsymbol{\theta})]_k = \begin{cases} [\boldsymbol{\theta}]_k - \lambda, & [\boldsymbol{\theta}]_k > \lambda \\ 0, & -\lambda \leq [\boldsymbol{\theta}]_k \leq \lambda \\ [\boldsymbol{\theta}]_k + \lambda, & [\boldsymbol{\theta}]_k < -\lambda \end{cases} \qquad k = 1, \ldots, p$$

where $[\boldsymbol{\theta}]_k$ means k-th coordinate of vector $\boldsymbol{\theta}$. Then at the t-th iteration, we define

$$\widehat{\boldsymbol{\theta}}^{t+1} = S_\lambda \left(\boldsymbol{\theta}_j^t - \frac{1}{\gamma} \boldsymbol{G}_j^t \right)$$

Thus, we derive $\boldsymbol{\theta}_j^{t+1} = \Pi_\Theta \left(\boldsymbol{\theta}^{t+1} \right)$. By the property of Euclidean projection, we have

$$\|\boldsymbol{\theta}_j^{t+1} - \boldsymbol{\theta}^*\|_2 \leq \|\widehat{\boldsymbol{\theta}}_j^{t+1} - \boldsymbol{\theta}^*\|_2$$

Recall that $\boldsymbol{G}_j^t = \mathbf{Median} \left\{ \boldsymbol{g}_i^t : i \in \mathcal{N}_j \right\}$, where

$$\boldsymbol{g}_i^t = \begin{cases} \boldsymbol{G}_i^{t-1} + \nabla L_i \left(\boldsymbol{\theta}_i^t \right) - L_i \left(\boldsymbol{\theta}_i^{t-1} \right) & i \in \mathcal{N}_j \backslash \mathcal{B}_j \\ * & i \in \mathcal{B}_j \end{cases}$$

In order to get the convergence properties, define $\boldsymbol{Q}_j^t = \mathbf{Median} \left\{ \boldsymbol{q}_i^t : i \in \mathcal{N}_j \right\}$ and $\boldsymbol{Q}_j^0 = \boldsymbol{G}_j^0 = \nabla L_j(\boldsymbol{\theta}_j^0)$, where

$$\boldsymbol{q}_i^t = \begin{cases} \boldsymbol{Q}_i^{t-1} + \nabla L_i(\boldsymbol{\theta}_j^t) - L_i(\boldsymbol{\theta}_j^{t-1}) & i \in \mathcal{N}_j \backslash \mathcal{B}_j \\ *, & i \in \mathcal{B}_j \end{cases}$$

We further derive

$$\begin{aligned} \|\boldsymbol{\theta}_j^{t+1} - \boldsymbol{\theta}^*\|_2 &\leq \left\| \boldsymbol{\theta}_j^t - \frac{1}{\gamma} \boldsymbol{G}_j^t - \boldsymbol{\theta}^* \right\|_2 + \sqrt{p}\lambda \\ &\leq \left\| \boldsymbol{\theta}_j^t - \frac{1}{\gamma} \boldsymbol{G}_j(\boldsymbol{\theta}_j^t) - \boldsymbol{\theta}^* \right\|_2 + \frac{1}{\gamma} \left(\left\| \boldsymbol{G}_j(\boldsymbol{\theta}_j^t) - \boldsymbol{Q}_j^t \right\|_2 + \left\| \boldsymbol{Q}_j^t - \boldsymbol{G}_j^t \right\|_2 \right) + \sqrt{p}\lambda \\ &:= T_1 + \frac{1}{\gamma} (T_2 + T_3) + \sqrt{p}\lambda \end{aligned} \tag{14}$$

where $T_1 = \left\| \boldsymbol{\theta}_j^t - \frac{1}{\gamma} G_j(\boldsymbol{\theta}_j^t) - \boldsymbol{\theta}^* \right\|_2$, $T_2 = \left\| \boldsymbol{G}_j(\boldsymbol{\theta}_j^t) - \boldsymbol{Q}_j^t \right\|_2$, $T_3 = \left\| \boldsymbol{Q}_j^t - \boldsymbol{G}_j^t \right\|_2$. We establish the upper bounds of T_1, T_2 and T_3 as follows.

Upper bound of T_1. We compute that

$$T_1 \leq \left\| \boldsymbol{\theta}_j^t - \frac{1}{\gamma}\nabla F(\boldsymbol{\theta}_j^t) - \boldsymbol{\theta}^* \right\|_2 + \frac{1}{\gamma}\left\| \boldsymbol{G}_j(\boldsymbol{\theta}_j^t) - \nabla F(\boldsymbol{\theta}_j^t) \right\|_2 \tag{15}$$

Meanwhile, we have

$$\left\| \boldsymbol{\theta}_j^t - \frac{1}{\gamma}\nabla F(\boldsymbol{\theta}_j^t) - \boldsymbol{\theta}^* \right\|_2^2 = \left\| \boldsymbol{\theta}_j^t - \boldsymbol{\theta}^* \right\|_2^2 - \frac{2}{\gamma}\left\langle \boldsymbol{\theta}_j^t - \boldsymbol{\theta}^*, \nabla F(\boldsymbol{\theta}_j^t) \right\rangle + \frac{1}{\gamma^2}\left\| \nabla F(\boldsymbol{\theta}_j^t) \right\|_2^2$$

Since $F(\boldsymbol{\theta})$ is λ_F-strongly convex, by the co-coercivity of strongly convex functions (see Lemma 3.11 in [Bubeck, 2015] for more details), we derive

$$\left\langle \boldsymbol{\theta}_j^t - \boldsymbol{\theta}^*, \nabla F(\boldsymbol{\theta}_j^t) \right\rangle \geq \frac{L_F \lambda_F}{L_F + \lambda_F}\left\| \boldsymbol{\theta}_j^t - \boldsymbol{\theta}^* \right\|_2^2 + \frac{1}{L_F + \lambda_F}\left\| \nabla F(\boldsymbol{\theta}_j^t) \right\|_2^2$$

Let $\gamma = L_F$, we have

$$\begin{aligned}
\left\| \boldsymbol{\theta}_j^t - \frac{1}{\gamma}\nabla F(\boldsymbol{\theta}_j^t) - \boldsymbol{\theta}^* \right\|_2^2 &\leq \left(1 - \frac{2\lambda_F}{L_F + \lambda_F}\right)\left\| \boldsymbol{\theta}_j^t - \boldsymbol{\theta}^* \right\|_2^2 - \frac{2}{L_F\left(L_F + \lambda_F\right)}\left\| \nabla F(\boldsymbol{\theta}_j^t) \right\|_2^2 + \frac{1}{L_F^2}\left\| \nabla F(\boldsymbol{\theta}_j^t) \right\|_2^2 \\
&\leq \left(1 - \frac{2\lambda_F}{L_F + \lambda_F}\right)\left\| \boldsymbol{\theta}_j^t - \boldsymbol{\theta}^* \right\|_2^2
\end{aligned}$$

where the second inequality holds by Lemma Lemma 1. Using the fact $\sqrt{1-x} \leq 1 - \frac{x}{2}$ for $x \leq 1$, we get

$$\left\| \boldsymbol{\theta}_j^t - \frac{1}{\gamma}\nabla F(\boldsymbol{\theta}_j^t) - \boldsymbol{\theta}^* \right\|_2 \leq \left(1 - \frac{\lambda_F}{L_F + \lambda_F}\right)\left\| \boldsymbol{\theta}_j^t - \boldsymbol{\theta}^* \right\|_2 \tag{16}$$

Combining (15) and (16), applying Lemma Lemma 1 we get

$$T_1 \leq \left(1 - \frac{\lambda_F}{L_F + \lambda_F}\right)\left\| \boldsymbol{\theta}_j^t - \boldsymbol{\theta}^* \right\|_2 + \frac{\Delta_j}{L_F} \tag{17}$$

where

$$\Delta_j = O_p\left(C_\delta V\left(\frac{\alpha_j}{\sqrt{n}} + \sqrt{\frac{p\log(nm_j\widehat{L}D)}{nm_j}} + \frac{S}{\sqrt{n}}\right)\right)$$

and

$$C_\delta = \sqrt{2\pi}\exp\left(\frac{1}{2}\left(\Phi^{-1}(1-\delta)\right)^2\right)$$

with $\Phi^{-1}(\cdot)$ being the inverse of the cumulative distribution function of the standard Gaussian distribution $\Phi(\cdot)$.

Upper bound of T_2. Without loss of generality, suppose that

$$\left\| \boldsymbol{G}_j(\boldsymbol{\theta}_j^t) - \boldsymbol{Q}_j^t \right\|_\infty = \left[\left| \boldsymbol{G}_j(\boldsymbol{\theta}_j^t) - \boldsymbol{Q}_j^t \right| \right]_{l_1} = \max_{1 \leq k \leq p}\left[\left| \boldsymbol{G}_j(\boldsymbol{\theta}_j^t) - \boldsymbol{Q}_j^t \right| \right]_k$$

for some $1 \leq l_1 \leq p$. By Assumption 5 we know that $\alpha_j < 1/2$, thus there exist four series $a_j(t), b_j(t), c_j(t), d_j(t) \in \mathcal{N}_j \backslash \mathcal{B}_j$, such that

$$\left[\nabla L_{a_j(t)}(\boldsymbol{\theta}_j^t)\right]_{l_1} \leq \left[\boldsymbol{G}_j(\boldsymbol{\theta}_j^t)\right]_{l_1} \leq \left[\nabla L_{b_j(t)}(\boldsymbol{\theta}_j^t)\right]_{l_1}$$

and

$$\left[\boldsymbol{Q}_{c_j(t)}^{t-1}+\nabla L_{c_j(t)}(\boldsymbol{\theta}_j^t)-\nabla L_{c_j(t)}(\boldsymbol{\theta}_j^{t-1})\right]_{l_1}\leq\left[\boldsymbol{Q}_j^t\right]_{l_1}\leq\left[\boldsymbol{Q}_{d_j(t)}^{t-1}+\nabla L_{d_j(t)}(\boldsymbol{\theta}_j^t)-\nabla L_{d_j(t)}(\boldsymbol{\theta}_j^{t-1})\right]_{l_1}$$

Thus

$$\begin{aligned}T_2&\leq\sqrt{p}\left\|\boldsymbol{G}_j(\boldsymbol{\theta}_j^t)-\boldsymbol{Q}_j^t\right\|_\infty\\&\leq\sqrt{p}\max\left\{\left[\left|\boldsymbol{Q}_{d_j(t)}^{t-1}+\nabla L_{d_j(t)}(\boldsymbol{\theta}_j^t)-\nabla L_{d_j(t)}(\boldsymbol{\theta}_j^{t-1})-\nabla L_{a_j(t)}(\boldsymbol{\theta}_j^t)\right|\right]_{l_1},\right.\\&\qquad\left.\left[\left|\boldsymbol{Q}_{c_j(t)}^{t-1}+\nabla L_{c_j(t)}(\boldsymbol{\theta}_j^t)-\nabla L_{c_j(t)}(\boldsymbol{\theta}_j^{t-1})-\nabla L_{b_j(t)}(\boldsymbol{\theta}_j^t)\right|\right]_{l_1}\right\}\\&:=\sqrt{p}\max\left\{T_{21},T_{22}\right\}\end{aligned}$$

We bound T_{21} as follows,

$$\begin{aligned}T_{21}&\leq\left[\left|\boldsymbol{Q}_{d_j(t)}^{t-1}-\nabla F(\boldsymbol{\theta}_j^{t-1})\right|\right]_{l_1}+\left[\left|\nabla L_{d_j(t)}(\boldsymbol{\theta}_j^{t-1})-\nabla F(\boldsymbol{\theta}_j^{t-1})\right|\right]_{l_1}\\&\quad+\left[\left|\nabla L_{d_j(t)}(\boldsymbol{\theta}_j^t)-\nabla F(\boldsymbol{\theta}_j^t)\right|\right]_{l_1}+\left[\left|\nabla L_{a_j(t)}(\boldsymbol{\theta}_j^t)-\nabla F(\boldsymbol{\theta}_j^t)\right|\right]_{l_1}\\&\leq\left[\left|\boldsymbol{Q}_{d_j(t)}^{t-1}-\nabla F(\boldsymbol{\theta}_j^{t-1})\right|\right]_{l_1}+O_p\left(\frac{1}{\sqrt{nm_j}}\right)\end{aligned}\tag{18}$$

where the last inequality holds by the central limit theorem (Assumption 3). Then we show the upper bound (order) of $\left[\left|\boldsymbol{Q}_i^{t-1}-\nabla F(\boldsymbol{\theta}_j^{t-1})\right|\right]_{l_1}$ is $O_p(D\sqrt{p/n})$ for any $i,j\notin\mathcal{B}$ by the inductive method.

When $t=1$, applying the central limit theorem we get

$$\begin{aligned}\left[\left|\boldsymbol{Q}_i^0-\nabla F(\boldsymbol{\theta}_j^0)\right|\right]_{l_1}&=\left[\left|\nabla L_i(\mathbf{0})-\nabla F(\mathbf{0})\right|\right]_{l_1}\\&=O_p\left(\frac{1}{\sqrt{n}}\right)\leq O_p\left(D\sqrt{\frac{p}{n}}\right)\end{aligned}$$

for any $i,j\notin\mathcal{B}$. Assume that $\left[\left|\boldsymbol{Q}_i^{t-2}-\nabla F(\boldsymbol{\theta}_j^{t-2})\right|\right]_{l_1}\leq O_p(1/\sqrt{n})$ for any $i,j\notin\mathcal{B}$, since $\alpha_j\leq 1/2$, thus there exist series $u_i(t),v_i(t)\in\mathcal{N}_i\backslash\mathcal{B}_i$, such that

$$\left[\boldsymbol{Q}_{u_i(t)}^{t-2}+\nabla L_{u_i(t)}(\boldsymbol{\theta}_{u_i(t)}^{t-1})-\nabla L_{u_i(t)}(\boldsymbol{\theta}_{u_i(t)}^{t-2})\right]_{l_1}\leq\left[\boldsymbol{Q}_i^{t-1}\right]_{l_1}\leq\left[\boldsymbol{Q}_{v_i(t)}^{t-2}+\nabla L_{v_i(t)}(\boldsymbol{\theta}_{v_i(t)}^{t-1})-\nabla L_{v_i(t)}(\boldsymbol{\theta}_{v_i(t)}^{t-2})\right]_{l_1}$$

Thus

$$\begin{aligned}&\left[\left|\boldsymbol{Q}_i^{t-1}-\nabla F(\boldsymbol{\theta}_j^{t-1})\right|\right]_{l_1}\leq\max\left\{\left[\left|\boldsymbol{Q}_{u_i(t)}^{t-2}+\nabla L_{u_i(t)}(\boldsymbol{\theta}_{u_i(t)}^{t-1})-\nabla L_{u_i(t)}(\boldsymbol{\theta}_{u_i(t)}^{t-2})-\nabla F(\boldsymbol{\theta}_j^{t-1})\right|\right]_{l_1}\right.\\&\quad\left.\left[\left|\boldsymbol{Q}_{v_i(t)}^{t-2}+\nabla L_{v_i(t)}(\boldsymbol{\theta}_{v_i(t)}^{t-1})-\nabla L_{v_i(t)}(\boldsymbol{\theta}_{v_i(t)}^{t-2})-\nabla F(\boldsymbol{\theta}_j^{t-1})\right|\right]_{l_1}\right\}\\&:=\max\left\{T_{23},T_{24}\right\}\end{aligned}$$

Then we bound T_{23} as follows

$$\begin{aligned}T_{23}&\leq\left[\left|\boldsymbol{Q}_{u_i(t)}^{t-2}-\nabla F(\boldsymbol{\theta}_j^{t-2})\right|\right]_{l_1}+\left[\left|\nabla L_{u_i(t)}(\boldsymbol{\theta}_{u_i(t)}^{t-2})-\nabla F(\boldsymbol{\theta}_j^{t-2})\right|\right]_{l_1}+\left[\left|\nabla L_{u_i(t)}(\boldsymbol{\theta}_{u_i(t)}^{t-1})-\nabla F(\boldsymbol{\theta}_j^{t-1})\right|\right]_{l_1}\\&:=S_{21}+S_{22}+S_{23}\end{aligned}$$

Consider the term S_{21}, by the assumption in the inductive method we get

$$S_{21} \leq O_p(D\sqrt{p/n}) \tag{19}$$

We further calculate

$$\begin{aligned}
S_{22} &= \left[\left|\nabla L_{u_i(t)}(\boldsymbol{\theta}^{t-2}_{u_i(t)}) - \nabla F(\boldsymbol{\theta}^{t-2}_j)\right|\right]_{l_1} \\
&\leq \left[\left|\nabla L_{u_i(t)}(\boldsymbol{\theta}^{t-2}_{u_i(t)}) - \nabla L_{u_i(t)}(\boldsymbol{\theta}^{t-2}_j)\right| + \left|\nabla L_{u_i(t)}(\boldsymbol{\theta}^{t-2}_j) - \nabla F(\boldsymbol{\theta}^{t-2}_j)\right|\right]_{l_1} \\
&= \left[\left|\nabla^2 L_{u_i(t)}(\boldsymbol{\theta}^{t-2}_j)(\boldsymbol{\theta}^{t-2}_{u_i(t)} - \boldsymbol{\theta}^{t-2}_j)\right|\right]_{l_2} + O_p\left(\frac{1}{\sqrt{n}}\right) \\
&\leq O_p\left(D\sqrt{\frac{p}{n}}\right) + O_p\left(\frac{1}{\sqrt{n}}\right) = O_p\left(D\sqrt{\frac{p}{n}}\right)
\end{aligned} \tag{20}$$

where the last equality holds by the Taylor expansion and the central limit theorem, the last inequality holds by the convexity and compactness of $\boldsymbol{\Theta}$ and the central limit theorem. Similar to S_{22}, we compute

$$\begin{aligned}
S_{23} &= \left[\left|\nabla L_{u_i(t)}(\boldsymbol{\theta}^{t-1}_{u_i(t)}) - \nabla F(\boldsymbol{\theta}^{t-1}_j)\right|\right]_{l_1} \\
&\leq \left[\left|\nabla L_{u_i(t)}(\boldsymbol{\theta}^{t-1}_{u_i(t)}) - \nabla L_{u_i(t)}(\boldsymbol{\theta}^{t-1}_j)\right| + \left|\nabla L_{u_i(t)}(\boldsymbol{\theta}^{t-1}_j) - \nabla F(\boldsymbol{\theta}^{t-1}_j)\right|\right]_{l_1} \\
&= \left[\left|\nabla^2 L_{u_i(t)}(\boldsymbol{\theta}^{t-1}_j)(\boldsymbol{\theta}^{t-1}_{u_i(t)} - \boldsymbol{\theta}^{t-1}_j)\right|\right]_{l_2} + O_p\left(\frac{1}{\sqrt{n}}\right) \\
&\leq O_p\left(D\sqrt{\frac{p}{n}}\right) + O_p\left(\frac{1}{\sqrt{n}}\right) = O_p\left(D\sqrt{\frac{p}{n}}\right)
\end{aligned} \tag{21}$$

Combining the results of (19), (20) and (21) we have

$$T_{23} \leq S_{21} + S_{22} + S_{23} \leq O_p\left(D\sqrt{\frac{p}{n}}\right)$$

Similar to T_{23}, we can bound the term $T_{24} \leq O_p(D\sqrt{p/n})$. Thus we derive $\left[\left|\boldsymbol{Q}^{t-1}_i - \nabla F(\boldsymbol{\theta}^{t-1}_j)\right|\right]_{l_1} \leq O_p(D\sqrt{p/n})$ for any $i, j \notin \mathcal{B}$ by the inductive method, which implies

$$\left[\left|\boldsymbol{Q}^{t-1}_{d_j(t)} - \nabla F(\boldsymbol{\theta}^{t-1}_j)\right|\right]_{l_1} \leq O_p\left(D\sqrt{\frac{p}{n}}\right) \tag{22}$$

Combining (18) and (22) we derive $T_{21} \leq O_p(D\sqrt{p/n})$. Similar to T_{21}, we can bound the term $T_{22} \leq O_p(D\sqrt{p/n})$. Thus we get the upper bound of T_2 as

$$T_2 \leq \sqrt{p}\max\{T_{21}, T_{22}\} \leq O_p\left(\frac{Dp}{\sqrt{n}}\right) \tag{23}$$

Upper bound of T_3. Without loss of generality, suppose that

$$\left\|\boldsymbol{G}^t_j - \boldsymbol{Q}^t_j\right\|_\infty = \left[|\boldsymbol{G}^t_j - \boldsymbol{Q}^t_j|\right]_{l_2} = \max_{1\leq k\leq p}\left[|\boldsymbol{G}^t_j - \boldsymbol{Q}^t_j|\right]_k$$

for some $1 \le l_2 \le p$. Then we show the upper bound (order) of $\left[|\boldsymbol{G}_i^t - \boldsymbol{Q}_j^t|\right]_{l_2}$ is $O_p(D\sqrt{p/n})$ for any $i, j \notin \mathcal{B}$ by the inductive method. When $t = 0, \boldsymbol{G}_i^0 = \nabla L_i(\mathbf{0})$ and $\boldsymbol{Q}_j^0 = \nabla L_j(\mathbf{0})$ for any $i, j \notin \mathcal{B}$. Applying the central limit theorem we get

$$\begin{aligned}\left[|\boldsymbol{G}_i^0 - \boldsymbol{Q}_j^0|\right]_{l_2} &= \left[|\nabla L_i(\mathbf{0}) - \nabla L_j(\mathbf{0})|\right]_{l_2} \\ &\le \left[|\nabla L_i(\mathbf{0}) - \nabla F(\mathbf{0})| + |\nabla L_j(\mathbf{0}) - \nabla F(\mathbf{0})|\right]_{l_2} \\ &= O_p\left(\frac{1}{\sqrt{n}}\right) = O_p\left(D\sqrt{\frac{p}{n}}\right)\end{aligned}$$

Assume that $\left[|\boldsymbol{G}_i^{t-1} - \boldsymbol{Q}_j^{t-1}|\right]_{l_2} = O_p(D\sqrt{p/n})$ for any $i, j \notin \mathcal{B}$. Since $\alpha_j < 1/2$, thus there exist four series $a_i'(t), b_i'(t), c_j'(t), d_j'(t) \in \mathcal{N}_j \backslash \mathcal{B}_j$, such that

$$\left[\boldsymbol{g}_{a_i'(t)}^t\right]_{l_2} \le \left[\boldsymbol{G}_i^t\right]_{l_2} \le \left[\boldsymbol{g}_{b_i'(t)}^t\right]_{l_2}$$

where $\boldsymbol{g}_{a_i'(t)}^t = \boldsymbol{G}_{a_i'(t)}^{t-1} + \nabla L_{a_i'(t)}(\boldsymbol{\theta}_{a_i'(t)}^t) - \nabla L_{a_i'(t)}(\boldsymbol{\theta}_{a_i'(t)}^{t-1}), \boldsymbol{g}_{b_i'(t)}^t = \boldsymbol{G}_{b_i'(t)}^{t-1} + \nabla L_{b_i'(t)}(\boldsymbol{\theta}_{a_i'(t)}^t) - \nabla L_{b_i'(t)}(\boldsymbol{\theta}_{a_i'(t)}^{t-1})$, and

$$\left[\boldsymbol{q}_{c_j'(t)}^t\right]_{l_2} \le \left[\boldsymbol{Q}_j^t\right]_{l_2} \le \left[\boldsymbol{q}_{d_j'(t)}^t\right]_{l_2}$$

where $\boldsymbol{q}_{c_j'(t)}^t = \boldsymbol{Q}_{c_j'(t)}^{t-1} + \nabla L_{c_j'(t)}(\boldsymbol{\theta}_j^t) - \nabla L_{c_j'(t)}(\boldsymbol{\theta}_j^{t-1}), \boldsymbol{q}_{d_j'(t)}^t = \boldsymbol{Q}_{d_j'(t)}^{t-1} + \nabla L_{d_j'(t)}(\boldsymbol{\theta}_j^t) - \nabla L_{d_j'(t)}(\boldsymbol{\theta}_j^{t-1})$. Thus we derive

$$\begin{aligned}\left[|\boldsymbol{G}_i^t - \boldsymbol{Q}_j^t|\right]_{l_2} &\le \max\left\{\left[\left|\boldsymbol{q}_{c_i'(t)}^t - \boldsymbol{g}_{b_i'(t)}^t\right|\right]_{l_2}, \left[\left|\boldsymbol{q}_{d_i'(t)}^t - \boldsymbol{g}_{a_i'(t)}^t\right|\right]_{l_2}\right\} \\ &:= \max\left\{T_{31}, T_{32}\right\} \end{aligned} \tag{24}$$

We further compute

$$\begin{aligned}T_{31} &= \left[\left|\boldsymbol{q}_{c_i'(t)}^t - \boldsymbol{g}_{b_i'(t)}^t\right|\right]_{l_2} \\ &\le \left[\left|\boldsymbol{Q}_{c_j'(t)}^{t-1} - \boldsymbol{G}_{b_i'(t)}^{t-1}\right|\right]_{l_2} + \left[\left|\nabla L_{c_j'(t)}(\boldsymbol{\theta}_j^t) - \nabla L_{b_i'(t)}(\boldsymbol{\theta}_{a_i'(t)}^t)\right|\right]_{l_2} + \left[\left|\nabla L_{c_j'(t)}(\boldsymbol{\theta}_j^{t-1}) - \nabla L_{b_i'(t)}(\boldsymbol{\theta}_{a_i^{(t)}}^{t-1})\right|\right]_{l_2} \\ &:= S_1 + S_2 + S_3 \end{aligned} \tag{25}$$

By the assumption in the inductive method we know that

$$S_1 = O_p\left(D\sqrt{\frac{p}{n}}\right) \tag{26}$$

Then we calculate

$$S_2 = \left[\left|\nabla L_{c_j'(t)}(\boldsymbol{\theta}_j^t) - \nabla L_{b_i'(t)}(\boldsymbol{\theta}_{a_i'(t)}^t)\right|\right]_{l_2} \tag{27}$$

$$\le \left[\left|\nabla L_{c_j'(t)}(\boldsymbol{\theta}_j^t) - \nabla F(\boldsymbol{\theta}_j^t)\right| + \left|\nabla L_{b_i'(t)}(\boldsymbol{\theta}_j^t) - \nabla F(\boldsymbol{\theta}_j^t)\right| + \left|\nabla L_{b_i'(t)}(\boldsymbol{\theta}_{a_i'(t)}^t) - \nabla L_{b_i'(t)}(\boldsymbol{\theta}_j^t)\right|\right]_{l_2} \tag{28}$$

$$= O_p\left(\frac{1}{\sqrt{n}}\right) + O\left[\left|\nabla^2 L_{b_i'(t)}(\boldsymbol{\theta}_j^t)(\boldsymbol{\theta}_{a_i'(t)}^t - \boldsymbol{\theta}_j^t)\right|\right]_{l_2} \tag{29}$$

$$\le O_p\left(\frac{1}{\sqrt{n}}\right) + O_p\left(D\sqrt{\frac{p}{n}}\right) = O_p\left(D\sqrt{\frac{p}{n}}\right) \tag{30}$$

where the last equality holds by the Taylor expansion and the central limit theorem, the last inequality holds by the convexity and compactness of $\boldsymbol{\Theta}$ and the central limit theorem. Similar to S_2, we have

$$\begin{aligned}
S_3 &= \left[\left|\nabla L_{c'_j(t)}(\boldsymbol{\theta}_j^{t-1}) - \nabla L_{b'_i(t)}(\boldsymbol{\theta}_{a'_i(t)}^{t-1})\right|\right]_{l_2} \\
&\leq \left[\left|\nabla L_{c'_j(t)}(\boldsymbol{\theta}_j^{t-1}) - \nabla F(\boldsymbol{\theta}_j^{t-1})\right| + \left|\nabla L_{b'_i(t)}(\boldsymbol{\theta}_j^{t-1}) - \nabla F(\boldsymbol{\theta}_j^{t-1})\right| + \left|\nabla L_{b'_i(t)}(\boldsymbol{\theta}_{a'_i(t)}^{t-1}) - \nabla L_{b'_i(t)}(\boldsymbol{\theta}_j^{t-1})\right|\right]_{l_2} \\
&= O_p\left(\frac{1}{\sqrt{n}}\right) + O\left[\left|\nabla^2 L_{b'_i(t)}(\boldsymbol{\theta}_j^{t-1})(\boldsymbol{\theta}_{a'_i(t)}^{t} - \boldsymbol{\theta}_j^{t-1})\right|\right]_{l_2} \\
&\leq O_p\left(\frac{1}{\sqrt{n}}\right) + O_p\left(D\sqrt{\frac{p}{n}}\right) = O_p\left(D\sqrt{\frac{p}{n}}\right) \qquad (31)
\end{aligned}$$

Combining (25), (26), (30) and (31) we obtain $T_{31} \leq O_p\left(D\sqrt{p/n}\right)$. Similar to T_{31}, we can bound the term $T_{32} \leq O_p(D\sqrt{p/n})$. Thus by the inductive method we derive

$$\left[\left|G_i^t - \boldsymbol{Q}_j^t\right|\right]_{l_2} = \max\left\{T_{31}, T_{32}\right\} \leq O_p\left(D\sqrt{\frac{p}{n}}\right) \qquad (32)$$

for any $i, j \notin \mathcal{B}$. Applying (26) yields the bound for T_3 as

$$\begin{aligned}
T_3 = \left\|\boldsymbol{G}_j^t - \boldsymbol{Q}_j^t\right\|_2 &\leq \sqrt{p}\left\|\boldsymbol{G}_j^t - \boldsymbol{Q}_j^t\right\|_\infty = \sqrt{p}\left[\left|\ \boldsymbol{G}_j^t - \boldsymbol{Q}_j^t\right\|_{l_2}\right. \\
&\leq O_p\left(\frac{Dp}{\sqrt{n}}\right) \qquad (33)
\end{aligned}$$

As a conclusion, combining (14), (17), (23), (33) and $\gamma = L_F$ we get the iteration error as

$$\left\|\boldsymbol{\theta}_j^{t+1} - \boldsymbol{\theta}^*\right\|_2 \leq \left(1 - \frac{\lambda_F}{L_F + \lambda_F}\right)\left\|\boldsymbol{\theta}_j^t - \boldsymbol{\theta}^*\right\|_2 + \frac{\Delta_j + \chi}{L_F}$$

where $\chi = O_p((Dp)/\sqrt{n})$. □

D.2 Proof of Theorem 2

Proof. The proof of Theorem 2 consists of two parts: (i) the analysis of coordinate-wise trimmed mean aggregation; (ii) the convergence analysis of the robustified optimization step in the algorithm. The rest of the proof is essentially the same as the proof of Theorem 1 .

Consider the first part, in fact, we essentially analyze a quasi gradient descent algorithm with bounded noise in the gradients. In the proof of Theorem 1 in Appendix D.1, the bound on the noise in gradients is Δ_j for node j, while here we replace Δ_j with $\widetilde{\Delta}_j$ as defined in Theorem 2.

Then we consider the second part, it is worthwhile pointing out that we bound the statistics after taking median in the proof of Theorem 1, while here applying Lemma 3 we need to found the lower and upper bound for **Trimmed mean** aggregation. For example, consider $\left[\boldsymbol{G}_j(\boldsymbol{\theta}_j^t)\right]_{l_1}$, there exist two series $a_j(t), b_j(t) \in \mathcal{N}_j \backslash \mathcal{B}_j$, such that

$$\left[\nabla L_{a_j(t)}(\boldsymbol{\theta}_j^t)\right]_{l_1} \leq \left[\boldsymbol{G}_j(\boldsymbol{\theta}_j^t)\right]_{l_1} \leq \left[\nabla L_{b_j(t)}(\boldsymbol{\theta}_j^t)\right]_{l_1}$$

Since $\alpha_j \leq \beta_j \leq 1/2 - \widetilde{\delta}$ for some $\widetilde{\delta} > 0$, thus there exist two series $\widetilde{a}_j(t), \widetilde{b}_j(t) \in \mathcal{N}_j \backslash \mathcal{B}_j$, such that

$$\left[\nabla L_{\widetilde{a}_j(t)}(\boldsymbol{\theta}_j^t)\right]_{l_1} \leq \left[\boldsymbol{G}_j(\boldsymbol{\theta}_j^t)\right]_{l_1} \leq \left[\nabla L_{\widetilde{b}_j(t)}(\boldsymbol{\theta}_j^t)\right]_{l_1}$$