

# Robust Federated Learning in a Heterogeneous Environment

Avishek Ghosh\*, Justin Hong\*, Dong Yin, and Kannan Ramchandran

{avishek\_ghosh, jjhong922, dongyin, kannanr}@berkeley.edu

Department of Electrical Engineering and Computer Sciences, UC Berkeley

## Abstract

We study a recently proposed large-scale distributed learning paradigm, namely Federated Learning, where the worker machines are end users' own devices. Statistical and computational challenges arise in Federated Learning particularly in the presence of heterogeneous data distribution (i.e., data points on different devices belong to different distributions signifying different clusters) and Byzantine machines (i.e., machines that may behave abnormally, or even exhibit arbitrary and potentially adversarial behavior). To address the aforementioned challenges, first we propose a general statistical model for this problem which takes both the cluster structure of the users and the Byzantine machines into account. Then, leveraging the statistical model, we solve the robust heterogeneous Federated Learning problem *optimally*; in particular our algorithm matches the lower bound on the estimation error in dimension and the number of data points. Furthermore, as a by-product, we prove statistical guarantees for an outlier-robust clustering algorithm, which can be considered as the Lloyd algorithm with robust estimation. Finally, we show via synthetic as well as real data experiments that the estimation error obtained by our proposed algorithm is significantly better than the non-Byzantine-robust algorithms; in particular, we gain at least by 53% and 33% for synthetic and real data experiments, respectively, in typical settings.

## 1 Introduction

Distributed computing is becoming increasingly important in many modern data-intensive applications like computer vision, natural language processing and recommendation systems. Federated Learning ([1, 2, 3]) is one recently proposed distributed computing paradigm that aims to fully utilize on-device machine intelligence—in such systems, data are stored in end users' own devices such as mobile phones and personal computers. Many statistical and computational challenges arise in Federated Learning, due to the highly decentralized system architecture. In this paper, we aim to tackle two challenges in Federated Learning: Byzantine robustness and heterogeneous data distribution.

In Federated Learning, robustness has become one of the major concerns since individual computing units (worker machines) may exhibit abnormal behavior owing to corrupted data, faulty hardware, crashes, unreliable communication channels, stalled computation, or even malicious and coordinated attacks. It is well known that the overall performance of such a system can be arbitrarily skewed even if a single machine behaves in a Byzantine way. Hence it is necessary to develop distributed learning algorithms that are provably robust against Byzantine failures. This is considered in a few recent works, and much progress has been made (see [4, 5, 6, 7, 8]).

In practice, since worker nodes are end users' personal devices, the issue of data heterogeneity naturally arises in Federated Learning. Exploiting data heterogeneity is particularly crucial in recommendation systems and personalized advertisement placement, which benefits both the users' and the enterprises. For example, mobile phone users who read news articles may be interested in different categories of news like politics, sports or fashion; advertisement platforms might need to send different categories of ads to different groups of customers. These indicate that leveraging

---

\*Equal contributions.

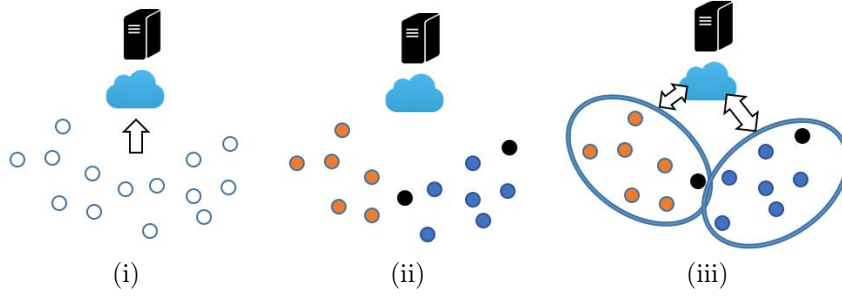


Figure 1: A modular algorithm for Byzantine-robust optimization with heterogeneous data. (i) The  $m$  worker machines send their local risk minimizers to the center machine. (ii) The center machine runs robust clustering algorithm (red: cluster 1, blue: cluster 2, black: Byzantine machines; center machine may not know which machines are Byzantine). (iii) In each cluster, the center and worker machines jointly run a robust distributed optimization algorithm.

cluster structures among the users is of potential interest—each machine itself may not have enough data and thus we need to better utilize the similarity among the users in the same cluster. This problem has recently received attention in [9] in a non-statistical multi-task setting.

We believe that more effort is needed in this area in order to achieve better statistical guarantees and robustness against Byzantine failures. In this paper, we aim to tackle the data heterogeneity and Byzantine-robustness problems simultaneously. We propose a statistical model, along with a 3 stage algorithm that solves the aforementioned problem yielding an estimation error which is *optimal* in dimension and number of data points. The crux of our approach lies in analyzing a clustering algorithm in the presence of adversarial data points. In particular, we study the classical Lloyd’s algorithm augmented with robust estimation. Specifically, we show that the number of misclustered points with the robust Lloyd algorithm decays at an exponential rate when initialized properly. We now summarize the contributions of the paper.

## 1.1 Our contributions

We propose a general and flexible statistical model and a general algorithmic framework to address the heterogeneous Federated Learning problem in the presence of Byzantine machines. Our algorithmic framework consists of three stages: finding local solutions, performing centralized robust clustering and doing joint robust distributed optimization. The error incurred by our algorithm is optimal in several problem parameters. Furthermore, our framework allows for flexible choices of algorithms in each stage, and can be easily implemented in a modular manner.

Moreover, as a by-product, we analyze an outlier-robust clustering scheme, which may be considered as the Lloyd’s algorithm with robust estimation. The idea of robustifying the Lloyd’s algorithm is not new (e.g. see [10, 11] and the references therein) and several robust Lloyd algorithms are empirically well studied. However, to the best of our knowledge, this is the first work that analyzes and prove guarantees for such algorithms in a statistical setting, and might be of independent interest.

We validate our theoretical results via simulations on both synthetic and real world data. For synthetic experiments, using a mixture of regressions model, we find that our proposed algorithm drastically outperforms the non-Byzantine-robust algorithms. Further, using Yahoo! Learning to Rank dataset, we demonstrate that our proposed algorithm is practical, easy to implement and dominates the standard non-robust algorithms.

## 1.2 Related work

**Distributed and Federated Learning:** Learning with a distributed computing framework has been studied extensively in various settings [12, 13, 14, 15, 16]. Since the paradigm of Federated Learning presented by [1, 3], several recent works focus on different applications of the problem, such as in deep learning [2], predicting health events from wearable devices, and detecting burglaries in smart homes [17, 18]. While [19] deals with fairness in Federated Learning, [20, 21] deal with non-iid data. A few recent works study heterogeneity under different setting in Federated Learning, for example see [9, 22, 23, 24] and the references therein. However, neither of these papers explicitly

---

**Algorithm 1** A 3 stage modular algorithm for robust Federated Learning in a heterogeneous environment

---

**Require:** Center node,  $m$  compute nodes, loss function  $f$ .

- 1: Worker nodes send ERM  $\hat{w}^{(i)} := \operatorname{argmin}_{w \in \mathcal{W}} F^{(i)}(w)$  (for all  $i \in [m]$ ) to the center.
  - 2: Center nodes cluster  $\{\hat{w}^{(i)}\}_{i=1}^m$  to obtain  $\mathcal{C}_1, \dots, \mathcal{C}_K$ .
  - 3: At each cluster  $\mathcal{C}_1, \dots, \mathcal{C}_K$  run distributed Byzantine tolerant iterative optimization algorithm.
- 

utilize the cluster structure of the problem in the presence of Byzantine machines. Also, in most cases, the objective is to learn a single optimal parameter for the whole problem, instead of learning optimal parameters for each cluster. In contrast, the MOCHA algorithm [9] considers a multi-task learning setting and forms an optimization problem with the correlation matrix of the users being a regularization term. Our work differs from MOCHA since we consider a statistical setting and the Byzantine-robustness.

**Byzantine-robustness:** The robustness and security issues in distributed learning has received much attention ([25, 26]). In particular, one recent work by [27] studies the Byzantine-robust distributed learning from heterogeneous datasets. However, the basic goal of this work differs from ours, since we aim to optimize different prediction rules for different users, whereas [27] tries to find a single optimal solution.

**Clustering and mixture models:** In the centralized setting, outlier-robust clustering and mixture models have been extensively studied. Robust clustering has been studied in many previous works [28, 29, 30]. One recent work [31] considers a statistical model for robust clustering, similar to ours. However, their algorithm is computationally heavy and hard to implement, whereas the robust clustering algorithm in our paper is more intuitive and straightforward to implement. Our work is also related to learning mixture models, such as mixture of experts [32] and mixture of regressions [33, 34].

## 2 Problem setup

We consider a standard statistical setting of empirical risk minimization (ERM). Our goal is to learn several parametric models by minimizing some (convex) loss functions defined by the data. Suppose we have  $m$  compute nodes,  $\alpha m$  ( $\alpha < 0.5$ ) of which are Byzantine nodes, i.e., nodes that are arbitrarily corrupted by some adversary. Out of the non-Byzantine compute nodes, we assume that there are  $K$  different data distributions,  $\mathcal{D}_1, \dots, \mathcal{D}_K$ , and that the  $(1 - \alpha)m$  machines are partitioned into  $K$  clusters,  $\mathcal{C}_1, \dots, \mathcal{C}_K$ . Suppose that every node  $i \in \mathcal{C}_k$  contains  $n$  i.i.d. data points  $x^{i,1}, \dots, x^{i,n}$  drawn from  $\mathcal{D}_k$ . We also assume that we have no control over the data distribution of the corrupt nodes. Let  $f(w; x) : \mathcal{W} \rightarrow \mathbb{R}$  be the loss function associated with data point  $x$ , where  $\mathcal{W} \subseteq \mathbb{R}^d$  is the parameter space. Our goal is to find the minimizers of all the  $K$  population risk functions. For the  $k$ -th cluster, the minimizer is  $w_k^* = \arg \min_{w \in \mathcal{W}} F_k(w) (= \mathbb{E}_{x \sim \mathcal{D}_k} f(w; x))$ .

The challenges in learning  $\{w_i^*\}_{i=1}^K$  are: (i) we need a clustering scheme that work in presence of adversaries. Since, we have no control over the corrupted nodes, it is not possible to cluster all the nodes perfectly. Hence we need a robust distributed optimization algorithm. (ii) we want our algorithm to minimize uplink communication cost ([3]). Throughout, we use  $C, C_1, \dots, c, c_1, \dots$  for universal constants; whose value may vary from line to line. Also,  $\|\cdot\|$  denotes  $\ell_2$  norm.

## 3 A modular algorithm for robust Federated Learning in heterogeneous environment

In this section, we present a modular algorithm that consists of 3 stages—(1) Compute local empirical risk minimizers (ERMs) and send them to the center machine (2) Run outlier-robust clustering algorithm on these local ERMs and (3) Run a communication-efficient, robust, distributed optimization on each cluster (Algorithm 1, also see Figure 1).

### 3.1 Stage I- compute ERMs

In this step, each compute node calculates the local empirical risk minimizer (ERM) associated to its risk function send them to the center machine. Since machine  $i$  is associated with the local risk function, defined as  $F^{(i)}(w) = \frac{1}{n} \sum_{j=1}^n f(w, x_i^j)$ , the local ERM,  $\hat{w}^{(i)} := \operatorname{argmin}_{w \in \mathcal{W}} F^{(i)}(w)$ . We assume the loss function  $f(\cdot, \cdot)$  is convex with respect to its first argument, and so the compute node can run a convex optimization program to solve for  $\hat{w}^{(i)}$ .

Instead of solving the local risk function directly, the compute node can run an “online-to-batch conversion” routine. Each compute node runs an online optimization algorithm like Online Gradient Descent [35]. At iteration  $l$ , the compute node picks  $w_l$ , and incurs a loss of  $f(w_l, x_l)$ . After  $n$  episodes with the sequence of functions  $f(\cdot, x_1), \dots, f(\cdot, x_n)$ , the compute node sets the predictor  $\bar{w}^{(i)}$  as the average of the online choices  $w_1, \dots, w_n$  made over  $n$  instances. This predictor has similar properties like ERMs, however in case of online optimization, there is no need to store all  $n$  data points apriori, and the entire operation is in a streaming setup.

### 3.2 Stage II- cluster the ERMs

The second step of the modular algorithms deals with clustering the compute nodes based on their local ERMs. All  $m$  compute nodes send local ERMs,  $\hat{w}^{(i)}$ , for  $i \in [m]$ <sup>1</sup> to the center machine, and the center machine runs a clustering algorithm on these data points to find  $K$  clusters  $\mathcal{C}_1, \dots, \mathcal{C}_K$ . Since compute nodes can be Byzantine, the clustering algorithm should be outlier-robust.

We show (in Section ??) that if the amount of data in each worker node,  $n$  is reasonably large, a simple threshold based clustering rule is sufficient. This scheme uses the fact that the local ERMs of 2 machines belonging to a same cluster are close, whereas they are far apart for different clusters. However, if  $n$  is small (which is pragmatic in Federated Learning), the aforementioned scheme fails to work. An alternative is to use a robust version of Lloyd algorithm ( $K$ -means). In particular: (i) at each iteration, assign the data points to its closest center (ii) compute a robust estimate of the mean with the assigned points for each cluster and use them as new centers and (iii) iterate until convergence.

The first step is identical to that of the data point assignment of  $K$ -means algorithm. There are a few options for robust estimation for mean. Out of them the most common estimates are geometric median [36], coordinate-wise median, and trimmed mean. Although these mean estimates are robust, the estimation error  $\sim \sqrt{d}$  ( $d$  being the dimension) which is prohibitive in large dimension. There is a recent line of work on robust mean estimation that adapts nicely to high dimension [37, 38]. In these results, the mean estimation error is either dimension-independent or very weakly dependent on dimension. In Section A, we analyze this clustering scheme rigorously both in moderate and high dimension.

Since we are dealing with the case where  $\alpha m$  workers are corrupted, and since we do not have control over the corrupt machines, no clustering algorithm can cluster all the compute nodes correctly, and hence we need a robust optimization algorithm that takes care of the adversarially corrupt (albeit Byzantine) nodes. This is precisely done in the third stage of the modular algorithm.

### 3.3 Stage III- outlier-robust distributed optimization

After clustering, we run an outlier-robust distributed algorithm on each cluster. Each cluster can be thought of an instance of homogeneous distributed learning problem with possibly Byzantine machines. Hence, we can use the trimmed mean algorithm of [7] (since it has optimal statistical rate) for low to moderate dimension and the iterative filtering algorithm of [8] for high dimension. These algorithms are communication-efficient; the number of parallel iterations needed matches the standard results of gradient descent algorithm.

## 4 Main results

We now present the main results of the paper. Recall the problem set-up of Section 2. Our goal is to learn the optimal weights  $w_1^*, \dots, w_k^*$ . By running the modular algorithm described in the previous section, we compute final output of the learned weights as  $\hat{w}_1, \dots, \hat{w}_k$ . All the proofs of this section are deferred to Section A. We start with the following set of assumptions.

<sup>1</sup>For integer  $q$ ,  $[q]$  denotes the set of integers  $\{1, \dots, q\}$ .

**Assumption 1.** The loss function  $f(\cdot, x)$  is  $G_1$  Lipschitz:  $|f(w, x) - f(w_1, x)| \leq G_1 \|w - w'\|$  for all  $w, w' \in \mathcal{W}$ .

**Assumption 2.**  $f(\cdot, x)$  is  $\lambda$ -strongly convex: for all  $w$  and  $w' \in \mathcal{W}$ ,

$$f(w, x) - f(w', x) - \langle \nabla f(w', x), w - w' \rangle \geq \frac{\lambda}{2} \|w - w'\|^2$$

**Assumption 3.**  $F_k(w)$  is  $\lambda_F$  strongly convex,  $L_1$  smooth (i.e.,  $\|\nabla^2 F_k(w)\|_{\text{op}} \leq L_1 \ \forall w \in \mathcal{W}$ ).

**Assumption 4.** The function  $f(\cdot, x)$  is  $L$  smooth. For any  $x$  the partial derivative of  $f(w, x)$  with respect to the  $j$ -th coordinate,  $\partial_j f(w, x)$  is  $L^{(j)}$  Lipschitz and  $v$ -sub exponential for all  $j \in [d]$ .

Note that, as illustrated in [7], the above structural assumptions on the partial derivative of the loss function are satisfied in several learning problems.

**Assumption 5.**  $\{w_k^*\}_{k=1}^K$  are separated:  $\min_{i \neq j} \|w_i^* - w_j^*\| \geq R$  and  $n \geq \frac{G_1^2 L_1 \log m}{\lambda^3}$ .

**Remark 1.** If  $f(w, \cdot)$  is  $G_1$  Lipschitz,  $\|\nabla f(w, \cdot)\| \leq G_1$ , and hence  $G_1^2$  can be  $\mathcal{O}(d)$ . Also  $\lambda$  could be potentially small in many applications. Hence Assumption 5 enforces a strict requirement on  $n$ .

Let the size of  $i$ -th cluster is  $M_i$  and  $\hat{\alpha}_i := \frac{\alpha m}{M_i + \alpha m}$ . Furthermore, let  $\max_{i \in [K]} \hat{\alpha}_i < \frac{1}{2}$ .

**Theorem 1.** Suppose Assumptions 1 – 5 hold. If Algorithm 1 is run with the “Edge cutting” (Section ??) algorithm for stage II and the trimmed mean algorithm (of [7]) for  $T$  iterations with constant step-size of  $1/L_1$  in stage III, then provided  $T \geq \tilde{\mathcal{O}}(\frac{L_1 + \lambda_F}{\lambda_F})$ , for all  $i \in [K]$ , we obtain

$$\|\hat{w}_i - w_i^*\| \leq \tilde{\mathcal{O}} \left( \frac{\hat{\alpha}_i d}{\sqrt{n}} + \frac{d}{\sqrt{n M_i}} \right).$$

with probability at least  $1 - m^{-10} - \mathcal{O} \left( \frac{d}{(1 + n M_i)^d} \right)$ .

**Remark 2.** We can remove the assumption of the strong convexity of  $f(\cdot, x)$  (Assumption 2). In that case, under the setting of Theorem 1, for all  $i \in [K]$ , we obtain

$$F_i(\hat{w}_i) - F_i(w_i^*) \leq \tilde{\mathcal{O}} \left( \frac{\hat{\alpha}_i d}{\sqrt{n}} + \frac{d}{\sqrt{n M_i}} \right)$$

with high probability.

*Comparison with an Oracle:* We compare the above bound with an Oracle inequality. We assume that the oracle knows the cluster identity for all the non-Byzantine machines. Since with high probability, the modular algorithm makes no mistake in clustering the non-Byzantine machines, the bound we get perfectly matches the oracle bound.

We now move to the setting where we have no restriction on  $n$ , and hence  $n$  may be potentially much smaller than  $d$ . This setting is more realistic since data arising from applications (like images and video) are high dimensional, and the amount of data in data owners’ device may be small ([1]). We start with the following assumption.

**Assumption 6.** The empirical risk minimizers,  $\{\hat{w}^{(i)}\}_{i=1}^{(1-\alpha)m}$ , corresponding to non-Byzantine machines are sampled from a mixture of  $K$   $\sigma$ -sub-gaussian distributions.

We emphasize that several learning problems satisfy Assumption 6. We now exhibit one such setting where the empirical risk minimizer is Gaussian. We assume that machine  $i$  belongs to cluster  $\mathcal{C}_k$ . Recall that  $\{x^{i,j}\}_{j=1}^n$  denote the data points for machine  $i$ .

**Proposition 1.** Suppose the data  $\{x^{i,j}\}_{j=1}^n$  are sampled from a parametric class of generative model:  $x^{i,j} = \langle \chi_j, w_k^* \rangle + \Upsilon_j$  with covariate  $\chi_j^T \in \mathbb{R}^d$  and i.i.d noise  $\Upsilon_j \sim \mathcal{N}(0, v^2)$ . Then, with quadratic loss, the distribution of the empirical risk minimizer  $\hat{w}^{(i)}$  is Gaussian with mean  $w_k^*$ .

---

**Algorithm 2** Trimmed  $K$ -means

---

**Require:** Observations  $\hat{w}^{(1)}, \dots, \hat{w}^{(m)}$ , initial labels  $\{z_i^{(0)}\}_{i=1}^m$ .

- 1: **for**  $s = 1, 2, \dots$  **do**
  - 2:   Form  $K$  buckets with data points in each bucket having same  $z_i^{(s)}$ . In each bucket:  
       Compute geometric median of the data points. Construct a ball of radius  $C\sigma\sqrt{d}$  around the geometric median (for constant  $C$ ) and compute sample mean of all the data points inside the norm ball as center estimates  $\{\hat{\theta}_g^{(s+1)}\}_{g=1}^K$ .
  - 3:   Re-assign data to the closest center: for all  $i \in [m]$ ,  $z_i^{(s+1)} = \operatorname{argmin}_{g \in [K]} \|\hat{w}^{(i)} - \hat{\theta}_g^{(s+1)}\|_2$ .
  - 4: **end for**
- 

In general, sub-Gaussian distributions form a huge class, including all bounded distributions. For non-Byzantine machines, we assume the observation model:  $\hat{w}^{(i)} = \theta_{z_i} + \tau_i$  where  $\{z_i\}_{i=1}^{(1-\alpha)m} \in [K]$  are unknown labels and  $\{\theta_i\}_{i=1}^K$  are the unknown means of the sub-gaussian distribution. We denote  $\{\tau_i\}_{i=1}^{(1-\alpha)m}$  as independent and zero mean sub-gaussian noise with parameter  $\sigma$ . We propose and analyze a robust clustering algorithm presented in Algorithm 2. At iteration  $s$ , let  $z_i^{(s)}$  be the label of the  $i$ -th data point, and  $\hat{\theta}_g^{(s)}$  for  $g \in [K]$  be the estimate of the centers.

In Algorithm 2, we retain the nearest neighbor assignment of the Lloyd algorithm but change the sample mean estimate to a robust mean estimate using geometric median-based trimming.

We now introduce a few new notations. Let  $\Delta := \min_{g \neq h \in [K]} \|\theta_g - \theta_h\|$  denote the minimum separation between clusters. The worst case error in the centers are determined by  $\Lambda_s = \max_{h \in [K]} \|\hat{\theta}_h^{(s)} - \theta_h\|/\Delta$ . Consequently we define  $G_s$  as the maximum fraction of misclustered points in a cluster (maximized over all clusters). In Section 5.2 and A.3 (of the supplementary material), these quantities are formally defined along with the initialization condition,  $\Lambda_0$ .

Recall that  $|\mathcal{C}_i| = M_i$  and note that from Theorem 7, when Algorithm 2 is run for a constant  $S$  number of iterations, we get  $G_S \leq \varrho$  with high probability. Also, let  $\tilde{\alpha}_i = \left(\frac{\varrho M_i + \alpha m}{M_i + \alpha m}\right)$ . Since  $G_S$  denotes fraction of non-Byzantine machines that are misclustered,  $\tilde{\alpha}_i$  denotes the worst case fraction of Byzantine machines in cluster  $i$ . We assume  $\max_{i \in [K]} \tilde{\alpha}_i < \frac{1}{2}$ .

**Theorem 2.** *Suppose Assumptions 2, 3, 4 and 6 hold along with the separation and initialization conditions (Assumptions 8 of Section 5). Furthermore, suppose Algorithm 1 is run with “Trimmed  $K$  means” (Algorithm 2) for stage II for a constant  $S$  iterations; and the trimmed mean algorithm (of [7]) for stage III for  $T$  iterations with constant step-size of  $1/L_1$ . Then, provided  $T \geq \tilde{\mathcal{O}}\left(\frac{L_1 + \lambda_F}{\lambda_F}\right)$ , for all  $i \in [K]$ , we have*

$$\|\hat{w}_i - w_i^*\| \leq \tilde{\mathcal{O}}\left(\frac{\tilde{\alpha}_i d}{\sqrt{n}} + \frac{d}{\sqrt{n M_i}}\right).$$

with probability at least  $1 - m^{-10} - \mathcal{O}\left(\frac{d}{(1+nM_i)^d}\right)$ .

**Remark 3.** *Like before, we can remove Assumption 2 and obtain guarantee on  $F_i(\cdot)$  for all  $i \in [K]$ .*

*Comparison with the oracle:* Recall that the oracle knows the cluster labels of all the non-Byzantine machines. Hence, the worst case fraction of Byzantine machines will be  $\hat{\alpha}_i = \frac{\alpha m}{M_i + \alpha m}$ . Consequently, we observe that the obliviousness of the clustering identity hurts by a factor of  $(\tilde{\alpha}_i - \hat{\alpha}_i) \frac{d}{\sqrt{n}} = \left(\frac{\varrho M_i}{M_i + \alpha m}\right) \frac{d}{\sqrt{n}}$  in the precision of learning weight  $w_i^*$ . A few remarks are in order.

**Remark 4.** *As seen in Section B.2, if  $K = 2$  and  $\theta_1 = -\theta_2$ , we show that  $\varrho = 0$  if “Trimmed  $K$  means” is run for at least  $3 \log m$  iterations provided  $\frac{\Delta}{\sigma} \geq C\sqrt{\log m}$ . Hence our precision bound matches perfectly with the oracle bound.*

**Remark 5.** *The dependence on  $d$  can be improved if iterative filtering algorithm ([8]) is used in stage III of the modular algorithm. We get  $\|\hat{w}_i - w_i^*\| \leq \tilde{\mathcal{O}}\left(\frac{\sqrt{\tilde{\alpha}_i}}{\sqrt{n}} + \frac{\sqrt{d}}{\sqrt{n M_i}}\right)$  with high probability.*

## 4.1 Oracle optimality

In the presence of the oracle, our problem decomposes to  $K$  homogeneous ones. We study the dependence of the estimation error of Theorem 2 on  $n, d, \alpha$ , and  $M_i$  under such a setting.

**Dependence on  $(n, M_i, \alpha)$ :** We compare our results with the lower bounds presented in [7, Observation 1] assuming  $d$  is constant. It is immediate that the dependence on  $n$  and  $M_i$  is optimal. To see the dependence on  $\alpha$ , we first consider the special case of  $K = 2$  with centers  $\theta_1 = -\theta_2$ . Here  $\tilde{\alpha}_i = \frac{\alpha m}{M_i + \alpha m}$ . Typically,  $M_i \gg \alpha m$  and hence  $\tilde{\alpha}_i \approx \frac{\alpha m}{M_i}$ . Comparing with the bound in [7, Observation 1], the dependence on  $\alpha$  is near optimal in this case. However for a  $K$  cluster setting,  $\tilde{\alpha}_i$  may not be linear in  $\alpha$  in general (since  $\varrho$  is not proportional to  $\tilde{\alpha}_i$ ).

**Dependence on dimension  $d$ :** In this setting, instead of running the trimmed mean algorithm as the distributed optimization subroutine, we run the iterative filtering algorithm of [8], and as shown in Remark 3, the dependence on  $d$  when compared with the lower bound of [7, Observation 1] is optimal. Note that in this case, the dependence on  $\tilde{\alpha}_i$  becomes sub-optimal.

## 5 Robust clustering

In Stage II of the modular algorithm, we cluster the local ERM,  $\hat{w}^{(i)}$  in the presence of Byzantine machines. To ease notation, we write  $y_i = \hat{w}^{(i)}$ . Recall that for non Byzantine data-points, we have  $y_i = \theta_{z_i} + \tau_i$ , with unknown labels  $\{z_i\}_{i=1}^{(1-\alpha)m} \in [K]$ , unknown centers  $\{\theta_i\}_{i=1}^K$  and  $\sigma$  sub-Gaussian noise  $\{\tau_i\}_{i=1}^{(1-\alpha)m}$ . For Byzantine data points  $y_i$  is arbitrary. It is worth mentioning here that the classical Lloyd can be arbitrarily bad since the adversary may put the data points far away, thus causing the sample mean-based subroutine of the algorithm to fail. As a performance measure, we define the fraction of misclustered non-Byzantine data points at iteration  $s$  as,  $A_s = \frac{1}{(1-\alpha)m} \sum_{i \in \mathcal{M}} I\{z_i^{(s)} \neq z_i\}$ , where  $\mathcal{M}$  denotes the set of non-Byzantine data points with  $|\mathcal{M}| = (1-\alpha)m$ . We first concentrate the special case where  $K = 2$  with centers  $\theta^*$  and  $-\theta^*$ , and hence  $y_i = z_i \theta^* + \tau_i$ . With slight abuse in notation, the labels are  $z_i \in \{-1, +1\}$  and hence,  $z_i y_i = \theta^* + z_i \tau_i = \theta^* + \xi_i$ , where  $\xi_i \sim \mathcal{N}(0, \sigma^2 I_d)$ . This can be thought of estimating  $\theta^*$  from samples  $z_i y_i$ .

### 5.1 Symmetric 2 clusters with Gaussian mixture

We analyze Algorithm 2 in the above-mentioned setting. The performance depends on the normalized signal-to-noise ratio,  $r := \|\theta^*\| / (\sigma \sqrt{1 + \eta})$ , where  $\eta = 9d/(1-\alpha)m$ . At iteration  $s$ , let  $\beta$  be the fraction of data-points being trimmed by Algorithm 2 and let  $\hat{\theta}^{(s)}$  be the estimate of  $\theta^*$ .

**Assumption 7.** (i) (SNR) We have  $\frac{\|\theta^*\|}{\sigma} \geq C_{th}$  and  $m \geq m_{th}$  (ii) (Initialization)  $A_0 < \frac{1}{2} - \frac{2.56}{r} - \frac{1}{2\sqrt{(1-\alpha)m}} - \frac{\varepsilon}{2}$ , where  $m_{th}$ ,  $C_{th}$  are sufficiently large and  $\varepsilon$  is sufficiently small constants.

Hence we require a constant SNR and  $A_0$  needs to be slightly better than a random guess.

**Theorem 3.** Suppose Assumptions 6 and 7 hold. For  $\alpha \leq \beta < c/d$  and for  $s \geq 0$ ,  $A_s$  satisfies

$$A_{s+1} \leq A_s \left( A_s + \frac{8}{r^2} \right) + \frac{2}{r^2} + \sqrt{4 \log((1-\alpha)m) / ((1-\alpha)m)}$$

with probability at least  $1 - c_1 m^{-3} - c_2 m \exp(-d) - 2 \exp(-\|\theta^*\|^2 / 3\sigma^2)$ . Furthermore, for  $s \geq 3 \log m$ ,  $A_s \leq \exp(-\|\theta^*\|^2 / 16\sigma^2)$  with high probability.

Hence, if  $\|\theta^*\|/\sigma \gtrsim \sqrt{\log m}$ , then after  $3 \log m$  steps,  $A_s < \frac{1}{(1-\alpha)m}$  implying  $A_s = 0$ , which matches the oracle bound ( $\varrho = 0$ ) mentioned after Theorem 2. Also, here we can tolerate  $\alpha \sim 1/d$ , which can be prohibitive for large  $d$ . In the general  $K$ -cluster case, we improve the tolerance level from  $1/d$  to  $1/\sqrt{d}$  (Theorem 4), and in Section 5.3 we completely remove the dependence on  $d$ .

### 5.2 $K$ clusters with sub-Gaussian mixture

We now analyze the general  $K$ -cluster setting and with sub-Gaussian noise. The details of this section are deferred to Section A.3 of the Appendix. Similar to  $A_s$ , we define a cluster-wise misclustering fraction  $G_s$  and the trimmed cluster-wise misclustering fraction as  $G_s^u$  at iteration  $s$ . Recall the definition of  $\Delta$  and  $\Lambda_0$  from Section 4 and denote the minimum cluster size at iteration  $s$  as  $\gamma_1$ . Also define  $\alpha_h$  and  $\beta_h$  as the fraction of adversaries and trimmed points respectively for the  $h$ -th cluster. Furthermore, let  $\alpha'$  be the maximum adversarial fraction (after trimming) in a cluster and  $r_1 = (\Delta/\sigma) \sqrt{\gamma_1 / (1 + \frac{Kd}{(1-\alpha)m})}$  be the normalized SNR.

**Assumption 8.** We have: (a)  $(1 - \alpha)m\gamma_1^2 \geq C_1 K \log((1 - \alpha)m)$ ; (b) (SNR)  $\Delta \geq C_3 \sigma \sqrt{K}$ ; and (c) (Initialization)  $\Lambda_0 \leq \frac{1}{2} - \frac{4}{\sqrt{\tau_1}} - \frac{\varsigma}{2}$ , for a small constant  $\varsigma$ .

Hence the separation (of means) is  $\mathcal{O}(\sqrt{K})$ , which matches the standard separation condition for non-adversarial clustering ([39]). Let  $\varrho = \Gamma'(c/r_1^2 + \sqrt{\frac{5K \log((1-\alpha)m)}{\gamma_1^2(1-\alpha)m}}) + \rho$ , where  $\Gamma' = \max_h \frac{1-\beta_h}{1-\alpha_h}$  and  $\rho = \max_h \frac{\beta_h}{1-\alpha_h}$ . We have the following result:

**Theorem 4.** With Assumption 8 and  $\alpha' \leq c/\sqrt{d}$ , the cluster-wise misclustering fraction  $G_S^{\mathcal{U}}$  satisfies

$$G_{s+1}^{\mathcal{U}} \leq \frac{C_5}{r_1^2} + \frac{2C_2}{r_1^2} G_s^{\mathcal{U}} + \frac{C_2 C_4}{r_1^2} G_s^{\mathcal{U}} + \sqrt{5K \log((1 - \alpha)m) / (\gamma_1^2(1 - \alpha)m)}$$

with probability exceeding  $1 - 2((1 - \alpha)m)^{-3} - \exp(-0.3(1 - \alpha)m) - \exp(-0.5(1 - \alpha)m)$ . Furthermore, if Algorithm 2 is run for a constant  $S$  iterations,  $G_S \leq \varrho$  with high probability.

### 5.3 Robust clustering in high dimension

In Sections 5.1 and 5.2, we see that the tolerable fraction of adversarial data-points decays fast with  $d$ , which makes Algorithm 2 unsuitable for large  $d$ . Here we analyze the symmetric 2-cluster setting only. However given ??, our analysis can be extended to general  $K$  cluster setting. We adapt a slightly different observation model:  $\{y_i\}_{i=1}^m$  are drawn i.i.d from the following Huber contamination model: with probability  $1 - \alpha$ ,  $y_i = \nu_i(\theta^* + \tau_i)$ , where  $\nu_i$  is a Rademacher random variable and  $\tau_i$  is a  $\zeta$ -sub-Gaussian random vector with zero mean, and is independent of  $\nu_i$ ; with probability  $\alpha$ ,  $y_i$  is drawn from an arbitrary distribution. We assume that the maximum eigenvalue of the covariance matrix of  $\tau_i$  is bounded. More specifically, we let  $\tilde{\sigma}^2 := \lambda_{\max}(\mathbb{E}[\tau_i \tau_i^\top])$ . We denote the distribution of  $y$  and  $\tau$  by  $\mathcal{D}_y$  and  $\mathcal{D}_\tau$ , respectively. Intuitively, with probability  $1 - \alpha$ ,  $y_i$  is an inlier, i.e., drawn from a mixture of two symmetric distributions, and with probability  $\alpha$ ,  $y_i$  is an outlier. The goal is to estimate  $\theta^*$  and find the correct labels (i.e.,  $\nu_i$ ) of the inliers. We propose Algorithm 3 where the total number of data points  $m$  is an integer multiple of the number of iterations  $T$ , and the algorithm uses the Iterative Filtering algorithm [40, 41, 42], denoted by `IterFilter` as a subroutine. The intuition of the iterative filtering algorithm is to use higher order statistics, such as the sample covariance to iteratively remove outliers.

---

#### Algorithm 3 Clustering with iterative filtering subroutine

---

**Require:** Observations  $y_1, y_2, \dots, y_m$ , initial guess  $\hat{\theta}^{(0)}$ , number of iterations  $T$ .

- 1: **for**  $s = 1, 2, \dots, T$  **do**
  - 2:   Label estimation:  $\hat{\nu}_i^{(t)} = \operatorname{argmin}_{\nu \in \{-1, 1\}} \|\nu y_i - \hat{\theta}^{(t-1)}\|^2$ ,  $i = \frac{(t-1)m}{T} + 1, \dots, \frac{tm}{T}$ .
  - 3:   Parameter estimation:  $\hat{\theta}^{(t)} = \text{IterFilter}(\{\hat{\nu}_i^{(t)} y_i\}, i = \frac{(t-1)m}{T} + 1, \frac{(t-1)m}{T} + 2, \dots, \frac{tm}{T})$ .
  - 4: **end for**
- 

The convergence guarantee of the algorithm is in Theorem 5. We start with the following assumption:

**Assumption 9.** We assume: (a) (Initialization)  $\|\hat{\theta}^{(0)} - \theta^*\|_2 \leq \frac{1}{2}\|\theta^*\|$  (b) (SNR)  $\|\theta^*\|/\zeta \geq C_1$  and (c) (Sample complexity)  $m \geq \frac{C_2}{\alpha}(d + \frac{1}{\alpha} \log(\frac{1}{\eta} \log(\frac{1}{\beta}))) \log(\frac{1}{\beta})$ .

We emphasize that the SNR requirement is standard and the initialization condition is slightly stronger than a random guess. Armed with the above assumption, we have the following result.

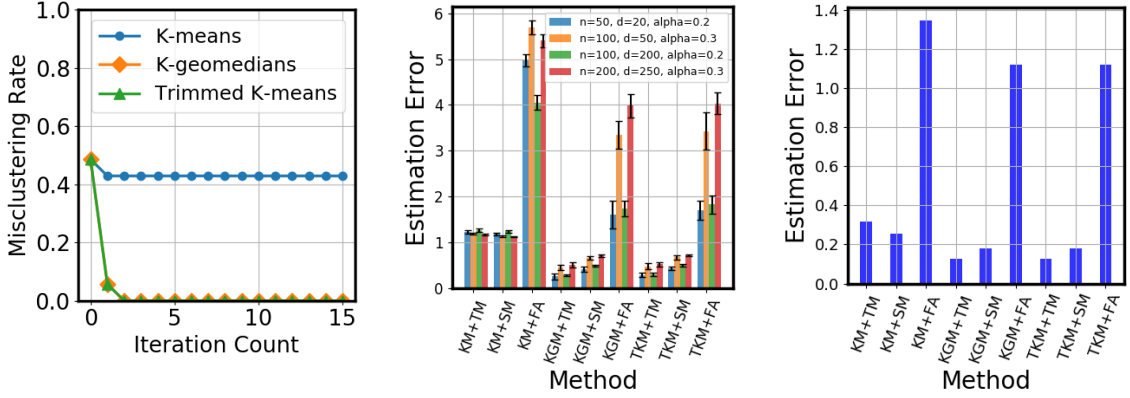
**Theorem 5.** Suppose that  $\alpha \leq 1/16$ ,  $\tilde{\sigma}/\zeta \leq C$ , and let  $\beta := 3\alpha + 8 \exp(-\|\theta^*\|^2/2\zeta^2)$ . With Assumption 9 and running Algorithm 3 for  $T = \Theta(\log(1/\beta))$  iterations, with probability at least  $1 - \eta$ , we have

$$\|\hat{\theta}^{(T)} - \theta^*\|_2 \leq C_3(\tilde{\sigma} + \zeta)(\sqrt{\alpha} + \exp(-\|\theta^*\|^2/4\zeta^2)), \quad \text{and}$$

$$\mathbb{P} \left\{ \operatorname{argmin}_{\hat{\nu} \in \{-1, 1\}} \|\hat{\nu} y - \hat{\theta}^{(T)}\|_2^2 \neq \nu \mid y \text{ is inlier} \right\} \leq C_4(\alpha + \exp(-\|\theta^*\|^2/2\zeta^2)).$$

Note that the tolerable level of  $\alpha$  has no dependence on dimension, which is an improvement over Theorem 3.





(a) Misclustering vs. iteration count (b) Synthetic data with  $m = 100$ ,  $K = 5, \sigma = 2$  (c) Yahoo! Learning to Rank Dataset

Figure 2: A comparison of K-means (KM), K-geomedians (KGM), and Trimmed K-means (TKM) in conjunction with Trimmed Mean robust optimization (TM), Sample Mean optimization (SM) and Federated Averaging (FA). In Figure 2a, we choose  $m = 100, d = 100, K = 5, \sigma = 3$ . The error bars in Figure 2b show the standard deviation over 20 trials.

## 6 Experiments

We perform extensive experiments on synthetic and real data and compare the performance of our algorithm to several non-robust clustering and/or optimization-based algorithms.

### 6.1 Synthetic data

For synthetic experiments, we use a mixture of linear regressions model. For each cluster, a  $d$  dimensional regression coefficient vector,  $\{w_i^*\}_{i=1}^K$ , is generated element-wise by a  $Bernoulli(\frac{1}{2})$  distribution. Then  $\lfloor (1 - \alpha)m \rfloor$  machines are uniformly assigned to the  $K$  clusters, and  $\lceil \alpha m \rceil$  machines are considered adversarial machines. For each good machine,  $j$  (belonging to cluster,  $i$ ),  $n$  data points are generated independently according to:  $x^{j,l} = \chi_l^T w_i^* + \tau_l$ , for all  $l \in [n]$ , where  $\chi_l \sim \mathcal{N}(0, I_d)$  and  $\tau_l \sim \mathcal{N}(0, \sigma^2)$ . For adversarial machines, the regression coefficients are sampled from  $3 * Bernoulli(\frac{1}{2})$ , resulting in outliers. We initialize the cluster assignments with 60 percent correct assignments for the good machines. We test the performance of Lloyd ( $K$ -means), Trimmed  $K$ -means (Algorithm 2) and  $K$ -geomedians (where the sample mean step of Lloyd is replaced by geometric median; note that this is Algorithm 2 excluding the trimming step). We set  $K = 5$  and  $m = 100$ . In Figure 2a, we see that the fraction of misclustered points (which we call misclustering rate) indeed diminishes with iteration at a fast rate which validates Theorem 4, whereas for  $K$ -means, it converges to a misclustering rate of 0.4.

We compare our algorithm consisting of robust clustering (using Trimmed  $K$ -means or  $K$ -geomedians) and robust distributed optimization with algorithms without robust subroutines in the clustering or the optimization step. In particular we use the classical  $K$ -means as a non robust clustering, and a naive sample averaging-based scheme (instead of robust trimmed mean-based scheme by [7]) as a non-robust, distributed algorithm. Also, in the robust optimization stage, we compare with a robust version of the Federated Averaging algorithm of [1] with 5 iterations of gradient descent in each worker node before the global model gets updated (by taking the trimmed mean of the local models in the worker nodes).

We first observe that the estimation error ( $\max_{i \in [K]} \|\hat{w}_i - w_i^*\|/\sqrt{d}$ ) for non-robust clustering schemes (KM in Figure 2b) is  $\geq 53\%$  higher than that using Trimmed  $K$ -means (TKM) and  $K$ -geomedians (KGM). Furthermore, trimmed mean-based distributed optimization (TM) strictly outperforms the sample mean-based (SM) optimization routine by  $\geq 29\%$  even with robust clustering. Federated averaging (FA) does orders of magnitude worse in estimation, likely due to the poor gradient updates provided by individual machines. Hence matching our theoretical intuition, robust clustering and robust optimization have the best performance in the presence of adversaries.

## 6.2 Yahoo! Learning to Rank dataset

The performance of the modular algorithm is evaluated on the Yahoo Learning to Rank dataset [43]. We use the `set2.test.txt` file for our experiment. We choose to treat the data as unsupervised, ignoring the labels for this simulation. Starting with 103174 queries and 595 features, we adopt the following thresholding rule: we draw an edge between the queries with  $\ell_2$  distance less than  $\gamma$  (which we optimize at 3.415). We then run a tree-search algorithm to detect the connected graphs which produce our true cluster assignments. Small groups are removed from the dataset. This results in 4 large clusters. Next, we take the mean of the features in each cluster to obtain  $w_1^*, \dots, w_4^*$ . The data points in each cluster is then split randomly in batches of 50 (hence,  $n = 50$ ). In addition, respecting  $\alpha = 0.3$ , 80 adversarial splits are incorporated via sampling 50 points randomly from the unused data and adding a  $Bernoulli(\frac{1}{2}) - 0.5$  vector to the ERM. Note, we synthetically perturb the data points primarily since it is hard to find datasets with explicit adversaries. We then compute the mean in each split (these can be thought analogous to local ERMs), and perform clustering on them using  $K$ -means, Trimmed  $K$ -means, and  $K$ -geomedians algorithms with fully random initialization. Then, we use trimmed mean, sample mean, or Federated Averaging optimization to estimate the  $w^*$  on each of the cluster assignment estimates with mean squared loss.

The results of the real data experiments are shown in Fig 2c. We see that Trimmed  $K$ -means in conjunction with trimmed mean optimization outperforms the other methods with an estimation error of 0.125. This algorithm is easy to implement and learns the optimal weights efficiently. On the other hand, the estimation error of  $K$ -means algorithm with sample mean optimization is 0.256, which is relatively two times worse than the robust algorithms. Also, Trimmed  $K$ -means and  $K$ -geomedians have similar final estimation error, which further confirms that trimming step after computing the geometric median may be redundant. Thus, we once again emphasize that our robust algorithm performs better than standard non-robust algorithms.

## 7 Conclusion and future work

We tackle the problem of robust Federated Learning in a heterogeneous environment. We propose a 3-step modular solution to the problem. For the second step, we analyze the classical Lloyd algorithm with a robust subroutine. Weakening the sub-Gaussian assumption along with a better initialization scheme are kept as our future endeavors.

## Acknowledgments

The authors would like to thank Swanand Kadhe and Prof. Peter Bartlett for helpful discussions.

## References

- [1] Brendan McMahan and Daniel Ramage. Federated learning: Collaborative machine learning without centralized training data. <https://research.googleblog.com/2017/04/federated-learning-collaborative.html>, 2017.
- [2] H Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, et al. Communication-efficient learning of deep networks from decentralized data. *arXiv preprint arXiv:1602.05629*, 2016.
- [3] Jakub Konečný, H Brendan McMahan, Daniel Ramage, and Peter Richtárik. Federated optimization: distributed machine learning for on-device intelligence. *arXiv preprint arXiv:1610.02527*, 2016.
- [4] Jiashi Feng, Huan Xu, and Shie Mannor. Distributed robust learning. *arXiv preprint arXiv:1409.5937*, 2014.
- [5] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. Byzantine-tolerant machine learning. *arXiv preprint arXiv:1703.02757*, 2017.
- [6] Yudong Chen, Lili Su, and Jiaming Xu. Distributed statistical machine learning in adversarial settings: Byzantine gradient descent. *arXiv preprint arXiv:1705.05491*, 2017.

- [7] Dong Yin, Yudong Chen, Ramchandran Kannan, and Peter Bartlett. Byzantine-robust distributed learning: Towards optimal statistical rates. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pages 5650–5659. PMLR, 10–15 Jul 2018.
- [8] Dong Yin, Yudong Chen, Kannan Ramchandran, and Peter Bartlett. Defending against saddle point attack in Byzantine-robust distributed learning. *arXiv preprint arXiv:1806.05358*, 2018.
- [9] Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S Talwalkar. Federated multi-task learning. In *Advances in Neural Information Processing Systems*, pages 4424–4434, 2017.
- [10] Sariel Har-Peled and Soham Mazumdar. On coresets for k-means and k-median clustering. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 291–300. ACM, 2004.
- [11] Moses Charikar, Sudipto Guha, Éva Tardos, and David B Shmoys. A constant-factor approximation algorithm for the k-median problem. *Journal of Computer and System Sciences*, 65(1):129–149, 2002.
- [12] Martin Zinkevich, Markus Weimer, Lihong Li, and Alex J Smola. Parallelized stochastic gradient descent. In *Advances in neural information processing systems*, pages 2595–2603, 2010.
- [13] Benjamin Recht, Christopher Re, Stephen Wright, and Feng Niu. Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In *Advances in neural information processing systems*, pages 693–701, 2011.
- [14] Ohad Shamir, Nathan Srebro, and Tong Zhang. Communication efficient distributed optimization using an approximate newton-type method. *CoRR*, abs/1312.7853, 2013.
- [15] Virginia Smith, Simone Forte, Chenxin Ma, Martin Takác, Michael I. Jordan, and Martin Jaggi. Cocoa: A general framework for communication-efficient distributed optimization. *CoRR*, abs/1611.02189, 2016.
- [16] Dong Yin, Ashwin Pananjady, Max Lam, Dimitris Papailiopoulos, Kannan Ramchandran, and Peter Bartlett. Gradient diversity: a key ingredient for scalable distributed learning. In *International Conference on Artificial Intelligence and Statistics*, pages 1998–2007, 2018.
- [17] Alexandros Pantelopoulos and Nikolaos G Bourbakis. A survey on wearable sensor-based systems for health monitoring and prognosis. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 40(1):1–12, 2010.
- [18] Parisa Rashidi and Diane J. Cook. Keeping the resident in the loop: Adapting the smart home to the user. *Trans. Sys. Man Cyber. Part A*, 39(5):949–959, September 2009.
- [19] Mehryar Mohri, Gary Sivek, and Ananda Theertha Suresh. Agnostic federated learning. *CoRR*, abs/1902.00146, 2019.
- [20] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated learning with non-iid data. *CoRR*, abs/1806.00582, 2018.
- [21] Felix Sattler, Simon Wiedemann, Klaus-Robert Müller, and Wojciech Samek. Robust and communication-efficient federated learning from non-iid data. *CoRR*, abs/1903.02891, 2019.
- [22] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018.
- [23] Anit Kumar Sahu, Tian Li, Maziar Sanjabi, Manzil Zaheer, Ameet Talwalkar, and Virginia Smith. On the convergence of federated optimization in heterogeneous networks. *CoRR*, abs/1812.06127, 2018.
- [24] Liping Li, Wei Xu, Tianyi Chen, Georgios B. Giannakis, and Qing Ling. RSA: byzantine-robust stochastic aggregation methods for distributed learning from heterogeneous datasets. *CoRR*, abs/1811.03761, 2018.

- [25] Dan Alistarh, Zeyuan Allen-Zhu, and Jerry Li. Byzantine stochastic gradient descent. *arXiv preprint arXiv:1803.08917*, 2018.
- [26] Cong Xie, Oluwasanmi Koyejo, and Indranil Gupta. Generalized Byzantine-tolerant SGD. *arXiv preprint arXiv:1802.10116*, 2018.
- [27] Liping Li, Wei Xu, Tianyi Chen, Georgios B Giannakis, and Qing Ling. Rsa: Byzantine-robust stochastic aggregation methods for distributed learning from heterogeneous datasets. *arXiv preprint arXiv:1811.03761*, 2018.
- [28] Ke Chen. A constant factor approximation algorithm for k-median clustering with outliers. In *Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 826–835. Citeseer, 2008.
- [29] Shalmoli Gupta, Ravi Kumar, Kefu Lu, Benjamin Moseley, and Sergei Vassilvitskii. Local search methods for k-means with outliers. *Proceedings of the VLDB Endowment*, 10(7):757–768, 2017.
- [30] Ravishankar Krishnaswamy, Shi Li, and Sai Sandeep. Constant approximation for k-median and k-means with outliers via iterative rounding. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 646–659. ACM, 2018.
- [31] Moses Charikar, Jacob Steinhardt, and Gregory Valiant. Learning from untrusted data. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 47–60. ACM, 2017.
- [32] Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.
- [33] Xinyang Yi, Constantine Caramanis, and Sujay Sanghavi. Alternating minimization for mixed linear regression. In *International Conference on Machine Learning*, pages 613–621, 2014.
- [34] Dong Yin, Ramtin Pedarsani, Yudong Chen, and Kannan Ramchandran. Learning mixtures of sparse linear regressions using sparse graph codes. *IEEE Transactions on Information Theory*, 65(3):1430–1451, 2018.
- [35] Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 928–936, 2003.
- [36] Stanislav Minsker. Geometric median and robust estimation in banach spaces. *Bernoulli*, 21(4):2308–2335, 11 2015.
- [37] Kevin Lai, Anup Rao, and S Vempala. Agnostic estimation of mean and covariance. *arXiv preprint arXiv:1604.06968*, 2016.
- [38] Ilias Diakonikolas, Gautam Kamath, Daniel M Kane, Jerry Li, Ankur Moitra, and Stewart Alistair. Being robust (in high dimensions) can be practical. *arXiv preprint arXiv:1703.00893*, 2018.
- [39] Amit Kumar and Ravindran Kannan. Clustering with spectral norm and the k-means algorithm. In *Foundations of Computer Science (FOCS), 2010 51st Annual IEEE Symposium on*, pages 299–308. IEEE, 2010.
- [40] Ilias Diakonikolas, Gautam Kamath, Daniel M Kane, Jerry Li, Ankur Moitra, and Alistair Stewart. Robust estimators in high dimensions without the computational intractability. In *Foundations of Computer Science (FOCS), 2016 IEEE 57th Annual Symposium on*, pages 655–664. IEEE, 2016.
- [41] Ilias Diakonikolas, Gautam Kamath, Daniel M Kane, Jerry Li, Ankur Moitra, and Alistair Stewart. Being robust (in high dimensions) can be practical. *arXiv preprint arXiv:1703.00893*, 2017.
- [42] Jacob Steinhardt, Moses Charikar, and Gregory Valiant. Resilience: A criterion for learning in the presence of arbitrary outliers. *arXiv preprint arXiv:1703.04940*, 2017.

- [43] Yahoo Learning to Rank Challenge (C-14) (<https://webscope.sandbox.yahoo.com/>).
- [44] Stéphane Boucheron, Gábor Lugosi, and Pascal Massart. *Concentration inequalities: A nonasymptotic theory of independence*. Oxford university press, 2013.
- [45] Yu Lu and Harrison H Zhou. Statistical and computational guarantees of lloyd’s algorithm and its variants. *arXiv preprint arXiv:1612.02099*, 2016.
- [46] Shai Shalev-Shwartz, Ohad Shamir, Nathan Srebro, and Karthik Sridharan. Stochastic convex optimization. In *COLT*, 2009.

## Appendix

### A Theoretical guarantees for Algorithm 2

#### A.1 Proof of Proposition 1

Given the parametric form of data generation, we first stack the covariates to form the matrix  $Z \in \mathbb{R}^{n \times d}$  where  $Z^\top := [\chi_1^\top \chi_2^\top \dots \chi_n^\top]$ . Also we form the vectors  $X_j^\top := [x^{i,1} x^{i,2} \dots x^{i,n}]$  and  $\Upsilon^T := [\Upsilon_1 \dots \Upsilon_n]$ . The objective is to estimate  $w_k^*$ . We run an ordinary least squares, i.e., we calculate the following,

$$\hat{w}^{(i)} = \arg \min_w \|X_j - Zw\|^2$$

From standard calculations, The ERM  $\hat{w}^{(i)}$  is given by

$$\hat{w}^{(i)} = (Z^\top Z)^{-1} Z^\top (Zw_k^* + \Upsilon) = w_k^* + (Z^\top Z)^{-1} Z^\top \Upsilon$$

Hence the distribution of  $\hat{w}^{(i)}$  is Gaussian. Since  $\mathbb{E}(\Upsilon_j) = 0$  for all  $j \in [n]$ ,  $\mathbb{E}(\hat{w}^{(i)}) = w_k^*$ .

#### A.2 Symmetric 2 cluster: proof of Theorem 3

Suppose after geometric median based trimming on both the centers at iteration  $s$ , we retain  $(1 - \beta)m$  data points.

Let  $\hat{\theta}^{(s)}$  be the estimate of  $\theta^*$  at iteration  $s$ . Let us fix a few notations here. At step  $s$ , we denote  $\mathcal{U}$  as the set of data-points that are not trimmed.  $\mathcal{T}$  denotes the set of trimmed points and  $\mathcal{B}$  denotes the set of adversarially corrupted data points. We have

$$\begin{aligned} \hat{\theta}^{(s)} &= \frac{1}{(1 - \beta)m} \sum_{i \in \mathcal{U}} z_i^{(s)} y_i \\ &= \frac{1}{(1 - \beta)m} \left[ \sum_{i \in \mathcal{M}} z_i^{(s)} y_i - \sum_{i \in \mathcal{M} \cap \mathcal{T}} z_i^{(s)} y_i + \sum_{i \in \mathcal{B} \cap \mathcal{U}} z_i^{(s)} y_i \right] \\ &= \frac{1}{(1 - \beta)m} \sum_{i \in \mathcal{M}} z_i^{(s)} y_i - T_1 + T_2 \end{aligned} \tag{1}$$

where,  $T_1 = \frac{1}{(1 - \beta)m} \sum_{i \in \mathcal{M} \cap \mathcal{T}} z_i^{(s)} y_i$  and  $T_2 = \frac{1}{(1 - \beta)m} \sum_{i \in \mathcal{B} \cap \mathcal{U}} z_i^{(s)} y_i$ . Consequently

$$\begin{aligned} \hat{\theta}^{(s)} - \theta^* &= \frac{1}{(1 - \beta)m} \sum_{i \in \mathcal{M}} z_i^{(s)} y_i - T_1 + T_2 - \theta^* \\ &= \frac{1}{(1 - \beta)m} \sum_{i \in \mathcal{M}} z_i^{(s)} y_i - \frac{1}{(1 - \beta)m} \sum_{i \in \mathcal{M}} z_i y_i + \frac{1}{(1 - \beta)m} \sum_{i \in \mathcal{M}} z_i y_i - T_1 + T_2 - \theta^*. \end{aligned}$$

Since  $z_i y_i = \theta^* + \xi_i$ , where  $z_i$  are the true label of the  $i$ -th data point, we have the following relation

$$z_i^{(s)} - z_i = -2I\{z_i^{(s)} \neq z_i\} z_i$$

Let  $\gamma := \frac{1-\alpha}{1-\beta}$ . Plugging in, we get

$$\begin{aligned}
\hat{\theta}^{(s)} - \theta^* &= \frac{1}{(1-\beta)m} \sum_{i \in \mathcal{M}} -2I\{z_i^{(s)} \neq z_i\}(\theta^* + \xi_i) + \frac{1}{(1-\beta)m} \sum_{i \in \mathcal{M}} z_i y_i - T_1 + T_2 - \theta^* \\
&= \frac{1}{(1-\beta)m} \left[ \sum_{i \in \mathcal{M}} -2I\{z_i^{(s)} \neq z_i\}(\theta^* + \xi_i) + \sum_{i \in \mathcal{M}} (\theta^* + \xi_i) \right] - T_1 + T_2 - \theta^* \\
&= \gamma \frac{1}{(1-\alpha)m} \left[ \sum_{i \in \mathcal{M}} -2I\{z_i^{(s)} \neq z_i\}(\theta^* + \xi_i) + \sum_{i \in \mathcal{M}} \xi_i \right] - T_1 + T_2 + (\gamma - 1)\theta^*. \tag{2}
\end{aligned}$$

We need a few definitions to proceed further. Recall  $A_s = \frac{1}{(1-\alpha)m} \sum_{i \in \mathcal{M}} I\{z_i^{(s)} \neq z_i\}$  denote the average error rate over good samples. Also define  $R := \frac{1}{(1-\alpha)m} \sum_{i \in \mathcal{M}} \xi_i$  and  $\bar{\tau} := \frac{1}{(1-\alpha)m} \sum_{i \in \mathcal{M}} \xi_i$ . With the above definitions, we get

$$\hat{\theta}^{(s)} - \theta^* = \gamma(-2A_s\theta^* - 2R + \bar{\tau}) - T_1 + T_2 + (\gamma - 1)\theta^*$$

As a result

$$\langle \hat{\theta}^{(s)}, \theta^* + \xi_i \rangle = \gamma \langle \theta^* + \xi_i, (1 - 2A_s)\theta^* - 2R + \bar{\tau} \rangle - \langle \theta^* + \xi_i, T_1 - T_2 \rangle$$

The extra term  $\langle \theta^* + \xi_i, T_1 - T_2 \rangle$  can be controlled using Cauchy Schwartz inequality in the following way

$$\langle \theta^* + \xi_i, T_1 - T_2 \rangle \leq \|\theta^* + \xi_i\| \|T_1 - T_2\| \leq (\|\theta^*\| + \|\xi_i\|)(\|T_1\| + \|T_2\|)$$

where the last inequality follows from triangle inequality. Furthermore

$$\|T_2\| \leq \frac{\alpha}{1-\beta} \max_{i \in \mathcal{U}} \|y_i\| \text{ and}$$

$$\|T_1\| \leq \frac{\beta}{1-\beta} \max_{i \in \mathcal{M}} \|y_i\|.$$

As a result

$$\|T_2\| + \|T_1\| \leq \frac{1}{1-\beta} \left( \beta \max_{i \in \mathcal{M}} \|y_i\| + \alpha \max_{i \in \mathcal{U}} \|y_i\| \right) \tag{3}$$

Let us first control the second term of the above equation. Let  $\mu_{geo}$  denote the geometric median of the data points. From Algorithm 2, for all  $i \in \mathcal{U}$ , we have

$$\|y_i - \mu_{geo}\| \leq C\sigma\sqrt{d} \Rightarrow \|y_i - \theta^* + \theta^* - \mu_{geo}\| \leq C\sigma\sqrt{d}$$

Invoking [36, Theorem 3.1], we obtain

$$\|\mu_{geo} - \theta^*\| \leq C_1\sigma\sqrt{d}$$

with probability at least  $1 - c\exp(-m)$ . Now, using a triangle inequality, we obtain

$$\|y_i - \theta^*\| \leq (C + C_1)\sigma\sqrt{d}$$

which upon further modification yields

$$\|y_i\| \leq \|\theta^*\| + (C + C_1)\sigma\sqrt{d}$$

For the first term of Equation 3, we just substitute  $y_i = z_i\theta^* + w_i$  to obtain

$$\max_{i \in \mathcal{M}} \|y_i\| = \|\theta^*\| + \max_{i \in \mathcal{M}} \|\xi_i\|$$

Now  $\xi_i$  is a gaussian random vector in dimension  $d$  with i.i.d gaussian coordinates. Therefore, the distribution of squared norm  $\|\xi_i\|^2 \sim \chi_d^2$  (a Chi-squared random variable of degree  $d$ ). Also, since

$\|\xi_i\| \geq 0$ , the term  $\max_{i \in \mathcal{M}} \|\xi_i\|^2 = (\max_{i \in \mathcal{M}} \|\xi_i\|)^2$ . From  $\chi_d^2$  concentration ([44]) and taking a union bound, for  $t \in (0, 1)$ , we get

$$\mathbb{P}\{\max_{i \in \mathcal{M}} \|\xi_i\|^2 \geq d(\sigma^2 + t)\} \leq (1 - \alpha)m \exp(-dt^2)$$

Therefore with probability at least  $1 - (1 - \alpha)m \exp(-dt^2)$ , the quantity  $\max_{i \in \mathcal{M}} \|\xi_i\| \leq \sqrt{d}(\sigma + \sqrt{t})$ . From the same  $\chi_d^2$  concentration, we get  $\|\xi_i\| \leq \sqrt{d}(\sigma + \sqrt{t_1})$  with probability at least  $1 - \exp(-dt_1^2)$  for  $t_1 \in (0, 1)$ . Substituting, and using  $\alpha \leq \beta$ , we obtain

$$\begin{aligned} \langle \theta^* + \xi_i, T_1 - T_2 \rangle &\leq \frac{2\beta}{1 - \beta} \left( \|\theta^*\|^2 + C_1 \sqrt{d} \sigma \|\theta^*\| + C_2 \sigma^2 d \right) \\ &\leq \frac{2\beta}{1 - \beta} \|\theta^*\|^2 \left( 1 + \frac{C_1 \sigma \sqrt{d}}{\|\theta^*\|} + \frac{C_2 \sigma^2 d}{\|\theta^*\|^2} \right). \end{aligned}$$

Since  $\alpha \leq \beta \leq \frac{c}{d}$ , we have

$$\langle \theta^* + \xi_i, T_1 - T_2 \rangle \leq \|\theta^*\|^2 \left( \frac{2c}{d(1 - c/d)} + \frac{2cC_1}{r\sqrt{d}(1 - c/d)} + \frac{2cC_2}{r^2(1 - c/d)} \right)$$

Define  $\Gamma := \left( \frac{2c}{d(1 - c/d)} + \frac{2cC_1}{r\sqrt{d}(1 - c/d)} + \frac{2cC_2}{r^2(1 - c/d)} \right)$ . We have,

$$\langle \theta^* + \xi_i, T_2 - T_1 \rangle \leq \Gamma \|\theta^*\|^2$$

With Assumption 7, we get

$$\Gamma \leq \left( \frac{2c}{d(1 - c/d)} + \frac{2cC_1}{C_{th}\sqrt{d}(1 - c/d)} + \frac{2cC_2}{C_{th}^2(1 - c/d)} \right)$$

Also, from Lemma 4 applied to the set  $S = \{i \in [(1 - \alpha)m] : I\{z_i^{(s+1)} \neq z_i^{(s)}\} = 1\}$ , we get  $\|R\| \leq \frac{\|\theta^*\|}{r} \sqrt{2A_s}$ . Invoking Lemma 5, we have

$$\langle 2R - \bar{\tau}, \theta^* \rangle \leq \frac{\|\theta^*\|^2}{r} \sqrt{2A_s} + \frac{\|\theta^*\|^2}{\sqrt{(1 - \alpha)m}}$$

Now, we analyze the error in the  $s + 1$ -th step. The iteration is still the nearest neighbor assignment. Hence

$$z_i^{(s+1)} = \operatorname{argmin}_{r \in \{-1, +1\}} \|ry_i - \hat{\theta}^{(s)}\|^2 = \operatorname{argmax}_{r \in \{-1, +1\}} \langle ry_i, \hat{\theta}^{(s)} \rangle = \operatorname{argmax}_{r \in \{-1, +1\}} \langle \theta^* + \xi_i, \hat{\theta}^{(s)} \rangle.$$

From this, the term  $I\{z_i^{(s+1)} \neq z_i^{(s)}\} = I\{\langle \theta^* + \xi_i, \hat{\theta}^{(s)} \rangle \leq 0\}$ . Using the above calculation

$$I\{z_i^{(s+1)} \neq z_i^{(s)}\} \leq I\left\{ \gamma \left( \Omega_0 \|\theta^*\|^2 + \langle \xi_i, \theta^* \rangle + \langle \xi_i, -2A_s \theta^* + 2R - \bar{\tau} \rangle \right) \leq 0 \right\}. \quad (4)$$

where  $\Omega_0 = 1 - 2A_s - \frac{2\sqrt{2A_s}}{r} - \frac{1}{\sqrt{(1 - \alpha)m}} - \frac{\Gamma}{\gamma}$ . A naive upper bound on  $\Omega_0$  is the following

$$\beta_0 \geq 1 - 2A_s - \frac{2}{r} - \frac{1}{\sqrt{(1 - \alpha)m}} - \frac{\Gamma}{\gamma}.$$

From the expression of  $\Gamma$  and using  $\beta \geq \alpha$ , we obtain

$$\frac{\Gamma}{\gamma} \leq \left( \frac{2c}{d(1 - c/d)} + \frac{2cC_1}{C_{th}\sqrt{d}(1 - c/d)} + \frac{2cC_2}{C_{th}^2(1 - c/d)} \right).$$

Also, we choose  $C_{th}$  sufficiently large to ensure that  $\frac{\Gamma}{\gamma} \leq \varepsilon$  with high probability, where  $\varepsilon$  is a (small) positive constant.

Since  $\gamma > 0$ , we can drop it inside the indicator function of Equation 4. Now we are ready to prove the theorem.

### A.2.1 Proof of the first part

For  $a, b \in \mathbb{R}$  and  $c > 0$ , we use the following inequality on the indicator function

$$I\{a + b \leq 0\} \leq I\{a \leq c\} + I\{b \leq -c\} \leq I\{a \leq c\} + \frac{b^2}{c^2}.$$

Using this, we have

$$I\{z_i^{(s+1)} \neq z_i^{(s)}\} \leq I\{\Omega \|\theta^*\|^2 \leq -\langle \xi_i, \theta^* \rangle\} + \frac{\langle \xi_i, 2R - \bar{\tau} - 2A_s \theta^* \rangle^2}{\delta^2 \|\theta^*\|^4}$$

where we define  $\Omega = \Omega_0 - \delta$  with  $\delta = \frac{3.12}{r}$ . We now take the average over all good data points and obtain  $A_s \leq I_1 + I_2$  where

$$I_1 = \frac{1}{(1-\alpha)m} \sum_{i=1}^{(1-\alpha)m} I\{\langle \xi_i, \theta^* \rangle \leq -\Omega \|\theta^*\|^2\} \quad (5)$$

$$I_2 = \frac{1}{(1-\alpha)m\delta^2 \|\theta^*\|^4} \sum_{i=1}^{(1-\alpha)m} \langle \xi_i, 2R - \bar{\tau} - 2A_s \theta^* \rangle^2 \quad (6)$$

*Upper Bound on  $I_1$ :*

We define  $\eta_a = 1 - 2a - \frac{5.12}{r} - \frac{1}{\sqrt{(1-\alpha)m}} - \frac{\Gamma}{\gamma}$ , and  $a \in \mathcal{D}$ , where  $\mathcal{D}$  is the set of discrete values  $A_s$  can take. We have

$$\mathcal{D} = \left\{ \frac{1}{(1-\alpha)m}, \frac{2}{(1-\alpha)m}, \dots, \frac{\lfloor (1-\alpha)m/2 \rfloor}{(1-\alpha)m} \right\}.$$

With this, we observe that  $I\{\langle \xi_i, \theta^* \rangle \leq -\eta_a \|\theta^*\|^2\}$  for all  $i \in [(1-\alpha)m]$  are independent Bernoulli random variable. Using Hoeffding's inequality (see e.g. Theorem 3.2 of [45]), we get the following upper bound on  $I_1$

$$I_1 \leq \exp\left(-\frac{\eta_{A_s}^2 \|\theta^*\|^2}{2\sigma^2}\right) + \sqrt{\frac{4 \log((1-\alpha)m/2)}{(1-\alpha)m}} \quad (7)$$

with probability at least  $1 - \frac{1}{((1-\alpha)m)^3}$ .

*Upper Bound on  $I_2$ :* We replace the parameter  $m$  by  $m' := (1-\alpha)m$ , since this is the effective sample size. We follow [45] (Theorem 3.2) and use Lemma 6 with  $m'$ . We get the following bound

$$I_2 \leq \frac{8}{r^2} A_s + \frac{1}{r^2} + \frac{1}{(1-\alpha)m} + A_s^2 \quad (8)$$

with probability exceeding  $1 - 2(1-\alpha)m \exp(-C_1 d) - \exp(-C_2 d) - 2 \exp(-\frac{\|\theta^*\|^2}{3\sigma^2})$ .

Combining all the pieces

$$A_{s+1} \leq \exp\left(-\frac{\eta_{A_s}^2 \|\theta^*\|^2}{2\sigma^2}\right) + \sqrt{\frac{4 \log((1-\alpha)m/2)}{(1-\alpha)m}} + \frac{8}{r^2} A_s + \frac{1}{r^2} + \frac{1}{(1-\alpha)m} + A_s^2$$

Let  $p = \frac{1}{2} - \frac{2.56}{r} - \frac{1}{2\sqrt{(1-\alpha)m}} - \frac{\varepsilon}{2}$ . From above, if  $A_0 \leq p$ , and if the constants  $C'_{th}$  and  $m_{th}$  are sufficiently large, we show via induction argument that  $A_s \leq p$  for all  $s$  and furthermore, we have  $\eta_{A_s} \geq \frac{2\sqrt{\log r}}{r}$  for all  $s \geq 0$ . Plugging in, we get

$$A_{s+1} \leq A_s(A_s + \frac{8}{r^2}) + \frac{2}{r^2} + \sqrt{\frac{4 \log((1-\alpha)m)}{(1-\alpha)m}}$$

with probability at least  $1 - \frac{1}{(m(1-\alpha))^3} - 2(1-\alpha)m \exp(-C_1 d) - \exp(-C_2 d) - 2 \exp(-\frac{\|\theta^*\|^2}{3\sigma^2})$ .



### A.2.2 Proof of the second part

By the above bound on  $A_{s+1}$ , if  $C_{th}$  is sufficiently large,

$$A_{s+1} \leq \frac{1}{2}A_s + \frac{2}{r^2} + \sqrt{\frac{4 \log m}{m}}$$

Iterating, we get,  $A_s \leq \frac{4}{r^2} + 5\sqrt{\frac{\log m}{m}}$  for all  $s \geq \log m$ . We now analyze the mistake bound  $I\{z_i^{(s+1)} \neq z_i^{(s)}\}$  for  $s \geq \log m$  similar to the proof in Section 7.3 of [45], which yields the following. For a sufficiently large  $m_{th}$ , with probability at least  $1 - \frac{C_3}{m(1-\alpha)} - 2(1-\alpha)m \exp(-C_1d) - \exp(-C_2d) - C_4 \exp(-\frac{\|\theta^*\|^2}{16\sigma^2})$

$$A_s \leq \exp\left(-\frac{\|\theta^*\|^2}{16\sigma^2}\right)$$

### A.3 Details of $K$ -clusters with sub-Gaussian mixture

We introduce a few new notations; let  $\Delta = \min_{g \neq h \in [K]} \|\theta_g - \theta_h\|$  be the signal strength, i.e. the minimum separation between clusters. For cluster  $h$ , let  $\hat{\theta}_h^{(s)}$  be the estimate of  $\theta_h$  at iteration  $s$ . Define  $\lambda = \max_{g \neq h \in [K]} \|\theta_g - \theta_h\| / \Delta$ , the maximum signal strength relative to the minimum. Our error rate of the center estimates can be measured by  $\Lambda_s = \max_{h \in [K]} \frac{1}{\Delta} \|\hat{\theta}_h^{(s)} - \theta_h\|$ .

Let  $T_h^*$  and  $T_h^{(s)}$  be the set of nodes in the true cluster  $h$  and the estimated cluster  $h$  at step  $s$ , respectively. Now, we define  $S_{gh}^{(s)} = T_g^* \cap T_h^{(s)}$ , the set of nodes in cluster  $g$  estimated to be in cluster  $h$  at step  $s$ . We define the cardinality of these sets as  $m_h^* = |T_h^*|$ ,  $m_h^{(s)} = |T_h^{(s)}|$ ,  $m_{gh}^{(s)} = |S_{gh}^{(s)}|$ . For simplicity, we will omit the superscript  $(s)$  when working at a single step of the algorithm. Let  $\mathcal{B}$  represent the set of adversarial nodes, and recall that  $\alpha$  is the fraction of adversarial nodes, i.e.  $\alpha = \frac{|\mathcal{B}|}{m}$ . For cluster  $h$ , let  $\alpha_h^{(s)}$  be the fraction of adversarials in  $T_h^{(s)}$ . We define a cluster-wise mis-clustering fraction at iteration  $s$  as

$$G_s = \max_{h \in [K]} \left\{ \frac{\sum_{g \neq h \in [K]} m_{gh}^{(s)}}{(1 - \alpha_h)m_h^{(s)}}, \frac{\sum_{g \neq h \in [K]} m_{hg}^{(s)}}{m_h^*} \right\}. \quad (9)$$

Let the set of untrimmed nodes at a given time step be  $\mathcal{U}$ , and the set of trimmed nodes as  $\mathcal{T}$ . For cluster  $h$ ,  $\mathcal{U}_h, \mathcal{T}_h$  represent the untrimmed and trimmed sets relative to the geomedian of  $\hat{\theta}_h$  respectively. Now, let the superscript  $\mathcal{U}$  represent the set contained in the untrimmed set for a given cluster. For example,  $T_h^{\mathcal{U}} = T_h \cap \mathcal{U}_h$ , and  $S_{gh}^{\mathcal{U}} = T_h^{\mathcal{U}} \cap T_g^*$ . Similarly,  $m_h^{\mathcal{U}} = |T_h^{\mathcal{U}}|$  and  $m_{gh}^{\mathcal{U}} = |S_{gh}^{\mathcal{U}}|$ .

Let  $\beta$  be the fraction of nodes trimmed by the algorithm at a certain step, i.e.  $\beta = \frac{|\mathcal{T}|}{m}$ . For cluster  $h$ , let  $\beta_h$  be the fraction trimmed from the ball centered at  $\hat{\theta}_h$ . In addition, to express the adversarial fraction within an untrimmed set, let  $\alpha_h^{\mathcal{U}(s)}$  be the fraction of adversarials in  $T_h^{\mathcal{U}(s)}$ . We also define  $\alpha' := \max_h \alpha_h^{\mathcal{U}(s)}$ . We use  $\alpha'$  to in the initialization as well as to upper bound the fraction of mis clustered points.

With respect to our trimmed-mean algorithm, we define a modified trimmed cluster-wise mis-clustering rate at iteration  $s$  as

$$G_s^{\mathcal{U}} = \max_{h \in [K]} \left\{ \frac{\sum_{g \neq h \in [K]} m_{gh}^{\mathcal{U}(s)}}{(1 - \alpha_h^{\mathcal{U}})m_h^{\mathcal{U}(s)}}, \frac{\sum_{g \neq h \in [K]} m_{hg}^{(s)} + m_{hh}^{\mathcal{T}(s)}}{m_h^*} \right\}$$

Define the minimum cluster size as  $\gamma_1 = \min_{h \in [K]} \frac{m_h^*}{(1-\alpha)m}$ . Lastly, we define a normalized signal-to-noise ratio for  $K$  clusters as  $r_1 = \frac{\Delta}{\sigma} \sqrt{\frac{\gamma_1}{1 + \frac{Kd}{(1-\alpha)m}}}$ .

With the above notations, invoking Assumption 8 and Theorem 4 ensures an exponential decay of  $G_s$ .

### A.4 General $K$ cluster: proof of Theorem 4

We begin by analyzing the following two results which will be crucial to prove the theorem. Let  $\mathcal{E}$  be the event of the intersection of Lemma 7, 8, 9 and 10. We have,

**Lemma 1.** On event  $\mathcal{E}$ , if  $G_s^{\mathcal{U}} \leq \frac{1}{2}$ , then we have

$$\Lambda_s \leq \frac{3}{r_1} + \frac{2\alpha' C \sigma \sqrt{d}}{\Delta} + \min \left\{ \frac{3}{r_1} \sqrt{k G_s^{\mathcal{U}}} + 2G_s^{\mathcal{U}} \Lambda_{s-1}, \lambda G_s^{\mathcal{U}} \right\}$$

where  $\alpha'^{(s)} = \max_{h \in [k]} \alpha_h^{\mathcal{U}(s)}$ .

Next, we present a bound on the misclustering rate per iteration.

**Lemma 2.** On event  $\mathcal{E}$ , if  $\Lambda_s \leq \frac{1-\epsilon}{2}$  and  $r_1 \geq 36\epsilon^{-2}$ , then

$$G_{s+1}^{\mathcal{U}} \leq \frac{2}{\epsilon^4 r_1^2} + \left( \frac{28}{\epsilon^2 r_1} \Lambda_s \right)^2 + \sqrt{\frac{5k \log((1-\alpha)m)}{\gamma_1^2 (1-\alpha)m}}$$

#### A.4.1 Proof of Lemma 1

Let  $\bar{Y}_B = \frac{1}{|B|} \sum_{i \in B} y_i$ . We begin by expanding the error of estimated centers at timestep  $s$ . For some cluster  $h$ :

$$\begin{aligned} \hat{\theta}_h - \theta_h &= \frac{1}{m_h^{\mathcal{U}}} \left( \sum_{i \in S_{hh}^{\mathcal{U}}} (y_i - \theta_h) + \sum_{a \neq h} \sum_{i \in S_{ah}^{\mathcal{U}}} (y_i - \theta_h) + \sum_{i \in S_{Bh}^{\mathcal{U}}} (y_i - \theta_h) \right) \\ &= \frac{1}{m_h^{\mathcal{U}}} \sum_{i \in S_{hh}^{\mathcal{U}}} \tau_i + \sum_{a \neq h} \frac{m_{ah}^{\mathcal{U}}}{m_h^{\mathcal{U}}} (\bar{Y}_{S_{ah}^{\mathcal{U}}} - \theta_h) + \frac{1}{m_h^{\mathcal{U}}} \sum_{i \in S_{Bh}^{\mathcal{U}}} (y_i - \theta_h) \end{aligned}$$

By the label update step of the algorithm, we know  $\|y_i - \hat{\theta}_h^{(s-1)}\| \leq \|y_i - \hat{\theta}_a^{(s-1)}\|$  for any  $i \in S_{ah}$  and in turn  $i \in S_{ah}^{\mathcal{U}}$ . Taking the average we have

$$\|\bar{Y}_{S_{ah}^{\mathcal{U}}} - \hat{\theta}_h^{(s-1)}\| \leq \|\bar{Y}_{S_{ah}^{\mathcal{U}}} - \hat{\theta}_a^{(s-1)}\|$$

By the triangle inequality

$$\|\bar{Y}_{S_{ah}^{\mathcal{U}}} - \theta_h\| \leq \|\bar{Y}_{S_{ah}^{\mathcal{U}}} - \theta_a\| + \|\hat{\theta}_a^{(s-1)} - \theta_a\| + \|\hat{\theta}_h^{(s-1)} - \theta_h\|$$

Let  $W_B = \sum_{i \in B} \tau_i$ . By Lemma 7 and substituting in the definition of  $\Lambda_{s-1}$

$$\begin{aligned} \|\bar{Y}_{S_{ah}^{\mathcal{U}}} - \theta_h\| &\leq \|W_{S_{ah}^{\mathcal{U}}}\| + 2\Lambda_{s-1}\Delta \\ &\leq \frac{\sigma \sqrt{3((1-\alpha)m+d)}}{\sqrt{m_{ah}^{\mathcal{U}}}} + 2\Lambda_{s-1}\Delta \end{aligned}$$

We take a weighted sum over  $a \neq h \in [k]$  to get

$$\begin{aligned} \sum_{a \neq h} \frac{m_{ah}^{\mathcal{U}}}{m_h^{\mathcal{U}}} \|\bar{Y}_{S_{ah}^{\mathcal{U}}} - \theta_h\| &\leq \frac{\sigma \sqrt{3((1-\alpha)m+d)}}{m_h^{\mathcal{U}}} \sum_{a \neq h} \sqrt{m_{ah}^{\mathcal{U}}} + 2\Lambda_{s-1}\Delta \sum_{a \neq h} \frac{m_{ah}^{\mathcal{U}}}{m_h^{\mathcal{U}}} \\ &\leq \frac{\sigma \sqrt{3((1-\alpha)m+d)}}{\sqrt{m_h^{\mathcal{U}}}} \sqrt{(k-1)G_s^{\mathcal{U}}} + 2G_s^{\mathcal{U}} \Lambda_{s-1}\Delta \end{aligned}$$

To bound  $\|W_{S_{hh}^{\mathcal{U}}}\|$  we use the fact that  $W_{S_{hh}^{\mathcal{U}}} = W_{T_h^*} - \sum_{a \neq h} W_{S_{ha}} - W_{S_{hh}^{\mathcal{T}}}$ . By the triangle inequality and Lemma 7 and 9, we have

$$\begin{aligned} \|W_{S_{hh}^{\mathcal{U}}}\| &\leq \|W_{T_h^*}\| + \left\| \sum_{a \neq h} W_{S_{ha}} + W_{S_{hh}^{\mathcal{T}}} \right\| \\ &\leq 3\sigma \sqrt{(d + \log(1-\alpha)m)m_h^*} + \sigma \sqrt{3((1-\alpha)m+d)(m_h^* - m_{hh}^{\mathcal{U}})} \end{aligned}$$

By our trimmed mean algorithm, any node within the untrimmed ball is within  $C\sigma\sqrt{d}$  from the geometric median. Thus, in the worst case, the adversarial fraction will increase the error by  $2C\sigma\sqrt{d}$  each. So we can bound the error in the adversarial fraction like so

$$\frac{1}{m_h^{\mathcal{U}}} \left\| \sum_{i \in S_{\mathcal{B}_h}^{\mathcal{U}}} (y_i - \theta_h) \right\| \leq 2\alpha_h^{\mathcal{U}} C\sigma\sqrt{d}$$

With the condition  $G_s^{\mathcal{U}} \leq \frac{1}{2}$ , we can lower bound  $m_h^{\mathcal{U}}$ .

$$m_h^{\mathcal{U}} \geq m_{hh}^{\mathcal{U}} \geq m_h^*(1 - G_s^{\mathcal{U}}) \geq \frac{1}{2}m_h^* \geq \frac{1}{2}\gamma_1(1 - \alpha)m$$

Putting it all together, we have

$$\begin{aligned} \left\| \hat{\theta}_h^{(s)} - \theta_h \right\| &\leq \frac{3\sigma\sqrt{(d + \log(1 - \alpha)m)m_h^*}}{m_h^{\mathcal{U}}} + \frac{\sigma\sqrt{3((1 - \alpha)m + d)(m_h^* - m_{hh}^{\mathcal{U}})}}{m_h^{\mathcal{U}}} \\ &\quad + \frac{\sigma\sqrt{3((1 - \alpha)m + d)m_h^{\mathcal{U}}(k - 1)G_s^{\mathcal{U}}}}{m_h^{\mathcal{U}}} + 2G_s^{\mathcal{U}}\Lambda_{s-1}\Delta + 2\alpha_h^{\mathcal{U}}C\sigma\sqrt{d} \\ &\leq \frac{3\sigma\sqrt{d + \log(1 - \alpha)m}}{\sqrt{\gamma_1(1 - \alpha)m}} + \frac{3\sigma\sqrt{((1 - \alpha)m + d)(m_h^* - m_{hh}^{\mathcal{U}} + m_h^{\mathcal{U}}(k - 1)G_s^{\mathcal{U}})}}{m_h^{\mathcal{U}}} \\ &\quad + 2G_s^{\mathcal{U}}\Lambda_{s-1}\Delta + 2\alpha_h^{\mathcal{U}}C\sigma\sqrt{d} \\ &\leq \frac{3\sigma\sqrt{d + \log(1 - \alpha)m}}{\sqrt{\gamma_1(1 - \alpha)m}} + \frac{3\sigma\sqrt{k((1 - \alpha)m + d)G_s^{\mathcal{U}}}}{\sqrt{\gamma_1(1 - \alpha)m}} \\ &\quad + 2G_s^{\mathcal{U}}\Lambda_{s-1}\Delta + 2\alpha_h^{\mathcal{U}}C\sigma\sqrt{d} \\ &\leq \left( \frac{3}{r_1}(1 + \sqrt{kG_s^{\mathcal{U}}}) + 2G_s^{\mathcal{U}}\Lambda_{s-1} \right) \Delta + 2\alpha_h^{\mathcal{U}}C\sigma\sqrt{d} \end{aligned}$$

This gives us the first term in the right hand side (RHS) of the lemma. For the second term we start with a different decomposition of the center estimate.

$$\begin{aligned} \hat{\theta}_h &= \frac{1}{m_h^{\mathcal{U}}} \sum_{i=1}^m (\theta_{z_i} + \tau_i) \mathbb{1}\{\hat{z}_i = h \cap i \in \mathcal{U}_h\} + \frac{1}{m_h^{\mathcal{U}}} \sum_{i \in S_{\mathcal{B}_h}} y_i \\ &= \frac{1}{m_h^{\mathcal{U}}} \sum_{a=1}^k \sum_{i=1}^m \theta_a \mathbb{1}\{z_i = a \cap \hat{z}_i = h \cap i \in \mathcal{U}_h\} + \frac{1}{m_h^{\mathcal{U}}} W_{T_h^{\mathcal{U}}} + \frac{1}{m_h^{\mathcal{U}}} \sum_{i \in S_{\mathcal{B}_h}} y_i \\ &= \sum_{a=1}^k \frac{m_{ah}^{\mathcal{U}}}{m_h^{\mathcal{U}}} \theta_a + \frac{1}{m_h^{\mathcal{U}}} W_{T_h^{\mathcal{U}}} + \frac{1}{m_h^{\mathcal{U}}} \sum_{i \in S_{\mathcal{B}_h}} y_i \end{aligned}$$

We use this bound the error of the estimate.

$$\begin{aligned} \left\| \hat{\theta}_h^{(s)} - \theta_h \right\| &= \left\| \sum_{a=1}^k \frac{m_{ah}^{\mathcal{U}}}{m_h^{\mathcal{U}}} (\theta_a - \theta_h) + \frac{1}{m_h^{\mathcal{U}}} W_{T_h^{\mathcal{U}}} + \frac{1}{m_h^{\mathcal{U}}} \sum_{i \in S_{\mathcal{B}_h}} (y_i - \theta_h) \right\| \\ &\leq \left\| \sum_{a \neq h} \frac{m_{ah}^{\mathcal{U}}}{m_h^{\mathcal{U}}} (\theta_a - \theta_h) \right\| + \left\| \frac{1}{m_h^{\mathcal{U}}} W_{T_h^{\mathcal{U}}} \right\| + \left\| \frac{1}{m_h^{\mathcal{U}}} \sum_{i \in S_{\mathcal{B}_h}} (y_i - \theta_h) \right\| \end{aligned}$$

By the triangle inequality we can bound the first term as

$$\begin{aligned} \left\| \sum_{a \neq h} \frac{m_{ah}^{\mathcal{U}}}{m_h^{\mathcal{U}}} (\theta_a - \theta_h) \right\| &\leq \sum_{a \neq h} \frac{m_{ah}^{\mathcal{U}}}{m_h^{\mathcal{U}}} \|\theta_a - \theta_h\| \\ &\leq \lambda \Delta \sum_{a \neq h} \frac{m_{ah}^{\mathcal{U}}}{m_h^{\mathcal{U}}} \\ &\leq \lambda \Delta G_s^{\mathcal{U}} \end{aligned}$$

Using Lemma 7 and the above, we get

$$\begin{aligned}\|\hat{\theta}_h^{(s)} - \theta_h\| &\leq \lambda \Delta G_s^{\mathcal{U}} + \sigma \sqrt{\frac{3((1-\alpha)m+d)}{m_h^{\mathcal{U}}}} + 2\alpha_h^{\mathcal{U}} C \sigma \sqrt{d} \\ &\leq (\lambda G_s^{\mathcal{U}} + \frac{3}{r_1}) \Delta + 2\alpha_h^{\mathcal{U}} C \sigma \sqrt{d}\end{aligned}$$

Taking the min of the two terms, the proof is now complete.

#### A.4.2 Proof of Lemma 2

Arguing similar to the proof of Lemma A.7 of [45], we can begin at the result

$$m_{gh}^{\mathcal{U}(s+1)} \leq m_{gh}^{(s+1)} \leq \sum_{i \in T_g^*} \mathbb{1} \left\{ \frac{\epsilon^2}{4} \|\theta_g - \theta_h\|^2 \leq \langle \tau_i, \theta_h - \theta_g \rangle \right\} + \sum_{i \in T_g^*} \frac{16}{\epsilon^4 \Delta^4} (\tau_i^T (\Delta_h - \Delta_g))^2$$

Note,  $\Delta_h = \hat{\theta}_h^{(s)} - \theta_h$  for  $h \in [k]$ . By the Lemma 10, we can bound the first term in the RHS as

$$m_g^* \exp(-\frac{\epsilon^4 \Delta^2}{32\sigma^2}) + \sqrt{5m_g^* \log((1-\alpha)m)}$$

Using Lemma 8 we can bound the second term as well as

$$\sum_{i \in T_g^*} \frac{16}{\epsilon^4 \Delta^4} (\tau_i^T (\Delta_h - \Delta_g))^2 \leq \frac{96(m_g^* + d)\sigma^2}{\epsilon^4 \Delta^4} \|\Delta_g - \Delta_h\|^2$$

Together with the bound  $\|\Delta_g - \Delta_h\|^2 \leq 4\Lambda_s^2 \Delta^2$ , we get

$$m_{gh}^{\mathcal{U}(s+1)} \leq m_g^* \exp(-\frac{\epsilon^4 \Delta^2}{32\sigma^2}) + \sqrt{5m_g^* \log((1-\alpha)m)} + \frac{384(m_g^* + d)\sigma^2}{\epsilon^4 \Delta^2} \Lambda_s^2$$

We can take the max over all clusters to get

$$\max_{g \in [k]} \sum_{h \neq g} \frac{m_{gh}^{\mathcal{U}(s+1)}}{m_g^*} \leq k \exp(-\frac{\epsilon^4 \Delta^2}{32\sigma^2}) + k \frac{\sqrt{5 \log((1-\alpha)m)}}{m_g^*} + \frac{384\sigma^2}{\epsilon^4 r_1^2} \Lambda_s^2$$

Since  $\Lambda_s \leq \frac{1}{2}$  and  $r_1 \geq 20\epsilon^2$ , when  $\gamma_1(1-\alpha) \geq 32 \log((1-\alpha)m)$  we have

$$\min_{g \in [k]} \frac{m_{gg}^{\mathcal{U}}}{m_g^*} = 1 - \max_{g \in [k]} \sum_{h \neq g} \frac{m_{gh}^{\mathcal{U}(s+1)}}{m_g^*} \geq \frac{1}{2}$$

Thus, for some  $h \in [k]$ ,

$$(1 - \alpha_h^{\mathcal{U}}) m_h^{\mathcal{U}(s+1)} \geq m_{hh}^{\mathcal{U}(s+1)} \geq \frac{1}{2} m_h^* \geq \frac{1}{2} \gamma_1 (1-\alpha) m$$

We apply this to get

$$\max_{h \in [k]} \sum_{g \neq h} \frac{m_{gh}^{\mathcal{U}(s+1)}}{(1 - \alpha_h^{\mathcal{U}}) m_h^{\mathcal{U}(s+1)}} \leq \frac{2}{\gamma_1} \exp(-\frac{\epsilon^4 \Delta^2}{32\sigma^2}) + \sqrt{\frac{5k \log((1-\alpha)m)}{\gamma_1^2 (1-\alpha)m}} + \frac{768}{\epsilon^4 r_1^2} \Lambda_s^2$$

These, two bounds give us

$$G_{s+1}^{\mathcal{U}} \leq \frac{2}{\gamma_1} \exp(-\frac{\epsilon^4 \Delta^2}{32\sigma^2}) + \sqrt{\frac{5k \log((1-\alpha)m)}{\gamma_1^2 (1-\alpha)m}} + \frac{768}{\epsilon^4 r_1^2} \Lambda_s^2$$

Assuming  $\epsilon^4 \gamma_1 \Delta^2 / \sigma^2 \geq r_1^2 \epsilon^4 \geq 36$  we get the result

$$G_{s+1}^{\mathcal{U}} \leq \frac{2}{\epsilon^4 r_1^2} + \left( \frac{28}{\epsilon^2 r_1} \Lambda_s \right)^2 + \sqrt{\frac{5k \log((1-\alpha)m)}{\gamma_1^2 (1-\alpha)m}}$$

### A.4.3 Proof of final results

By Assumption 8, Lemma 1 parallels Lemma A.6 in [45] with an additional term on the order of a constant,  $c$ , where  $\alpha' \leq \frac{c}{\sqrt{d}}$ . Setting  $\epsilon = \frac{12}{\sqrt{r}}$ ,  $C_1 \geq 500$ , and under the assumption that  $r_1 \geq 160\sqrt{k}$  we can characterize  $\delta$ . For example, with  $\frac{2C_2\sigma}{\Delta} \leq \frac{1}{4}$ ,  $\delta = \frac{1}{2}$  suffices to guarantee  $G_s^{\mathcal{U}} \leq 0.35$  and  $\Lambda_s \leq \frac{1}{2} - \frac{c}{2}$  for all  $s$ . Thus, given an analogous recursive relationship between the two lemmas, we know the conditions for the lemmas will hold for any step  $s$  as long as

$$\Lambda_0 \leq \frac{1}{2} - \frac{4}{\sqrt{r_1}} - \delta c$$

or equivalently,

$$G_0^{\mathcal{U}} \leq \left(\frac{1}{2} - \frac{6}{\sqrt{r_1}} - \delta c\right) \frac{1}{\lambda}$$

holds for some constant  $\delta$  to combat the additional offset produced by the extra term. Along with Assumptions 8 with  $C_1 = 16$ , we can simplify Lemma 1, as

$$\Lambda_s \leq \frac{3}{r_1} + \frac{3}{r_1} \sqrt{kG_s^{\mathcal{U}} + G_s^{\mathcal{U}} + 2Cc} \frac{C_3}{r_1} \leq \frac{1}{2} + G_s^{\mathcal{U}} + \frac{CC_3c}{8}$$

Combining this with Lemma 2 with some constant  $C_2$  and  $C_4 = \frac{CC_3c}{8}$ , we get

$$\begin{aligned} G_{s+1}^{\mathcal{U}} &\leq \frac{C_2}{r_1^2} + \frac{C_2}{r_1^2} \left(\frac{1}{4} + G_s^{\mathcal{U}} + (G_s^{\mathcal{U}})^2 + \frac{C_4}{2} + C_4G_s^{\mathcal{U}} + C_4^2\right) + \sqrt{\frac{5k \log((1-\alpha)m)}{\gamma_1^2(1-\alpha)m}} \\ &\leq \frac{2C_2}{r_1^2} + \frac{2C_2}{r_1^2} G_s^{\mathcal{U}} + \frac{C_2C_4}{r_1^2} (G_s^{\mathcal{U}} + C_4) + \sqrt{\frac{5k \log((1-\alpha)m)}{\gamma_1^2(1-\alpha)m}} \end{aligned}$$

which upon further simplification yields the result.

**Error floor** From the above result,  $G_{s+1}^{\mathcal{U}}$  satisfy the following inequality

$$G_{s+1}^{\mathcal{U}} \leq \frac{c_1}{r_1^2} G_s^{\mathcal{U}} + \frac{c_2}{r_1^2} + \sqrt{\frac{5K \log((1-\alpha)m)}{\gamma_1^2(1-\alpha)m}}$$

For a sufficiently large  $C$ , we can write,

$$G_{s+1}^{\mathcal{U}} \leq \frac{c'_1}{r_1^2} G_s^{\mathcal{U}} + \frac{c'_2}{r_1^2} + \sqrt{\frac{5K \log((1-\alpha)m)}{\gamma_1^2(1-\alpha)m}}$$

with  $r_1^2 > c'_1$ . Let  $\varrho_1 := \frac{c'_2}{r_1^2} + \sqrt{\frac{5K \log((1-\alpha)m)}{\gamma_1^2(1-\alpha)m}}$  and  $\delta' = \frac{c'_1}{r_1^2}$ . We have

$$G_{s+1}^{\mathcal{U}} \leq \delta' G_s^{\mathcal{U}} + \varrho_1$$

Iterating this for  $S$  iterations, where  $S$  is a constant, we get

$$G_S^{\mathcal{U}} \leq (\delta')^S G_0^{\mathcal{U}} + \varrho (1 + \delta' + \dots + (\delta')^{S-1}) \leq \frac{1}{2} (\delta')^S + \frac{1}{1 - \delta'} \varrho_1 \quad (10)$$

We now relate  $G_s^{\mathcal{U}}$  to the untrimmed, cluster-wise misclustering rate  $G_s$ . To upper bound  $G_s$ , we only need to bound the first term of  $G_s$  because the second term of  $G_s$  strictly increases after trimming. Assume the first term dominates in  $G_s$ .

$$\begin{aligned} G_s &= \max_{h \in [K]} \frac{\sum_{g \neq h \in [K]} m_{gh}}{(1 - \alpha_h) m_h} \\ &= \max_{h \in [K]} \frac{\sum_{g \neq h \in [K]} m_{gh}^{\mathcal{U}} + m_h^{\mathcal{T}}}{(1 - \alpha_h) m_h} \\ &= \max_{h \in [K]} \frac{(1 - \alpha_h^{\mathcal{U}}) m_h^{\mathcal{U}}}{(1 - \alpha_h) m_h} \frac{\sum_{g \neq h \in [K]} m_{gh}^{\mathcal{U}}}{(1 - \alpha_h^{\mathcal{U}}) m_h^{\mathcal{U}}} + \frac{m_h^{\mathcal{T}}}{(1 - \alpha_h) m_h} \\ &\leq \max_{h \in [K]} \frac{1}{1 - \alpha_h} ((1 - \beta_h) G_s^{\mathcal{U}} + \beta_h) \end{aligned}$$

Thus, we can apply our error floor to  $G_s$  with an additional term.

$$G_S \leq \Gamma'(\frac{1}{2}(\delta')^S + \frac{1}{1-\delta'}\varrho_1) + \zeta$$

Now plug the values of  $\delta'$ , we get that for  $S \geq 2$ , the first term in Equation 10 is order-wise negligible compared to the second term. Hence, we get  $G_S \leq \varrho$  for  $S \geq 2$ .

### A.5 High dimension: proof of Theorem 5

For the  $t$ -th iteration, suppose that  $y_i$  is a good data point with label  $\nu_i$ . Since  $\hat{\nu}_i^{(t)} = \underset{\nu \in \{-1,1\}}{\operatorname{argmin}} \|\nu y_i - \hat{\theta}^{(t-1)}\|_2^2$ , we have  $I\{\hat{\nu}_i^{(t)} \neq \nu_i\} = I\{\langle \theta^* + \tau_i, \hat{\theta}^{(t-1)} \rangle \leq 0\}$ . Define the following probability

$$A_t := \mathbb{P}_{\tau \sim \mathcal{D}_\tau} \left\{ \langle \theta^* + \tau, \hat{\theta}^{(t-1)} \rangle \leq 0 \right\}. \quad (11)$$

We start by analyzing  $A_1$ . When  $t = 1$ , since  $\tau$  is  $\zeta$ -sub-Gaussian, we have

$$A_1 = \mathbb{P} \left\{ \langle \tau, \hat{\theta}^{(0)} \rangle \leq -\langle \theta^*, \hat{\theta}^{(0)} \rangle \right\} \leq \exp \left( -\frac{\langle \theta^*, \hat{\theta}^{(0)} \rangle^2}{2\zeta^2 \|\hat{\theta}^{(0)}\|_2^2} \right).$$

By simple algebra, one can check that for any two vectors  $\theta$  and  $\hat{\theta}$ , we have

$$\langle \theta, \frac{\hat{\theta}}{\|\hat{\theta}\|_2} \rangle^2 \geq \|\theta\|_2^2 - \|\theta - \hat{\theta}\|_2^2. \quad (12)$$

Combining with initialization of Assumption 9 we have

$$A_1 \leq \exp(-\frac{3}{8\zeta^2} \|\theta^*\|_2^2) \quad (13)$$

Due to the SNR condition of Assumption 9, we know that  $A_1 \leq \frac{1}{16}$ . For simplicity, let  $n_0 = \frac{n}{T}$  denote the number of data points used in each iteration. Thus the data points used in the  $t$ -th iteration are  $y_{(t-1)n_0+1}, y_{(t-1)n_0+2}, \dots, y_{tn_0}$ . We use induction to prove the following result: for any  $\delta > 0$ , suppose that  $n_0 \geq \frac{1}{\alpha}(d + \frac{1}{\alpha} \log(2/\delta))$ , then, with probability at least  $1 - 2t\delta$  over the data points used in the first  $t$  iterations, we have

$$A_{t+1} \leq \frac{1}{8}(A_t + 3\alpha) + \exp \left( -\frac{\|\theta^*\|_2^2}{2\zeta^2} \right) \leq \frac{1}{16}. \quad (14)$$

Suppose that (14) holds for  $t - 1$ , then we consider the  $t$ -th iteration. We partition the  $n_0$  data points used in the  $t$ -th iteration into three parts:  $N_{t,0}$  data points which are inliers and have correct estimated labels (i.e.,  $\hat{\nu}_i^{(t)} = \nu_i$ );  $N_{t,1}$  data points which are inliers and the estimated labels are wrong;  $N_{t,2}$  data points which are outliers.

According to our contamination model and Hoeffding's inequality, we have

$$\mathbb{P}\{N_{t,2} \geq 2\alpha n_0\} \leq \exp(-2\alpha^2 n_0). \quad (15)$$

Conditioned on all the previous iterations and the fact that there are  $N_{t,2}$  outliers, we can use Hoeffding's inequality to bound  $N_{t,1}$ :

$$\mathbb{P}\{N_{t,1} \geq (A_t + \alpha)(n_0 - N_{t,2}) \mid N_{t,2}\} \leq \exp(-2\alpha^2(n_0 - N_{t,2})). \quad (16)$$

Therefore, with probability at least  $1 - 2\exp(-\alpha^2 n_0)$ , we have

$$N_{t,0} \geq (1 - (A_t + 3\alpha))n_0. \quad (17)$$

This implies that if  $n_0 \geq \frac{1}{\alpha^2} \log(2/\delta)$ , then with probability  $1 - \delta$ , (17) holds.

The next step in the algorithm is to use the iterative filtering algorithm subroutine to conduct a robust mean estimation of  $\theta^*$ . To this end, we first construct  $n_0$  inliers: for every  $i = (t-1)n_0 + 1, (t-1)n_0 + 2, \dots, tn_0$ , if  $y_i$  is an inlier, we let  $\tilde{y}_i := \nu_i y_i$ ; if  $y_i$  is an outlier, we draw  $\tau_i$  from  $\mathcal{D}_\tau$

independently from all other data points, and let  $\tilde{y}_i := \theta^* + \tau_i$ . Here we note that all the new data points  $\tilde{y}_i$  (with  $y_i$  being outliers) are *virtual*, i.e., they are only used for the analysis purpose.

Now we have  $n_0$  virtual data points  $\tilde{y}_i$ ,  $i = (t-1)n_0 + 1, (t-1)n_0 + 2, \dots, tn_0$  drawn from the inlier distribution with mean  $\theta^*$ . In the algorithm implementation, we have  $\hat{\nu}_i^{(t)} y_i$ ,  $i = (t-1)n_0 + 1, (t-1)n_0 + 2, \dots, tn_0$ . In fact, if  $y_i$  is an inlier and has correct label, we have  $\tilde{y}_i = \hat{\nu}_i^{(t)} y_i$ . Thus, the set of data points used in the algorithm  $\{\hat{\nu}_i^{(t)} y_i\}$  can be considered as a corrupted sample from  $\{\tilde{y}_i\}$ . Conditioned on the event in (17), we know that with probability at least  $1 - 2\exp(-\alpha^2 n_0)$ , the fraction of corrupted data (including outliers and inliers with wrong labels) is at most  $A_t + 3\alpha$ . According to [41, 42], the following lemma holds deterministically.

**Lemma 3.** [41, 42] Suppose that  $A_t + 3\alpha \leq \frac{1}{4}$ . Let  $\bar{y} = \frac{1}{n_0} \sum_{i=(t-1)n_0+1}^{tn_0} \tilde{y}_i$  and suppose that

$$\left\| \frac{1}{n_0} \sum_{i=(t-1)n_0+1}^{tn_0} (\tilde{y}_i - \bar{y})(\tilde{y}_i - \bar{y})^\top \right\|_2 \leq \hat{\sigma}^2.$$

Let  $\hat{\theta}^{(t)}$  be the output of the iterative filtering algorithm. Then, there exists an absolute constant  $c$  such that  $\|\hat{\theta}^{(t)} - \bar{y}\|_2 \leq c_0 \hat{\sigma} \sqrt{A_t + 3\alpha}$ .

Using similar derivations as in [8], by choosing proper value of  $\hat{\sigma}$  as a parameter in the iterative filtering algorithm, we know that there exists an absolute constant  $c_1$  such that with probability  $1 - \delta$ ,

$$\|\hat{\theta}^{(t)} - \theta^*\|_2 \leq c_1 \left( (\sigma + \zeta) \sqrt{A_t + 3\alpha} + \zeta \sqrt{\frac{d + \log(2/\delta)}{n_0}} \right) \quad (18)$$

Suppose that  $n_0 \geq \frac{1}{\alpha}(d + \log(2/\delta))$ . Then we have

$$\|\hat{\theta}^{(t)} - \theta^*\|_2 \leq c_2(\sigma + \zeta) \sqrt{A_t + 3\alpha}. \quad (19)$$

Then we proceed to analyze  $A_{t+1}$ . According to (11), we have

$$A_{t+1} = \mathbb{P}_{\tau \sim \mathcal{D}_\tau} \left\{ \langle \theta^* + \tau, \hat{\theta}^{(t)} \rangle \leq 0 \right\}$$

Since  $\tau$  is sub-Gaussian, similar to the derivation of  $A_1$ , we have

$$A_{t+1} \leq \exp \left( -\frac{\|\theta^*\|_2^2 - \|\theta^* - \hat{\theta}^{(t)}\|_2^2}{2\zeta^2} \right) \leq \exp \left( -\frac{\|\theta^*\|_2^2}{2\zeta^2} \right) \exp \left( \frac{c_2^2(\sigma + \zeta)^2(A_t + 3\alpha)}{2\zeta^2} \right).$$

Since we assume  $\frac{\sigma}{\zeta} \leq \Theta(1)$ , we have

$$\begin{aligned} A_{t+1} &\leq \exp \left( -\frac{\|\theta^*\|_2^2}{2\zeta^2} \right) \exp(c_3(A_t + 3\alpha)) \\ &\leq \exp \left( -\frac{\|\theta^*\|_2^2}{2\zeta^2} \right) (1 + e^{c_3(A_t + 3\alpha)}) \\ &\leq \exp \left( c_3 - \frac{\|\theta^*\|_2^2}{2\zeta^2} \right) (A_t + 3\alpha) + \exp \left( -\frac{\|\theta^*\|_2^2}{2\zeta^2} \right). \end{aligned}$$

Thus, as long as  $\frac{\|\theta^*\|_2^2}{2\zeta^2}$  is greater than or equal to a constant that is large enough, and we can guarantee that  $\exp(c_3 - \frac{\|\theta^*\|_2^2}{2\zeta^2}) \leq \frac{1}{8}$  and  $\exp(-\frac{\|\theta^*\|_2^2}{2\zeta^2}) \leq \frac{1}{32}$ , we have

$$A_{t+1} \leq \frac{1}{8}(A_t + 3\alpha) + \exp \left( -\frac{\|\theta^*\|_2^2}{2\zeta^2} \right) \leq \frac{1}{16}.$$

Combining the probabilistic arguments (17) and (18) and by union bound, we know that conditioned on the first  $t-1$  iterations, and the fact that (14) holds for  $t-1$ , and  $n_0 \geq \frac{1}{\alpha}(d + \frac{1}{\alpha} \log(2/\delta))$ , with probability at least  $1 - 2\delta$  over the data points in the  $t$ -th iteration, we have (14) holds for  $t$ .

Thus, we have proved (14). The final results can be obtained by iterating (14).

## B Guarantees of stage I and III of the modular algorithm

### B.1 Guarantees for stage-I:

#### B.1.1 ERM computation

We now show closeness guarantees of the ERMs,  $\hat{w}^{(i)}$  to its true risk minimizer,  $w_k^*$  for some  $k \in [K]$ . The closeness results will be required for the provable guarantees of the threshold based clustering algorithm described in Section ??.

Suppose machine  $i \in \mathcal{C}_k$ . The goal is to prove an upper bound on  $\|w^{(i)} - w_k^*\|$ . We now present the result for completeness.

**Theorem 6.** [46] *Under Assumptions 1, 2 and 3, with probability at least  $1 - \delta$ , we have*

$$F_k(\hat{w}^{(i)}) - F(w_k^*) \leq \mathcal{O}\left(\frac{G_1^2 L_1 \log(1/\delta)}{\lambda^2 n}\right)$$

Using strong convexity (Assumption 2), with probability exceeding  $1 - \delta$ , we obtain

$$\|\hat{w}^{(i)} - w_k^*\|^2 \leq \mathcal{O}\left(\frac{G_1^2 L_1 \log(1/\delta)}{\lambda^3 n}\right).$$

#### B.1.2 Online-to-batch conversion

Here the  $i$ -th compute node runs an *online-to-batch conversion* routine to obtain  $\bar{w}^{(i)}$ . Suppose the loss function  $f(w, \cdot)$  is convex and  $G_1$  Lipschitz with respect to  $w$ , and  $w \in \mathcal{W}$  is bounded by  $D_1$ . [46] along with Assumption 2 shows that, with probability greater than or equal to  $1 - \delta$ , we have

$$\bar{w}^{(i)} - w_k^* \leq \mathcal{O}\left(\frac{D_1^2 G_1^2 \log(1/\delta)}{\lambda n}\right).$$

From the above results, computing the ERM directly and performing an online optimization over  $n$  episodes are order-wise identical. All the closeness results for ERMs holds only for non-Byzantine machines.

### B.2 Stage III-robust distributed optimization

Note that, since we have  $\alpha m$  Byzantine machines and since we cannot control the clustering of Byzantine machines, the clustering results of this section are not perfect. We let the third phase of the algorithm, i.e., the robust distributed optimization to take care of this.

In the third stage of our algorithm, we implement robust distributed optimization algorithms to learn models for every cluster. After the robust clustering step, we obtain the clustering results  $\hat{\mathcal{C}}_1, \hat{\mathcal{C}}_2, \dots, \hat{\mathcal{C}}_K$ . In the adversarial setting, the clustering result is not guaranteed to be perfect. Without loss of generality, we assume that more than half of the worker machines in  $\hat{\mathcal{C}}_k$  are from true cluster  $\mathcal{C}_k$  for every  $k \in [K]$ . We denote the fraction of worker machines in  $\hat{\mathcal{C}}_k$  that do not belong to  $\mathcal{C}_k$  by  $\hat{\alpha}_k$  and  $\max_{k \in [K]} \hat{\alpha}_k < 0.5$ . The goal of this stage of the algorithm is to learn a model for every cluster, by jointly using all the machines in  $\hat{\mathcal{C}}_k$ .

To do this, we use the recently developed Byzantine-robust distributed learning algorithms. These algorithms usually take the following steps: in every iteration, the master machine sends the model parameter to the worker machines; the worker machines compute the gradient of their loss functions with respect to the model parameter; the master machine conducts a robust estimation of the gradients collected from all the worker machines and run a gradient descent step. Here, the robust estimation of gradients is a subroutine of the algorithm, and there are a few choices that we can consider. Some examples are median, trimmed mean, and high dimensional robust estimation algorithms such as iterative filtering. The statistical error rates of robust distributed gradient descent with median and trimmed mean subroutine have been analyzed by [7], and the error rates of the robust distributed gradient descent with iterative filtering has been analyzed by [8]. Without loss of generality, we analyze the  $k$ th cluster, where  $k \in [K]$  and assume  $M := |\hat{\mathcal{C}}_k|$ .

We will now state the convergence result of trimmed mean and iterative filtering based robust distributed algorithm. Note that, the number of Byzantine nodes in cluster  $\mathcal{C}_k$  is at most  $\alpha m$ , since in the worst case all the Byzantine nodes will be wrongly clustered together in  $\mathcal{C}_k$ . Thus, the fraction of Byzantine nodes in this cluster will be at most  $\hat{\alpha}_k = \frac{\alpha m}{M}$ . Let  $w^T$  and  $w^0$  are the  $T$ -th



iterate and the initial value of the optimization algorithm respectively and  $w^*$  is the global minima. We have the following guarantee.

**Theorem 7.** [7, 8] Suppose that Assumptions 4, 2 and 3 hold, and  $\hat{\alpha}_k \leq \frac{1}{2} - \epsilon$  for some  $\epsilon > 0$ . With constant step-size of  $1/L_1$  and with probability at least  $1 - \mathcal{O}(\frac{d}{(1+nM)^d})$ , after  $T$  iterations, we have  $\|w^T - w^*\| \leq (1 - \frac{\lambda_F}{L_1 + \lambda_F})^T \|w^0 - w^*\| + \frac{2}{\lambda_F} \Delta'$ , where,  $\Delta' := \tilde{\mathcal{O}}(\frac{\hat{\alpha}_k d}{\sqrt{n}} + \frac{d}{\sqrt{nM}})$  for trimmed mean and  $\Delta' := \tilde{\mathcal{O}}(\frac{\sqrt{\hat{\alpha}_k}}{\sqrt{n}} + \frac{\sqrt{d}}{\sqrt{nM}})$  for iterative filtering.

**Remark 6.** We can relax Assumption 2 and obtain

$$F_i(w^T) - F_i(w^*) \leq \tilde{\mathcal{O}}(\frac{\hat{\alpha}_k d}{\sqrt{n}} + \frac{d}{\sqrt{nM}})$$

with high probability for the trimmed mean algorithm. Similarly for the iterative filtering, we have

$$F_i(w^T) - F_i(w^*) \leq \tilde{\mathcal{O}}(\frac{\sqrt{\hat{\alpha}_k}}{\sqrt{n}} + \frac{\sqrt{d}}{\sqrt{nM}})$$

with high probability.

**Remark 7.** By running  $T \geq \frac{L_1 + \lambda_F}{\lambda_F} \log(\frac{\lambda_F}{2\Delta'} \|w^0 - w^*\|)$  parallel iterations, we can obtain a solution  $w^T$  satisfying  $\|w^T - w^*\|_2 \leq \mathcal{O}(\Delta')$ . Also, as shown by [8] the term  $\frac{\sqrt{d}}{\sqrt{nM}}$  is unavoidable, and hence the error rate for iterative trimmed means is order optimal in dimension.

## C Proof of Theorem 2

The proof of the theorem comes directly via combining Theorem 4 and 7. Let  $\tilde{\alpha}_i = \left(\frac{\rho M_i + \alpha m}{M_i + \alpha m}\right)$ . Note that, since  $G_S$  denotes fraction of non-Byzantine machines that are mis-clustered,  $\tilde{\alpha}_i$  denotes the worst case fraction of Byzantine machines for  $i$ -th cluster. Assuming  $\max_{i \in [K]} \tilde{\alpha}_i < \frac{1}{2}$  and invoking Theorem 7 yields the result.

## D Technical lemmas

We now list a few technical lemmas. These lemmas (along with the proofs) appear in the Appendix of [45], and typically follow from the concentration phenomenon of sub-gaussian random variables. For the sake of completeness of the arguments made in the proofs of Theorem 3 and 4, we are re-writing the results here without proofs.

In the setup of Section 5.1, suppose we have  $\tau_1, \dots, \tau_t$  such that  $\tau_i \sim \mathcal{N}(0, \sigma^2 I_d)$  for all  $i \in [t]$  and  $\tau_i$ 's are independent.

**Lemma 4.** Let  $S \subset [t]$  with  $\tau_S = \sum_{i \in S} \tau_i$ . We have,

$$\|\tau_S\| \leq \sigma \sqrt{2(n + 9d)|S|}$$

with probability at least  $1 - \exp(-0.1t)$ .

**Lemma 5.** Let  $\bar{\tau} = \frac{1}{t} \sum_{i=1}^t \tau_i$ . We have,  $\langle \bar{\tau}, \theta^* \rangle \geq -\frac{\|\theta^*\|^2}{\sqrt{t}}$  and  $\|\bar{\tau}\|^2 \leq \frac{3d\sigma^2}{t} + \frac{\|\theta^*\|^2}{t}$  with probability at least  $1 - 2\exp(-\frac{\|\theta^*\|^2}{3\sigma^2})$

**Lemma 6.** Consider the matrix  $\sum_{i=1}^t \tau_i \tau_i^T$ . The maximum eigenvalue of the matrix is upper bounded by  $1.62(n + 4d)\sigma^2$  with probability greater than  $1 - \exp(-0.1t)$ .

We now assume that  $\tau_i$  are independent sub-gaussian with zero mean and with parameter  $\sigma^2$ . We have the following results:

**Lemma 7.** For  $S \subset [t]$ ,  $\tau_S = \sum_{i \in S} \tau_i$  satisfy,

$$\|\tau_S\| \leq \sqrt{\sigma(3(t + d)|S|)}$$

with probability at least  $1 - \exp(-0.3t)$ .

**Lemma 8.**  $\lambda_{max}\left(\sum_{i=1}^t \tau_i \tau_i^T\right) \leq 6\sigma^2(t + d)$   
with probability greater than  $1 - \exp(-0.5t)$ .

We now follow the notations of Section 5.2 for the remaining couple of results.

**Lemma 9.** Let  $\tau_{T_h^*} = \sum_{i \in T_h^*} \tau_i$ . Then,  $\|\tau_{T_h^*}\| \leq 3\sigma\sqrt{(d + \log m)|T_h^*|}$  for all  $h \in [K]$  with probability at least  $1 - \frac{1}{m^3}$ .

**Lemma 10.** For fixed  $x_1, \dots, x_k \in \mathbb{R}^d$  and  $b > 0$ ,

$$\sum_{i \in T_g^*} 1\{b\|x_h - x_g\|^2 \leq \langle \tau_i, \|x_h - x_g\| \rangle\} \leq n_g^* \exp\left(-\frac{b^2 \Delta^2}{2\sigma^2}\right) + \sqrt{5n_g^* \log m}$$

for all  $g \neq h$  with probability at least  $1 - \frac{1}{m^3}$ .