Approximation Guarantees for Robust Aggregation in Federated Learning

Mélanie Cambus
Aalto University
Espoo, Finland
melanie.cambus@aalto.fi

Darya Melnyk
TU Berlin
Berlin, Germany
melnyk@tu-berlin.de

Tijana Milentijević
TU Berlin
Berlin, Germany
tijana.milentijevic@tu-berlin.de

Stefan Schmid
TU Berlin & Weizenbaum Institute
Berlin, Germany
stefan.schmid@tu-berlin.de

Abstract

Byzantine-tolerant aggregation is a central challenge in federated learning, especially under heterogeneous client data where malicious updates may be statistically indistinguishable from honest but unusual clients. Robustness of aggregation algorithms is crucial to the quality of the final model. In this work, we study the robustness of the Byzantine-tolerant aggregation through the lens of centroid approximation, examining how closely an aggregation rule can match the average of honest client updates when up to t of n clients are Byzantine. We connect this objective to classical validity conditions from Byzantine agreement and show that validity alone is insufficient to guarantee good centroid approximation. Our main theoretical contribution is the first lower bound of $\sqrt{\min\{(n-t)/t, d\}}/2$ on the centroid approximation for aggregation under box validity and a matching upper bound of $2\sqrt{\min\{n, d\}}$ in the practically relevant case $n > 2t$. In addition, we present a new algorithm that achieves a $2d$-approximation under convex validity, which also proves that the existing lower bound in the literature is tight. These results expose a fundamental tradeoff between validity conditions and approximation quality. We complement the theory with experiments in FedSGD and FedAvg under standard Byzantine attacks, showing that stronger validity improves stability, while tighter centroid approximation can improve accuracy in heterogeneous settings.

1 Introduction

Federated learning (FL) [1, 2] is a decentralized technique for training machine learning models based on sharing model parameters while keeping the training data *locally*. In this work, we focus on the setting where the clients share updates — either the gradients in case of *federated stochastic gradient descent (FedSGD)* or the model parameters in case of *federated averaging (FedAvg)* — with a trusted central server. After the server has received the updates, it aggregates the results, updates the model parameters, and then shares the new model parameters with the clients for the next training round. This technique is popular when data privacy requirements prevent clients from sharing their data directly with the server [3, 4, 5].

The aggregation step performed by the server is particularly interesting since it can have a large influence on the quality of the final ML model. The most common aggregation rule used to select a representative vector (gradient or model parameters) is averaging [1, 2, 6, 7, 8, 9, 10, 11, 12, 13]. However, when averaging is used, training can fail if some clients do not behave as expected. In

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

particular, a single faulty vector can arbitrarily shift the average in any direction, leading to erroneous updates of the model parameters. Especially in the context of federated learning, it is crucial to be robust to malicious behavior and Byzantine faults, which is also the focus of our paper. In the case of homogeneous training data, it is usually possible to use similarities between vectors to exclude such outliers [14, 15, 16]. If the data is heterogeneous, such similarities may not exist.

Lack of Byzantine-tolerant robustness measures. Previously proposed Byzantine-tolerant FL methods for heterogeneous datasets focus on showing convergence of the training process and apply statistical methods for vector aggregation [17, 18, 19]. To mitigate Byzantine behavior, their methods remove outliers from the data and make additional assumptions on the input vectors of the clients. However, **outliers in heterogeneous settings are not necessarily bad and removing them may reduce the quality of the trained model**. An alternative approach is to use the absolute or average distance to the average to evaluate federated learning algorithms [20]. However, **such absolute measures only allow one to analyze Byzantine attacks under worst-case input distributions**. A popular robustness measure that incorporates different input distributions and Byzantine vectors is (f, κ)-robustness [21]: Assuming $f < n/2$ Byzantine nodes and $\kappa \geq 0$, an aggregation rule F is said to be (f, κ)-robust if for any vectors $x_1, \ldots, x_n \in \mathbb{R}^d$, and any set $S \subseteq [n]$ of size $n - f$,

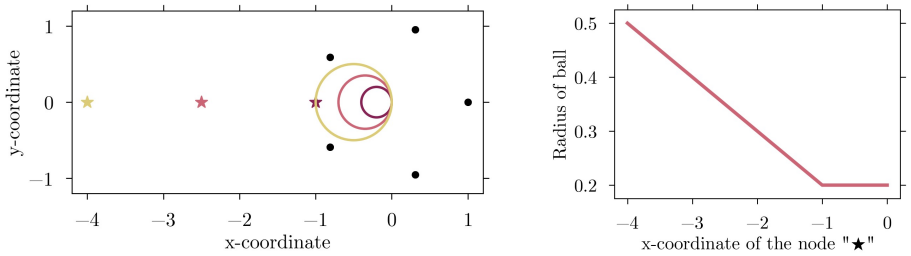
$$\|F(x_1, \ldots, x_n) - \bar{x}_S\|^2 \leq \frac{\kappa}{|S|} \sum_{i \in S} \|x_i - \bar{x}_S\|^2$$

where $\bar{x}_S = \frac{1}{|S|} \sum_{i \in S} x_i$. The recent work [22, 23] improves this measure by introducing a robustness factor $\kappa^\star$ that maximizes the ratio between $\|F(x_1, \ldots, x_n) - \bar{x}_S\|^2$ and $\frac{1}{|S|} \sum_{i \in S} \|x_i - \bar{x}_S\|^2$ from the above term. Observe that for the special case where S consists only of non-faulty nodes, both robustness notions provide a desired property for any Byzantine-tolerant vector aggregation algorithm: the distance between the aggregated vector and the average of vectors in S should depend on the distribution of the nodes in S. However, since the robustness considers every subset S of $n - f$ nodes, certain **Byzantine attacks can lead to bad (f, κ)-robustness guarantees, even for an optimal algorithm**. In the following, we present a possible attack:

Example 1.1 (Unbounded κ for an optimal aggregation algorithm). Consider a vector aggregation algorithm that outputs the centroid of non-faulty vectors, denoted $\mathrm{Cent}^\star$. This algorithm is optimal. For simplicity, assume a setting where all input vectors lie on a line. Let the $n - 2f$ non-faulty vectors have the input value 0, the f non-faulty nodes have the input value $a > \frac{(n-2f)(n-f)}{f}$, and the f Byzantine nodes have the input value $-\varepsilon$, where $\varepsilon > 0$ is an arbitrarily small constant. Observe that in the definition of (f, κ)-robustness, one subset will contain all Byzantine vectors and the $n - 2f$ non-faulty nodes with input value 0. We denote this subset S_{Byz}. Also observe that the average of the non-faulty nodes is located at $\mathrm{Cent}^\star = \frac{af}{n-f}$, the average of the nodes in S_{Byz} is located at $\bar{x}_{S_{Byz}} = -\frac{\varepsilon f}{n-f}$, and the squared average distance from the nodes in S_{Byz} to $\bar{x}_{S_{Byz}}$ is $\frac{\varepsilon^2 f(n-2f)}{n-f}$. Therefore, κ has to be chosen such that $\left(\frac{af}{n-f} + \frac{\varepsilon f}{n-f}\right)^2 \leq \kappa \cdot \frac{\varepsilon^2 f(n-2f)}{n-f}$ since $F(x_1, \ldots, x_n) = \mathrm{Cent}^\star$. We thus have the result that $\kappa \geq \frac{fa^2}{\varepsilon^2(n-2f)(n-f)} > \frac{1}{\varepsilon^2}$. This shows that the constant κ can however be arbitrarily large for an optimal algorithm. This example also extends to the newer robustness factor $\kappa^\star$.

This example suggests that the (f, κ)-robustness provides a reasonable robustness measure when the failing nodes are outliers. However, as the above example shows, the measure is not guaranteed to be small when the failing nodes show arbitrary, i.e., Byzantine, behavior. Note that no distributed algorithm can compute $\mathrm{Cent}^\star$ deterministically under the assumption in [21] that Byzantine faults are undetectable. In Appendix A.1, we therefore provide an additional example, where the distributed algorithm computes the centroids of all subsets of $n - f$ vectors, stores them in a candidate set, and chooses the midpoint of the minimum covering ball (see Definition 2.3) of the candidate set. In [24], the authors argue that this algorithm computes an optimal output under the assumption that the Byzantine nodes are undetectable, since the midpoint minimizes the maximum distance to any point in the candidate set. We show that also for this algorithm κ can be unbounded, using a construction similar to the above. We therefore argue that the (f, κ)-robustness definition is not suited to evaluate the quality of a Byzantine-tolerant algorithm.

In search of a Byzantine-tolerant robustness condition, we focus on a new approximation measure introduced in distributed computing for approximate agreement algorithms [24]. This measure



(a) Distribution of the input vectors

(b) Radius of the ball of averages

Figure 1: This figure shows how the radius of the smallest ball containing all averages depends on different input distributions. There are 6 clients, one of which is possibly Byzantine. On the left, the points represent fixed input vectors. The three stars represent three different scenarios of the input of the sixth client. The circles represent the smallest balls containing all possible averages on subsets of five points. On the right, the radius of the minimum covering ball is presented when the x-coordinate of the last client is moved from -4 to 0. Observe that the radius of the minimum covering ball cannot be zero, as any of the points in the figure are also potentially Byzantine. The yellow scenario is a "bad" input setting where there is either one non-faulty client with very distinct data, or a Byzantine party tries to disrupt the training process. The dark red scenario is a benign setting where an aggregation algorithm should be able to output a vector close to the actual centroid. Accordingly, the radius of the ball is small in this scenario.

estimates the quality of an aggregated average in a Byzantine environment and allows one to not only analyze the worst-case input setting, but also estimate the quality of an algorithm based on the given input distribution.

In this work, we transfer the idea of approximating the average vector to the traditional federated learning setting with n clients and one trusted server. In distributed computing, validity conditions are used to restrict an algorithm from terminating on arbitrary inputs. We investigate the trade-off between the validity conditions and the approximation of the average vector for federated learning. This allows us to present aggregation algorithms that perform well under different input distributions.

The benefits of average approximation. We consider the approximation of the average to evaluate the quality of our algorithms. As we motivate in the following, a low average approximation ratio implies that an algorithm performs well for a given input distribution. Formally, given n vectors, up to t of which can be Byzantine, an optimal choice of the average vector under Byzantine attacks is defined as the midpoint of the smallest ball B that encloses each average obtained from every subset of $n - t$ vectors. When t clients are Byzantine, exactly one of these averages was computed from only non-faulty vectors. Therefore, the midpoint minimizes the maximum distance to the non-faulty average vector in the worst case. The approximation ratio is then defined as the ratio between the distance from the aggregation vector to the non-faulty average and the radius of B.

The main advantage of this approximation ratio is that it is defined relative to the input setting: Under heterogeneous data, Byzantine vectors cannot be differentiated from non-faulty vectors. That is, a large radius of the minimum covering ball either represents "bad" Byzantine behavior, or a "bad" initial configuration where each client has vastly different input. In such a scenario, no aggregation algorithm can choose a representative average vector. The large ball radius prevents one from punishing an algorithm for a large absolute distance to the average vector. A small radius, however, represents "benign" Byzantine behavior and very similar inputs. In such a scenario, an aggregation algorithm should be able to choose an aggregation vector that is close to the original average. Figure 1 visualizes the continuous change in the ball radius depending on the input vectors of the clients.

Contributions. We first show that known validity conditions from the literature do not guarantee good approximation of the average. We then show that under weak and strong validity conditions, both of which only require the server to output the same vector as the non-faulty client if all non-faulty clients send the server the same vector, a constant approximation of the average can be achieved.

Our first main contribution is almost tight bounds for algorithms that satisfy box validity, where the aggregation vector lies in the coordinate-parallel hyperbox of non-faulty vectors. We present a new

lower bound of $\sqrt{\min\{(n-t)/t, d\}}/2$ for the centroid approximation and show that the existing Box algorithm can achieve an approximation of $2\sqrt{\min\{n, d\}}$ by providing a new upper bound proof for the case $n < d$. Our second main contribution is a tight upper bound (a $2d$-approximation) for convex validity, where the aggregation vector lies in the convex hull of all vectors. Convex validity is the predominant validity condition used to solve multidimensional Byzantine agreement [25, 26]. Note that this setting is only of theoretical interest to this work, as it requires $n > (d+1) \cdot t$, where d is the dimension. In fact, [27] show that box validity is the only k-relaxed convex hull, with $k = 1$, that allows reducing the number of nodes from $n > (d+1) \cdot t$ to $n > 3t$. We show that all presented bounds can also be achieved in the distributed peer-to-peer setting. The agreement algorithms presented differ from [20, 24], since only exact agreement is considered in this paper.

Finally, we extend our analytical results with simulations. Since the dimension of the data is much larger than the number of clients, we focus on algorithms that satisfy box validity and weaker validity conditions. In our evaluation, we differentiate between the settings where the gradients (*FedSGD*) and the model parameters (*FedAvg*) are aggregated, and show how selected algorithms perform under different failure scenarios.

1.1 Related Work

Dean et al. [28] proposed a first distributed solution to train a large machine learning model on tens of thousands of CPU cores. Their work initiated a study of asynchronous algorithms for distributed stochastic gradient descent (SGD) that focus on scalability and communication efficiency [29, 30, 31, 32]. The synchronous version of SGD has been proposed by Chen et al. [33]. We refer to this framework as *FedSGD*. *FedSGD* has also been considered under Byzantine adversaries, both in synchronous [34, 35] and asynchronous settings [36]. While the mentioned work assumes homogeneous data distributions, some efforts have also been made to incorporate data heterogeneity [18, 37, 19, 17]. To tackle Byzantine behavior of the clients, these approaches make use of homogeneity of the data, apply robust statistical methods, or try to detect and remove Byzantine behavior.

Federated averaging was introduced in [1, 2] to perform training where the data is private, unbalanced, non-IID, and distributed across mobile devices. Here, model parameters instead of gradients are exchanged with a server. We refer to this framework as *FedAvg* in this paper. Much of the follow-up work has focused on showing convergence of the models in this framework without failures [9, 38, 10, 12, 13, 39]. Byzantine-tolerant approaches have also been introduced for this setting, where the goal is to remove Byzantine behavior via stochastic quantization and outlier detection mechanisms [40].

In contrast to previous work (e.g., robust statistics), we do not make any assumptions on the data distribution and do not focus on removing Byzantine clients from the training process, as such a process may influence the accuracy when the data is heterogeneous and no malicious behavior is present in the system. Instead, we use the approximation definition for the average from [24] that naturally incorporates Byzantine clients. Moreover, other than in [24], we consider a stronger model without agreement, which makes our lower bound results more powerful, and introduce new algorithms that achieve an optimal approximation.

2 Model and Definitions

We consider a client/server setting with one server and n clients. The goal is to train a global neural network on the server with data spread heterogeneously among clients. In order to train the global model without gathering data from clients, each client possesses its own copy of the model and then shares only vectors generated from their local data and model with the server. The server then needs to aggregate the received vectors to advance the training of the global model. The training process is performed in synchronous rounds.

On top of the training set-up, we consider that up to $t < n/2$ of the clients can be Byzantine, i.e., they can behave arbitrarily, can collude, and are not bound to following the protocol. The aggregation algorithms used by the server hence need to account for this. Note that we use the standard assumption from distributed computing that Byzantine clients are not differentiable from non-faulty clients as long as they follow the protocol and only lie about their input. We treat all clients equally and do not

weigh their inputs based on the size of their local dataset. This is because we derive bounds based on the number of Byzantine clients as opposed to bounds based on the size of the clients' local datasets.

The focus of this work is on the aggregation function. Consider a specific communication round, in which each client sends a vector to the server, and the server aggregates those vectors. To account for the potential presence of Byzantine clients in the system, the aggregation algorithm used by the server needs to compute an aggregation that is as little influenced by Byzantine vectors as possible. In this work, we focus on the most common aggregation rule in federated learning – the averaging aggregation rule. Since Byzantine clients can be present in the system and are undetectable, an aggregation algorithm cannot determine the centroid of vectors of non-faulty clients. We are therefore interested in the quality of the computed aggregated vector.

Centroid approximation. Let $\{v_i, i \in [n]\}$ be the set of all input vectors (gradients or model parameters) that each client starts with at the beginning of an aggregation step. We refer to this set as the input layout. Note that up to $t < n/2$ of those vectors could be faulty or not sent because of Byzantine behavior. We assume that the server receives up to m vectors $\{v_i, i \in [m]\}$, where $n-t \leq m \leq n$. Each vector v is in the normed vector space $\left(\mathbb{R}^d, \|\cdot\|_2\right)$, where $\forall x = (x_1, \ldots, x_d) \in \mathbb{R}^d, \|x\|_2 = \sqrt{\sum_{k=1}^{d} x_k^2}$ and the distance between any two vectors v and w is their Euclidean distance $\mathrm{dist}(v, w) = \|v - w\|_2$. When not specified, $\|\cdot\|$ refers to the 2-norm. We use the following definition of the average/centroid:

Definition 2.1 (Centroid)**.** The *centroid* of a finite set of k vectors $\{v_i, i \in [k]\}$ is $\frac{1}{k}\sum_{i=1}^{k} v_i$.

We define the centroid approximation as in [24]. Let $\mathrm{Cent}^\star$ be the centroid computed from non-faulty vectors only. Note that there can be up to n non-faulty vectors as t is only an upper bound on the number of Byzantine clients. In the following, we define the set of candidate centroids, which are computed based on the worst case where exactly t vectors are Byzantine.

Definition 2.2 (Set of candidate centroids)**.** Let $L = \{v_i, i \in [n]\}$ be the input layout. The *set of candidate centroids*, denoted $\mathrm{S_{Cent}}$, is defined as

$$\mathrm{S_{Cent}} \coloneqq \left\{ \frac{1}{n-t} \sum\nolimits_{i \in I} v_i \middle| \forall I \subseteq [n] \; s.t. \; |I| = n - t \right\}.$$

Observe that $\mathrm{S_{Cent}}$ cannot always be computed by an algorithm, if the algorithm does not receive all the input vectors. Instead, the algorithms compute a subset of $\mathrm{S_{Cent}}$. Since Byzantine clients are not differentiable from non-faulty clients as long as they follow the protocol, we can only define the centroid approximation based on the worst case where exactly t clients are Byzantine. We define the point minimizing the maximum distance to all vectors in the set of candidate centroids defined above as the center of the following ball:

Definition 2.3 (Minimum covering ball [41])**.** The minimum covering ball $\mathrm{Ball_{cov}}(\mathrm{S_{Cent}})$ is the smallest ball containing all vectors in $\mathrm{S_{Cent}}$. Its radius is denoted $\mathrm{Rad_{cov}}$.

Finally, the centroid approximation is defined as follows:

Definition 2.4 (Centroid approximation)**.** Let $f \leq t$ the actual number of Byzantine faults. Given an input layout $L = \{v_i, i \in [n]\}$, let $O_{\mathcal{A}}$ be the output of an algorithm $\mathcal{A}$ computing an approximation of the centroid of the $n - f$ non-faulty vectors. The approximation ratio of $\mathcal{A}$ given L is the smallest ρ s.t. $\mathrm{dist}(O_{\mathcal{A}}, \mathrm{Cent}^\star) \leq \rho \cdot \mathrm{Rad_{cov}}$. The algorithm $\mathcal{A}$ is said to compute an *ρ-approximation of the centroid* if, for all input layouts L, the approximation ratio of $\mathcal{A}$ given L is upper bounded by ρ.

To satisfy box validity, we will define algorithms that rely on a less restrictive area than $\mathrm{Ball_{cov}}(\mathrm{S_{Cent}})$:

Definition 2.5 (Centroid hyperbox)**.** The *centroid hyperbox* CH is the smallest coordinate-parallel hyperbox containing $\mathrm{S_{Cent}}$.

Validity conditions. We noted above that a Byzantine client can shift the centroid of vectors of all clients arbitrarily, and thus it can also shift the midpoint of the minimum covering ball arbitrarily far away from $\mathrm{Cent}^\star$. Just choosing the center as the centroid approximation might not be sufficient to ensure that we can trust the output of a certain algorithm. We hence take inspiration from the distributed agreement algorithms and use validity conditions to get additional guarantees on the

output of different algorithms, complementing the guarantees given by the centroid approximation ratio. A validity condition is satisfied when the output of an algorithm is guaranteed to be in a specific area, depending only on the input layout. In this work, we focus on common validity conditions from the literature. The following notation is used for one of the validity definitions:

Notation 1. The smallest coordinate-parallel hyperbox containing only non-faulty vectors is called the trusted hyperbox and denoted TH.

Definition 2.6 (Validity conditions)**.** Let n denote the number of clients, up to t of which can be Byzantine. An algorithm $\mathcal{A}$ satisfies

weak validity [42, 43, 44] if, when all clients are non-faulty and all input vectors v_i are equal to a single vector v, the output of $\mathcal{A}$ is v;

strong validity [45, 46, 47] if, when all non-faulty input vectors v_i are equal to a single vector v, the output of $\mathcal{A}$ is v;

box validity [24, 48, 49] if the output of $\mathcal{A}$ is inside TH (see Notation 1);

convex validity [50, 25, 51] if the output of $\mathcal{A}$ is inside the convex hull of all non-faulty input vectors.

Note that TH cannot be computed in practice. However, we prove in Section 3 (Lemma 3.2) that an algorithm satisfies the box validity condition if and only if it agrees inside a hyperbox called the trimmed trusted hyperbox (TTH):

Definition 2.7 (Trimmed trusted hyperbox)**.** Let $v_1, \ldots, v_m$ be the received input vectors, where m is the number of received messages. The number of Byzantine values for each coordinate is at most $m - (n - t)$. Denote $\phi : [m] \to [m]$ a bijection s.t. $v_{\phi(j_1)}[k] \leq v_{\phi(j_2)}[k], \forall j_1, j_2 \in [m]$. The *trimmed trusted hyperbox* is the Cartesian product of $\mathrm{TTH}[k] \coloneqq \left[v_{\phi(m-(n-t)+1)}[k], v_{\phi(n-t)}[k]\right]$ for all $k \in [d]$.

In a similar manner, it is proved in [24] that, in order to satisfy the convex validity condition, an algorithm must agree inside the following area:

Definition 2.8 (Safe area [25])**.** Consider m vectors $\{v_1, \ldots, v_m\} =: V$, $n - t \leq m \leq n$, $t < n/(\max\{3, d+1\})$ of which can be Byzantine. Let $C_1, \ldots, C_{\binom{m}{n-t}}$ be the convex hulls of every subset of V of size $n - t$. The *safe area* is the intersection of these convex hulls: $\bigcap_{i \in [\binom{m}{n-t}]} C_i$.

3 Centroid approximation in Byzantine-tolerant FL

In this section, we first consider approximation guarantees that are given by validity conditions only. We show that only the box validity condition guarantees a bounded approximation ratio of the $\mathrm{Cent}^\star$. In the second part, we consider the best possible approximation that can be achieved under various validity conditions. We provide tight approximation bounds for each validity condition, apart from the box validity condition, where a gap remains for some specific values of n and d. In Appendix A.4, we show how our results can be transferred to federated learning in a peer-to-peer network.

3.1 Approximation guarantees given by validity conditions

In Appendix A.2, we show that weak, strong, and convex validity conditions are not sufficient to guarantee that an algorithm achieves a bounded approximation ratio of $\mathrm{Cent}^\star$ (see Lemma A.2—Lemma A.4). This is because it is possible to build specific inputs for which there exists an algorithm satisfying the respective validity condition such that the output of the algorithm is at a nonzero distance from the centroid of non-faulty vectors and the minimum covering ball is reduced to a single point. Thus, satisfying the validity condition alone is not sufficient for an algorithm to be guaranteed to have a bounded approximation ratio of the centroid of non-faulty vectors. On the other hand, box validity allows one to achieve a $t/(n-t) \cdot 2 \cdot \sqrt{d}$-approximation in the worst case (Lemma A.5).

3.2 Upper and lower bounds for centroid approximation

In this section, we present upper and lower bounds for centroid approximation under different validity conditions. An overview of these results is presented in Table 1. Note that most bounds are tight. Only in the case $n < d$, there is a gap for approximation under box validity that remains to be

investigated. The bounds for weak and strong validity are presented in the appendix (Lemma A.6—Lemma A.8). In [24], a lower bound of $2d$ has been presented for convex validity for the worst case where $n = (d+1)t + 1$. In Appendix A.3, we generalize this bound to hold for any $n > (d+1)t$ (see Lemma A.11). Our algorithm outputs a vector contained in the *safe area* that minimizes the distance to the center of $\mathrm{Ball}_{\mathrm{cov}}(\mathrm{S}_{\mathrm{Cent}})$. We also present the first matching upper bound for convex validity in Lemma A.9. We next give an upper bound result for the box validity condition. Note that there are two algorithms in the literature that achieve the same approximation ratio.

Lemma 3.1 (Upper bound for box validity)**.** *One round of the Box algorithm [24] or the RB-TM algorithm [20] achieves an approximation ratio of* $2\sqrt{\min\{n, d\}}$*, where* $n > 2t$*.*

Proof. Note that both algorithms were presented to solve approximate agreement. We can however let the server run one round of these algorithms as if the server were one of the nodes in the distributed network. In [24], it was shown that the output vector of one node at the end of a round is inside the intersection of CH and TTH. Note that in their analysis the two boxes are non-empty and intersect already if $n > 2t$. This condition is sufficient to achieve a $2\sqrt{d}$-approximation [24]. This solves the case $n > d$ for $n > 2t$.

We next consider the case $n < d$. Note that if CH has dimension n, the diagonal length argument from [24] implies a $2\sqrt{n}$ bound on the approximation ratio. Suppose that CH has dimension d' where $n < d' \leq d$. Since there are n input vectors and all elements of $\mathrm{S}_{\mathrm{Cent}}$ are computed from those vectors, $\mathrm{Conv}(\mathrm{S}_{\mathrm{Cent}})$ has to be contained in a subspace U_{input} of dimension n. The hyperbox CH of dimension d' is the smallest possible hyperbox containing the convex polytope $\mathrm{Conv}(\mathrm{S}_{\mathrm{Cent}})$. Hence, $\mathrm{Conv}(\mathrm{S}_{\mathrm{Cent}})$ has to intersect all $2d'$ faces of CH, otherwise there exists a hyperbox strictly contained in CH that contains $\mathrm{Conv}(\mathrm{S}_{\mathrm{Cent}})$. For the sake of simplicity, assume that CH is the unit hypercube of dimension d' placed at the origin with non-negative coordinates only. Note that translation and rotation of all points do not influence the approximation ratio. Further, all following computations can be adjusted with the length of the longest edge of CH to achieve the same in the general case. Observe that $\mathrm{Conv}(\mathrm{S}_{\mathrm{Cent}})$ has to intersect all faces of CH that contain the origin. Consider the set of centroids in $\mathrm{S}_{\mathrm{Cent}}$ that lie on these d' faces. Any two such centroids that lie on different faces are linearly independent. Since $\mathrm{Conv}(\mathrm{S}_{\mathrm{Cent}})$ spans at most an n-dimensional subspace, at most n centroids in this set can be linearly independent. Note that the radius of $\mathrm{Ball}_{\mathrm{cov}}(\mathrm{S}_{\mathrm{Cent}})$ is maximized when the centroids lie on intersections of many faces. Consider the largest subset of linearly independent centroids that intersect the d' considered faces (this subset can be chosen greedily). On average, each centroid in this subset lies in the intersection of at least d'/n faces. Thus, at least one of these centroids must lie in the intersection of at least d'/n faces of the unit hypercube. This implies that the radius of the minimum covering ball is at least $\sqrt{d'/n}/2$ (the intersection of k faces is at distance $\sqrt{k}/2$ from the center of the hyperbox). However, since the centroid of non-faulty vectors has to be contained inside $\mathrm{Conv}(\mathrm{S}_{\mathrm{Cent}}) \subseteq$ CH, the distance between the output of an algorithm agreeing inside CH and $\mathrm{Cent}^\star$ centroid is at most $\sqrt{d'}$, hence the approximation ratio is at most $2 \cdot \sqrt{n}$. Hence, the approximation ratio of the hyperbox algorithm is at most $2 \cdot \sqrt{\min\{n, d\}}$. □

Before addressing the lower bound for algorithms satisfying box validity, we first prove:

Lemma 3.2. *An algorithm satisfying box validity has to agree inside the trimmed trusted hyperbox.*

Proof. We assume that t Byzantine parties follow the algorithm with their own (worst-case) input vectors, thus being undetectable. Let us consider a consensus algorithm such that the output vector v always satisfies box validity. For the sake of contradiction, suppose this output vector is outside of

validity	LB for $n > (d+1)t$	LB for $n < (d+1)t$	upper bound
weak	1	1	1 (Lem. A.6)
strong	2 (CM[24])	2 (CM[24])	2 (Lem. A.7)
box	$\sqrt{d}$ (Lem. 3.3)	$\sqrt{\min\{(n-t)/t, d\}}/2$ (Lem. 3.3)	$2\sqrt{\min\{n, d\}}$ (Lem. 3.1)
convex	$2d$ (CM[24], Lem. A.11)	not possible (MHVG[25])	$2d$ (Lem. A.9)

Table 1: This is an overview of the results established in this section. Already known results are cited in the respective cells. The lower bound on weak validity follows from the definition of approximation.

TTH. By definition of TTH, there exists a coordinate k for which $v[k]$ is strictly larger than $n-t$ of the input vectors at coordinate k. Since Byzantine clients are undetectable, these $n-t$ input vectors could be the non-faulty ones. This implies that the output vector v is not in the trusted box, thus violating the box validity condition. This is a contradiction. Hence, the output vector of any algorithm satisfying the box validity condition must be in the TTH. □

Lemma 3.3 (Lower bound for box validity)**.** *The approximation ratio of any algorithm satisfying box validity is at least* $\sqrt{1/2 \cdot \min\{\lfloor (n-t)/t \rfloor, d\}}$*, where* $t > 0$*.*

Proof. In order to prove the lower bound on the approximation ratio, we present a construction where TTH consists of just one vector. Consider a setting where $n - t - \min\left\{\lfloor \frac{n-t}{t} \rfloor t, dt\right\}$ input vectors are at coordinate $(0, \ldots, 0)$. We further assume that t vectors are at coordinate $e_k = x \cdot u_k, \forall k \in \left[\min\left\{\lfloor \frac{n-t}{t} \rfloor, d\right\}\right]$, where u_k is the kth unit vector and $x > 0$. Suppose the t Byzantine vectors choose their input vectors to be $(0, \ldots, 0)$. Then, TTH is $(0, \ldots, 0)$.

The centroid of non-faulty vectors is $\frac{t}{n-t} \sum_{k=1}^{\min\{\lfloor (n-t)/t \rfloor, d\}} e_k$ and the distance between the trimmed trusted hyperbox and $\mathrm{Cent}^\star$ is

$$\mathrm{dist}\big(\mathrm{Cent}^\star, (0, \ldots, 0)\big) = \sqrt{\sum_{k=1}^{\min\{\lfloor (n-t)/t \rfloor, d\}} \left(\frac{t \cdot x}{n-t}\right)^2} = \sqrt{\min\left\{\left\lfloor \frac{n-t}{t} \right\rfloor, d\right\}} \cdot \frac{tx}{n-t}.$$

Now the radius of the minimum covering ball is at most the largest distance between two possible centroids:

$$\mathrm{Rad}_{\mathrm{cov}} \leq \left\| \sum_{k=2}^{\min\{\lfloor (n-t)/t \rfloor, d\}} \left(\frac{t}{n-t} \cdot e_k\right) - \sum_{k=1}^{\min\{\lfloor (n-t)/t \rfloor, d\}-1} \left(\frac{t}{n-t} \cdot e_k\right) \right\|_2 = 2 \cdot \sqrt{\left(\frac{tx}{n-t}\right)^2}.$$

Note that since $t > 0$, $\mathrm{Rad}_{\mathrm{cov}} > 0$ holds in our construction. Hence, the approximation ratio is at least $\mathrm{dist}\big(\mathrm{Cent}^\star, (0, \ldots, 0)\big) / \mathrm{Rad}_{\mathrm{cov}} \geq \sqrt{1/2 \cdot \min\{\lfloor (n-t)/t \rfloor, d\}}$. □

Observe that in the case $t = 0$, the proposed algorithms from Lemma 3.1 compute the true centroid and thus are optimal. We finally consider convex validity. Note that no guarantees can be given for algorithms satisfying convex validity in the case $n \leq \max\{3, d+1\}t$ since the safe area cannot be guaranteed to exist in such cases. The results presented here are therefore only of interest in applications where the number of clients surpasses the dimension of the training model.

This concludes the discussion of the approximation guarantees under box validity. In Appendix A.4, we discuss how the algorithms can be generalized to a setting where the clients perform the vector aggregation in a peer-to-peer setting.

4 Empirical results and discussion

We conclude our work with simulation results for the *FedSGD* protocol. The description of the experimental setup, an analysis of different Byzantine attacks, and an evaluation of the *FedAvg* method are presented in Appendix B. The experiments are run with 100 clients under mild ($\alpha = 1$), moderate ($\alpha = 0.5$), strong ($\alpha = 0.2$) and extreme heterogeneity. Experiments are repeated five times and the variability is shown using shaded error bands in the figures. Figure 2 shows how the MDA and the Box algorithms perform under the *fall of empires* attack [52] with $f \in \{10, 20, 33\}$ Byzantine clients. Overall, the higher number of Byzantine clients affects the system and prevents convergence in more heterogeneous cases. For $f = 33$, MDA fails to converge across all distributions. However, the Box algorithm converges under mild heterogeneity and achieves 69% accuracy. In moderate heterogeneity case, it seems as if Box algorithm could converge, but slowly and requiring a significant amount of rounds. This implies that algorithms satisfying stronger validity conditions are more robust against Byzantine clients.

Figure 3 illustrates Fall of Empires (FoE), A Little Is Enough (ALIE), Sign Flip (SF) and mimic attack in a setting with mild heterogeneity and $f = 49$. Even though MNIST is considered a smaller

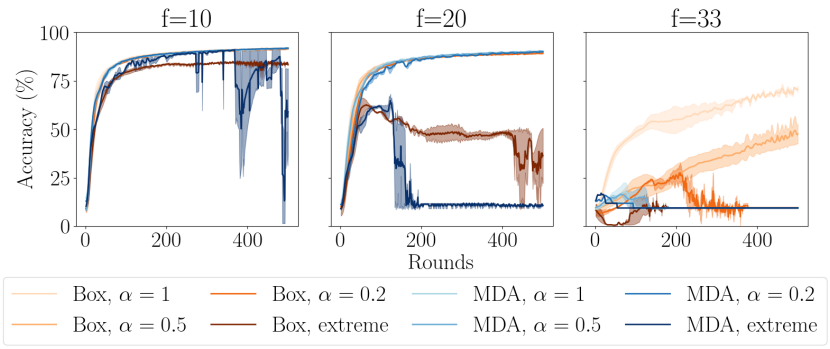


Figure 2: Fall of empires attack with $n = 100, f = \{10, 20, 33\}$ in *FedSGD* setting under mild ($\alpha = 1$), moderate ($\alpha = 0.5$), strong ($\alpha = 0.2$) and extreme heterogeneity

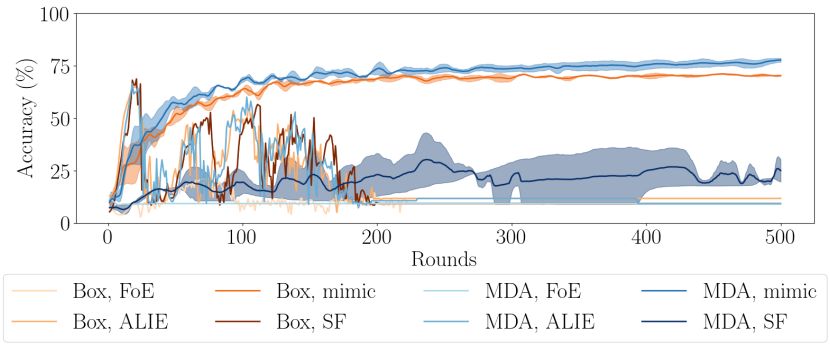


Figure 3: Attacks with $n = 100$, $f = 49$ in *FedSGD* setting under mild ($\alpha = 1$) heterogeneity

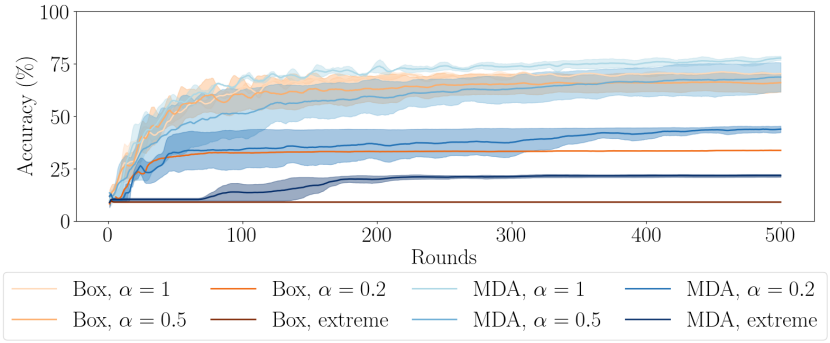


Figure 4: Mimic attack with $n = 100, f = 49$ in *FedSGD* setting under mild ($\alpha = 1$), moderate ($\alpha = 0.5$), strong ($\alpha = 0.2$) and extreme heterogeneity

dataset, the Byzantine-tolerant algorithms struggle to deal with a large numbers of Byzantine nodes. Only Box and MDA algorithm under the mimic attack converge achieving 70% and 76% accuracy, respectively. In Figure 4 we investigate the mimic attack in more detail, by considering different data distributions. Generally, having a more heterogeneous setting lowers the overall accuracy. MDA achieves higher accuracy than the Box algorithm, reflecting to the better approximation results from Table 1. However, MDA also shows small fluctuations in accuracy over rounds, as it satisfies weaker validity conditions. On the other side, box algorithm, which satisfies the box validity condition, seems very stable, but reaches slightly lower accuracy, as it provides a larger approximation of the centroid.

Overall, the experiments support the tradeoff predicted by our theoretical results. Aggregation rules with stronger validity guarantees, such as Box, are often more stable under severe attacks and high data heterogeneity, while algorithms with tighter centroid approximation, such as MDA, can achieve higher accuracy. Thus, robust aggregation in heterogeneous federated learning requires balancing both validity and centroid approximation.

5 Discussion and limitations

We propose centroid approximation in combination with validity conditions as a robustness measure for Byzantine-tolerant aggregation under heterogeneous client data. We show that there is a trade-off between validity conditions and the approximation quality also for aggregation algorithms in FL.

Our analysis is intentionally focused on a single aggregation step. We believe that this is the first step towards building a theoretical foundation for robustness over multiple iterations. Establishing convergence guarantees for FedSGD under centroid approximation and heterogeneous data is another important direction for future work. Additionally, we assume a synchronous setting in which the server receives all client vectors in each round. In the future, we plan to extend our work to partial client participation, stragglers, and asynchronous updates.

Acknowledgements

This work was supported in part by the Research Council of Finland, Grant 333837 and the German Research Foundation (DFG), Schwerpunktprogramm, SPP 2378 (project ReNO), 2023-2027.

References

[1] H Brendan McMahan, Eider Moore, Daniel Ramage, and Blaise Agüera y Arcas. Federated learning of deep networks using model averaging. *arXiv preprint arXiv:1602.05629*, 2(2), 2016.

[2] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 1273–1282. PMLR, 20–22 Apr 2017.

[3] Chen Zhang, Yu Xie, Hang Bai, Bin Yu, Weihong Li, and Yuan Gao. A survey on federated learning. *Knowledge-Based Systems*, 216:106775, 2021.

[4] Consumer data privacy in a networked world: A framework for protecting privacy and promoting innovation in the global digital economy. *Journal of Privacy and Confidentiality*, 4, 03 2013.

[5] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *Foundations and trends® in machine learning*, 14(1–2):1–210, 2021.

[6] 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.

[7] Sashank J. Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečný, Sanjiv Kumar, and Hugh Brendan McMahan. Adaptive federated optimization. In *International Conference on Learning Representations*, 2021.

[8] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. SCAFFOLD: Stochastic controlled averaging for federated learning. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*. PMLR, 2020.

[9] Aritra Mitra, Rayana Jaafar, George J. Pappas, and Hamed Hassani. Linear convergence in federated learning: Tackling client heterogeneity and sparse gradients. In *Advances in Neural Information Processing Systems*, 2021.

[10] Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H. Vincent Poor. Tackling the objective inconsistency problem in heterogeneous federated optimization. NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc.

[11] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020.

[12] Divyansh Jhunjhunwala, Shiqiang Wang, and Gauri Joshi. Fedexp: Speeding up federated averaging via extrapolation. In *The Eleventh International Conference on Learning Representations*, 2023.

[13] Divyansh Jhunjhunwala, Pranay Sharma, Aushim Nagarkatti, and Gauri Joshi. Fedvarp: Tackling the variance due to partial client participation in federated learning. In *Uncertainty in Artificial Intelligence*, pages 906–916. PMLR, 2022.

[14] Cheng Fang, Zhixiong Yang, and Waheed U. Bajwa. Bridge: Byzantine-resilient decentralized gradient descent. *IEEE Transactions on Signal and Information Processing over Networks*, 8:610–626, 2022.

[15] Zhixiong Yang and Waheed U. Bajwa. Byrdie: Byzantine-resilient distributed coordinate descent for decentralized learning. *IEEE Transactions on Signal and Information Processing over Networks*, 5(4):611–627, December 2019.

[16] El-Mahdi El-Mhamdi, Rachid Guerraoui, Arsany Guirguis, Lê Nguyên Hoang, and Sébastien Rouault. Genuinely distributed byzantine machine learning. In *Proceedings of the 39th Symposium on Principles of Distributed Computing*, PODC '20, 2020.

[17] Deepesh Data and Suhas Diggavi. Byzantine-resilient high-dimensional sgd with local iterations on heterogeneous data. In *International Conference on Machine Learning*, pages 2478–2488. PMLR, 2021.

[18] Liping Li, Wei Xu, Tianyi Chen, Georgios B. Giannakis, and Qing Ling. Rsa: Byzantine-robust stochastic aggregation methods for distributed learning from heterogeneous datasets. AAAI'19/IAAI'19/EAAI'19. AAAI Press, 2019.

[19] Avishek Ghosh, Justin Hong, Dong Yin, and Kannan Ramchandran. Robust federated learning in a heterogeneous environment, 2019.

[20] El Mahdi El-Mhamdi, Sadegh Farhadkhani, Rachid Guerraoui, Arsany Guirguis, Lê-Nguyên Hoang, and Sébastien Rouault. Collaborative learning in the jungle (decentralized, byzantine, heterogeneous, asynchronous and nonconvex learning). *Advances in neural information processing systems*, 34:25044–25057, 2021.

[21] Youssef Allouah, Sadegh Farhadkhani, Rachid Guerraoui, Nirupam Gupta, Rafael Pinot, and John Stephan. Fixing by mixing: A recipe for optimal byzantine ml under heterogeneity. In *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, volume 206 of *Proceedings of Machine Learning Research*, pages 1232–1300. PMLR, 25–27 Apr 2023.

[22] Gilles Bareilles, Wassim Bouaziz, Julien Fageot, and El-Mahdi El-Mhamdi. Byzantine machine learning: Multikrum and an optimal notion of robustness, 2026.

[23] Gilles Bareilles, Wassim Bouaziz, Julien Fageot, and El-Mahdi El-Mhamdi. Brief announcement: Byzantine machine learning: Multikrum and an optimal notion of robustness. In *Proceedings of the ACM Symposium on Principles of Distributed Computing*, PODC '26, 2026.

[24] Mélanie Cambus and Darya Melnyk. Centroid approximation with multidimensional approximate agreement protocols. In *Stabilization, Safety, and Security of Distributed Systems*, pages 93–110, 2025.

[25] Hammurabi Mendes, Maurice Herlihy, Nitin Vaidya, and Vijay K. Garg. Multidimensional agreement in byzantine systems. *Distrib. Comput.*, 28(6), 2015.

[26] Diana-Elena Ghinea. *Convex Validity*. PhD thesis, ETH Zurich, 2025.

[27] Zhuolun Xiang and Nitin H. Vaidya. Relaxed Byzantine Vector Consensus. In *20th International Conference on Principles of Distributed Systems (OPODIS 2016)*, Leibniz International Proceedings in Informatics (LIPIcs), pages 26:1–26:15, 2017.

[28] Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Marc' aurelio Ranzato, Andrew Senior, Paul Tucker, Ke Yang, Quoc Le, and Andrew Ng. Large scale distributed deep networks. In *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[29] Mu Li, David G. Andersen, Jun Woo Park, Alexander J. Smola, Amr Ahmed, Vanja Josifovski, James Long, Eugene J. Shekita, and Bor-Yiing Su. Scaling distributed machine learning with the parameter server. In *Proceedings of the 11th USENIX Conference on Operating Systems Design and Implementation*, OSDI'14, page 583–598, USA, 2014. USENIX Association.

[30] Mu Li, David G Andersen, Alexander J Smola, and Kai Yu. Communication efficient distributed machine learning with the parameter server. In *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

[31] Yuchen Zhang, John Duchi, Michael I Jordan, and Martin J Wainwright. Information-theoretic lower bounds for distributed statistical estimation with communication constraints. In *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013.

[32] Ohad Shamir, Nati Srebro, and Tong Zhang. Communication-efficient distributed optimization using an approximate newton-type method. In *Proceedings of the 31st International Conference on Machine Learning*, number 2 in Proceedings of Machine Learning Research, pages 1000–1008, Bejing, China, 22–24 Jun 2014. PMLR.

[33] Jianmin Chen, Rajat Monga, Samy Bengio, and Rafal Jozefowicz. Revisiting distributed synchronous sgd. In *International Conference on Learning Representations Workshop Track*, 2016.

[34] Dan Alistarh, Zeyuan Allen-Zhu, and Jerry Li. Byzantine stochastic gradient descent. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[35] El Mahdi El Mhamdi, Rachid Guerraoui, and Sébastien Rouault. The hidden vulnerability of distributed learning in Byzantium. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 3521–3530. PMLR, 10–15 Jul 2018.

[36] Georgios Damaskinos, El Mahdi El Mhamdi, Rachid Guerraoui, Rhicheek Patra, and Mahsa Taziki. Asynchronous Byzantine machine learning (the case of SGD). In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1145–1154. PMLR, 10–15 Jul 2018.

[37] Cong Xie, Sanmi Koyejo, and Indranil Gupta. Zeno: Distributed stochastic gradient descent with suspicion-based fault-tolerance. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6893–6901. PMLR, 09–15 Jun 2019.

[38] *Adaptive Federated Optimization*, 2021.

[39] Yae Jee Cho, Jianyu Wang, and Gauri Joshi. Towards understanding biased client selection in federated learning. In *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*. PMLR, 2022.

[40] Jinhyun So, Başak Güler, and A. Salman Avestimehr. Byzantine-resilient secure federated learning. *IEEE Journal on Selected Areas in Communications*, 39(7), 2021.

[41] D. Elzinga and Donald Hearn. The minimum covering sphere problem. *Management Science*, 19:96–104, 09 1972.

[42] Pierre Civit, Muhammad Ayaz Dzulfikar, Seth Gilbert, Vincent Gramoli, Rachid Guerraoui, Jovan Komatovic, and Manuel Vidigueira. Byzantine consensus is θ (n^2): The dolev-reischuk bound is tight even in partial synchrony! In *36th International Symposium on Distributed Computing (DISC 2022)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2022.

[43] Pierre Civit, Seth Gilbert, and Vincent Gramoli. Polygraph: Accountable byzantine agreement. In *2021 IEEE 41st International Conference on Distributed Computing Systems (ICDCS)*, pages 403–413. IEEE, 2021.

[44] Maofan Yin, Dahlia Malkhi, Michael K Reiter, Guy Golan Gueta, and Ittai Abraham. Hotstuff: Bft consensus with linearity and responsiveness. In *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing*, pages 347–356, 2019.

[45] Amotz Bar-Noy and Danny Dolev. Families of consensus algorithms. In *VLSI Algorithms and Architectures*, 1988.

[46] Gabriel Bracha. Asynchronous Byzantine Agreement Protocols. *Information and Computation*, 75(2):130–143, 1987.

[47] Gabriel Bracha and Sam Toueg. Resilient consensus protocols. In *Proceedings of the Second Annual ACM Symposium on Principles of Distributed Computing*, PODC '83, 1983.

[48] Danny Dolev, Nancy A. Lynch, Shlomit S. Pinter, Eugene W. Stark, and William E. Weihl. Reaching approximate agreement in the presence of faults. *J. ACM*, 33(3):499–516, May 1986.

[49] Darya Melnyk and Roger Wattenhofer. Byzantine agreement with interval validity. In *2018 IEEE 37th Symposium on Reliable Distributed Systems (SRDS)*, pages 251–260, 2018.

[50] Waseem Abbas, Mudassir Shabbir, Jiani Li, and Xenofon Koutsoukos. Resilient distributed vector consensus using centerpoint. *Automatica*, 136:110046, 2022.

[51] Xuan Wang, Shaoshuai Mou, and Shreyas Sundaram. A resilient convex combination for consensus-based distributed algorithms. *Numerical Algebra, Control and Optimization*, 9(3):269–281, 2019.

[52] Cong Xie, Oluwasanmi Koyejo, and Indranil Gupta. Fall of empires: Breaking byzantine-tolerant sgd by inner product manipulation. In *Uncertainty in artificial intelligence*, pages 261–270. PMLR, 2020.

[53] Qi Xia, Zeyi Tao, Zijiang Hao, and Qun Li. Faba: an algorithm for fast aggregation against byzantine attacks in distributed neural networks. In *IJCAI*, 2019.

[54] M. Pease, R. Shostak, and L. Lamport. Reaching agreement in the presence of faults. *J. ACM*, 27(2), April 1980.

[55] Michael J. Fischer and Nancy A. Lynch. A lower bound for the time to assure interactive consistency. *Information Processing Letters*, 14(4):183–186, 1982.

[56] Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:1909.06335*, 2019.

[57] Sadegh Farhadkhani, Rachid Guerraoui, Nirupam Gupta, Lê-Nguyên Hoang, Rafael Pinot, and John Stephan. Robust collaborative learning with linear gradient overhead. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 9761–9813. PMLR, 23–29 Jul 2023.

[58] Youssef Allouah, Rachid Guerraoui, Nirupam Gupta, Ahmed Jellouli, Geovani Rizk, and John Stephan. Adaptive gradient clipping for robust federated learning. In *International Conference on Learning Representations (ICLR)*, 2025.

[59] Junyu Shi, Wei Wan, Shengshan Hu, Jianrong Lu, and Leo Yu Zhang. Challenges and approaches for mitigating byzantine attacks in federated learning. In *2022 IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pages 139–146. IEEE, 2022.

[60] Battista Biggio, Blaine Nelson, and Pavel Laskov. Poisoning attacks against support vector machines. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, ICML'12, Madison, WI, USA, 2012.

[61] Saeed Mahloujifar, Mohammad Mahmoody, and Ameer Mohammed. Data poisoning attacks in multi-party learning. In *ICML*, pages 4274–4283, 2019.

[62] Sadegh Farhadkhani, Rachid Guerraoui, Nirupam Gupta, and Rafael Pinot. Brief announcement: A case for byzantine machine learning. In *Proceedings of the 43rd ACM Symposium on Principles of Distributed Computing*, PODC '24, pages 131–134, 2024.

[63] Sadegh Farhadkhani, Rachid Guerraoui, Nirupam Gupta, Rafael Pinot, and John Stephan. Byzantine machine learning made easy by resilient averaging of momentums. In *International Conference on Machine Learning*, pages 6246–6283. PMLR, 2022.

[64] Gilad Baruch, Moran Baruch, and Yoav Goldberg. A little is enough: Circumventing defenses for distributed learning. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[65] Richeng Jin, Yufan Huang, Xiaofan He, Huaiyu Dai, and Tianfu Wu. Stochastic-sign sgd for federated learning with theoretical guarantees. *arXiv preprint arXiv:2002.10940*, 2020.

[66] Jeremy Bernstein, Jiawei Zhao, Kamyar Azizzadenesheli, and Anima Anandkumar. signsgd with majority vote is communication efficient and fault tolerant. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[67] Zeyuan Allen-Zhu, Faeze Ebrahimianghazani, Jerry Li, and Dan Alistarh. Byzantine-resilient non-convex stochastic gradient descent. In *International Conference on Learning Representations (ICLR) Posters*, 2021. Poster.

[68] Zhaoxian Wu, Qing Ling, Tianyi Chen, and Georgios B. Giannakis. Federated variance-reduced stochastic gradient descent with robustness to byzantine attacks. *IEEE Transactions on Signal Processing*, 68:4583–4596, 2020.

[69] Yongkang Wang, Yuanqing Xia, and Yufeng Zhan. Elite: Defending federated learning against byzantine attacks based on information entropy. In *2021 China Automation Congress (CAC)*, pages 6049–6054, 2021.

[70] Jian Xu, Shao-Lun Huang, Linqi Song, and Tian Lan. Byzantine-robust federated learning through collaborative malicious gradient filtering. In *2022 IEEE 42nd International Conference on Distributed Computing Systems (ICDCS)*, pages 1223–1235, 2022.

[71] Anee Sharma and Ningrinla Marchang. Probabilistic sign flipping attack in federated learning. In *2024 15th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, 2024.

[72] Sai Praneeth Karimireddy, Lie He, and Martin Jaggi. Byzantine-robust learning on heterogeneous datasets via bucketing. In *International Conference on Learning Representations*, 2022.

[73] Sai Praneeth Karimireddy, Lie He, and Martin Jaggi. Learning from history for byzantine robust optimization. In *International conference on machine learning*, pages 5311–5319. PMLR, 2021.

A Detailed theoretical bounds

A.1 Example for unbounded κ without the knowledge of Byzantine clients

In the following, we provide an example where the algorithm with optimal centroid approximation has an unbounded κ value. Note that this algorithm does not assume any knowledge about the Byzantine clients.

Example A.1. Consider an aggregation algorithm which outputs the center of the minimum covering ball of all candidate centroids, which are all possible centroids computed on subsets of $n - f$ vectors. This algorithm is optimal, as it minimizes the distance to any possible centroid and does not require knowledge of Byzantine clients. In the following, we show that κ may be unbounded for this aggregation algorithm. Assume that the robustness coefficient κ is upper bounded by some constant $u > 0$. Choose $0 < \varepsilon < x\sqrt{\frac{t}{4(n-2t)u}}$. We construct an example which contradicts the assumption $\kappa < u$. Consider the following construction in $d = 1$, where $t < n/2$: place $n - 2t$ nodes at 0, t nodes at x, and t nodes at $-\varepsilon$ and , where $x > 0$. Consider the aggregation rule which outputs the center of the minimum covering ball around the set of all candidate centroids. Note that this aggregation rule requires no knowledge of where the Byzantine nodes are, and it satisfies optimal centroid approximation. Now, the smallest and the largest centroids are $Cent_{min} = -\frac{t\varepsilon}{n-t}$ and $Cent_{max} = \frac{tx}{n-t}$. The output of the aggregation rule is $AGG = \frac{Cent_{min}+Cent_{max}}{2} = \frac{t(x-\varepsilon)}{2(n-t)}$.

Next, consider the definition of robustness with a set S of $n - t$ nodes which does not contain the t values x. Its centroid is $Cent_{min}$. We plug this into the (f, κ) robustness definition:

$$\|AGG - Cent_{\min}\|^2 \leq \frac{\kappa}{n-t} \sum_{i \in S} \left\| x_i - \frac{-t\varepsilon}{n-t} \right\|^2$$

$$\left\| \frac{t(x-\varepsilon)}{2(n-t)} - \frac{-t\varepsilon}{n-t} \right\|^2 \leq \kappa \frac{t(n-2t)\varepsilon^2}{(n-t)^2}$$

$$\frac{t^2(x+\varepsilon)^2}{4(n-t)^2} \leq \kappa \frac{t(n-2t)\varepsilon^2}{(n-t)^2}.$$

So $\kappa \geq \frac{t(x+\varepsilon)^2}{4(n-2t)\varepsilon^2}$. By our choice of ε, $\varepsilon^2 < \frac{tx^2}{4(n-2t)u}$, which implies $\frac{tx^2}{4(n-2t)\varepsilon^2} > u$. This contradicts the assumption that $\kappa < u$, hence, κ cannot be bounded.

A.2 Guarantees given by the validity conditions

Lemma A.2. *Satisfying weak validity is not a sufficient condition for an algorithm to achieve a bounded approximation ratio of* $\mathrm{Cent}^\star$.

Proof. Without loss of generality, we can consider an algorithm that either agrees on the unique input vector, or outputs the origin. Now consider the case where all clients have input $x \cdot (1, \ldots, 1)$. Then, the diameter of the minimum covering ball can be arbitrarily small, but the distance between the origin and $x \cdot (1, \ldots, 1)$ is $\sqrt{d} \cdot x$. Hence, the ratio between this distance and the radius of the minimum covering ball is unbounded.

□

Lemma A.3. *Satisfying strong validity is not a sufficient condition for an algorithm to achieve a bounded approximation ratio of* $\mathrm{Cent}^\star$.

Proof. As before, we can consider an algorithm that either agrees on the unique non-faulty input vector, or outputs the origin (we do not need to know how the algorithm achieves this, only that it is a general algorithm satisfying strong validity). Assume the case, where the $n - t$ non-faulty input vectors are all ϵ away from $(1, \ldots, 1)$, and the Byzantine clients do not send any vector. The distance between the origin and the average of the non-faulty vectors is $\sqrt{d} \cdot x$. The radius of the minimum covering ball is however 0. Hence, the approximation ratio is unbounded. □

Lemma A.4 (from ([24], Observation 4.1))**.** *The worst-case approximation ratio that can be achieved by any algorithm satisfying convex validity is unbounded.*

Next, we show that the box validity condition is the only validity condition that, by itself, guarantees that any algorithm satisfying it has a bounded approximation ratio. More precisely, we show that outputting a vector inside TH is sufficient to ensure that the output is a bounded approximation of $\mathrm{Cent}^\star$.

Lemma A.5. *The worst-case approximation ratio that can be achieved by any algorithm satisfying box validity is at most* $\frac{t}{n-t} \cdot 2\sqrt{d}$.

Proof. Consider the coordinate $k \in [d]$ in which TTH realizes its longest edge. We define a bijection $\phi : [n] \to [n]$ such that, $i < j \Rightarrow v_{\phi(i)}[k] < v_{\phi(j)}[k], \forall i, j \in [n]$. Then,

$$\begin{aligned}
|\mathrm{CH}[k]| &= \frac{1}{n-t}\sum_{i=t+1}^{n} v_{\phi(i)} - \frac{1}{n-t}\sum_{i=1}^{n-t} v_i[k] \\
&= \frac{1}{n-t}\sum_{i=n-t+1}^{n} v_{\phi(i)} + \frac{1}{n-t}\sum_{i=t+1}^{n-t} v_{\phi(i)} - \frac{1}{n-t}\sum_{i=1}^{t} v_i[k] - \frac{1}{n-t}\sum_{i=t+1}^{n-t} v_{\phi(i)} \\
&= \frac{1}{n-t}\sum_{i=n-t+1}^{n} v_{\phi(i)} - \frac{1}{n-t}\sum_{i=1}^{t} v_i[k] \geq \frac{t}{n-t} v_{\phi(n-t)} - \frac{t}{n-t} v_{\phi(t)} = \frac{t}{n-t}|\mathrm{TTH}[k]|.
\end{aligned}$$

Since CH and TTH are necessarily intersecting [24], the furthest a vector satisfying box validity can be from $\mathrm{Cent}^\star$ is if $\mathrm{Cent}^\star$ is in CH and the vector is on the opposite vertex of TTH. We showed above that the diagonal of TTH is at most $\frac{t}{n-t}$ times the diagonal of CH.

The diagonal of CH being upper bounded by $2\sqrt{d} \cdot \mathrm{Rad}_{\mathrm{cov}}$, the furthest we can be from $\mathrm{Cent}^\star$ by satisfying box validity is

$$\left(1 + \frac{t}{n-t}\right) \cdot 2\sqrt{d} \cdot \mathrm{Rad}_{\mathrm{cov}}.$$

The centroid approximation ratio of any algorithm satisfying box validity will hence be upper bounded by $\left(1 + \frac{t}{n-t}\right) \cdot 2\sqrt{d}$.

□

A.3 Lower and upper bounds

In the following, we present the upper bound for weak validity.

Lemma A.6 (upper bound for weak validity)**.** *The best approximation ratio that can be achieved by an algorithm satisfying weak validity is* 1 *in the worst case.*

Proof. We can achieve 1 with the optimum algorithm picking the center of the minimum covering ball (see [24]). This algorithm satisfies weak validity. □

Note that this upper bound is tight, as the lower bound cannot be less than 1 by definition. We now present the algorithm that highlights the upper bound for strong validity.

Lemma A.7 (Upper bound for strong validity)**.** *The MDA algorithm [20] outputs the average of the subset of* $n - t$ *vectors that have the smallest diameter, this diameter is defined as the maximum distance between any two vectors. The MDA computes a* 2*-approximation of the centroid, where* $n > 2t$.

Proof. Observe that the output vector of the MDA algorithm is in $\mathrm{S}_{\mathrm{Cent}}$ and is thus inside $\mathrm{Ball}_{\mathrm{cov}}(\mathrm{S}_{\mathrm{Cent}})$. The largest distance between any two vectors in $\mathrm{Ball}_{\mathrm{cov}}(\mathrm{S}_{\mathrm{Cent}})$ is upper bounded by the diameter of the ball. Thus, the algorithm computes at most a 2-approximation. □

Note that MDA is not the only algorithm achieving 2-approximation under strong validity. The FABA algorithm [53], which discards outlier vectors from the current average, also satisfies strong validity. Assuming that all honest clients hold the same vector v, in every iteration FABA removes the vector farthest from the average of all vectors. Since the honest vectors are in majority and Byzantine vectors are not identical to v, the removed vector is Byzantine. Thus, after f rounds, only the vectors in v stay and are averaged. This also implies that FABA is a 2-approximation of the centroid, because it outputs a candidate centroid in $\mathrm{S}_{\mathrm{Cent}}$ used to compute the minimum covering ball.

The following lemma gives a lower bound of 2 on the approximation ratio of the centroid in the context of strong validity, which matches the upper bound above. This shows that the approximation ratio of the MDA algorithm is tight.

Lemma A.8 (Lower bound for strong validity [24])**.** *The best approximation ratio that can be achieved by an algorithm satisfying strong validity is* 2 *in the worst case.*

We finally present the upper and lower bounds for convex validity below.

Lemma A.9 (Upper bound for convex validity)**.** *Consider the algorithm that outputs a vector contained in the safe area that minimizes the distance to the center of* $\mathrm{Ball}_{\mathrm{cov}}(\mathrm{S}_{\mathrm{Cent}})$*. This algorithm computes a* $2d$*-approximation of the centroid, when* $n > \max\{3, d+1\}t$*.*

Proof. Observe that the algorithm computes at most a 2-approximation of $\mathrm{Cent}^\star$ if the *safe area* and $\mathrm{Ball}_{\mathrm{cov}}(\mathrm{S}_{\mathrm{Cent}})$ intersect. This is because the algorithm then chooses a vector in $\mathrm{Ball}_{\mathrm{cov}}(\mathrm{S}_{\mathrm{Cent}})$.

Now we consider the remaining case, where the *safe area* and $\mathrm{Ball}_{\mathrm{cov}}(\mathrm{S}_{\mathrm{Cent}})$ are disjoint. Let x denote the distance between the *safe area* and $\mathrm{Ball}_{\mathrm{cov}}(\mathrm{S}_{\mathrm{Cent}})$ and let S denote the closest point of the *safe area* to $\mathrm{Ball}_{\mathrm{cov}}(\mathrm{S}_{\mathrm{Cent}})$ and B the closest point of $\mathrm{Ball}_{\mathrm{cov}}(\mathrm{S}_{\mathrm{Cent}})$ to the *safe area*, so that the distance between S and B is x. We start by projecting all input vectors orthogonally onto $\overline{S,B}$. The approximation ratio of the algorithm is computed as $(x+\mathrm{Rad}_{\mathrm{cov}})/\mathrm{Rad}_{\mathrm{cov}}$. Observe that the distance between any two centroids after their orthogonal projection onto $\overline{S,B}$ cannot increase due to the triangle inequality, while the distance between S and B remains unchanged. Therefore, the distance between any two projected centroids onto $\overline{S,B}$ is a lower bound on the diameter of $\mathrm{Ball}_{\mathrm{cov}}(\mathrm{S}_{\mathrm{Cent}})$. To simplify the discussion on distances, we assume that S is at coordinate 0 and B is at coordinate x.

In the following, we will lower bound the size of $\mathrm{Rad}_{\mathrm{cov}}$ and upper bound the size of x. Let the projection of the vectors $v_1, \ldots, v_n$ be denoted $p_1, \ldots, p_n$ such that the vector projected on the smallest coordinate is denoted p_1 and the one projected on the largest coordinate is p_n. Note that there must be at least $t+1$ vectors p_i having negative coordinates, otherwise there would exist a convex hull of $n-t$ vectors that would project onto only strictly positive coordinates, which is a contradiction. There are also at least $t+1$ vectors p_j that have coordinate at least x. If this was not true, there would exists a centroid with a smaller coordinate than x, which is a contradiction. Further, there are at most td vectors with a positive coordinate (see proof of Lemma A.10).

Let l denote the number of vectors p_i with negative coordinates. Let r denote the number of vectors p_i with a larger coordinate than x, and let $y_1, \ldots, y_r$ denote the coordinates of these vectors in increasing order. Further, we say that the smallest $r-t$ coordinates have an average value of $\overline{y}_{min}$ while the largest t coordinates have an average of $\overline{y}_{max}$. The average of all vectors p_i with coordinates between 0 and x is defined to be a.

Observe that x is upper bounded by the coordinate of any possible centroid. We choose the following centroid to upper bound x: the average of some $t+1$ vectors with negative coordinates, all the vectors between 0 and x, and the remaining smallest $r-t$ vectors with coordinates larger than x. This gives the following bound:

$$x \leq \frac{1}{n-t}\left(\sum_{i=1}^{r-t} y_i + a\cdot(n-r-l)\right) \leq \frac{n-t-l}{n-t}\cdot\overline{y}_{min}$$

Note that we upper bounded all vectors with coordinates smaller than 0 by 0.

To lower bound the diameter of $\mathrm{Ball}_{\mathrm{cov}}(\mathrm{S}_{\mathrm{Cent}})$, we consider the difference between its largest and smallest coordinates:

$$\mathrm{Rad}_{\mathrm{cov}} \geq \frac{1}{2(n-t)}\left(\sum_{i=t+1}^{n} p_i - \sum_{i=1}^{n-t} p_i\right) \quad \geq \frac{1}{2(n-t)}\left(t\cdot\overline{y}_{max} - \sum_{i=1}^{t} p_i\right) \geq \frac{t}{2(n-t)}\cdot\overline{y}_{max}$$

where $\frac{1}{n-t}\sum_{i=1}^{t} p_i \leq 0$ since there are at least $t+1$ vectors p_i with negative coordinates.

The approximation ratio achieved by the algorithm can now be upper bounded by:

$$\frac{x}{\mathrm{Rad}_{\mathrm{cov}}} + 1 \leq \frac{\frac{1}{n-t}(n-t-l)\cdot \overline{y}_{min}}{\frac{t}{2(n-t)}\cdot \overline{y}_{max}} + 1 \leq \frac{2(n-t-l)}{t} + 1 \leq \frac{2dt}{t} + 1 = 2d+1.$$

The last inequality holds because there can be at most dt vectors with positive coordinates, i.e., $n-t-l \leq dt$. □

Lemma A.10. *Assume that the safe area is a q-dimensional convex polytope, where $1 \leq q \leq d$. Consider the q-dimensional subspace in which the safe area is defined. Let H be a hyperplane that touches the safe area and divides the q-dimensional space into two subspaces. Then, there can be at most qt points on the opposite side of H wrt. the safe area.*

Proof. Consider a vertex s_v of the safe area that lies at the intersection of the *safe area* with the hyperplane H. Note that at least one such vertex must exist since the *safe area* is a convex polytope.

Observe that exactly q $(q-1)$-faces of *safe area* meet in s_v. Each of these faces are hyperplanes , denoted $H_1, \ldots, H_d$, and go through s_d, each of them defined by a face of the safe area. The *safe area* is defined such that, for each face F_i, at most t vectors can lie outside of *safe area* and thus on the opposite side of H w.r.t. *safe area*. In total, at most qt can lie on the opposite side of H. And at least $n-qt > n-dt$ vectors must lie inside *safe area*. □

Lemma A.11 (Lower bound for convex validity)**.** *The best approximation ratio that can be achieved by an algorithm satisfying convex validity is at least $2d$.*

Proof. In [24], a lower bound of $2d$ has been shown for the worst case $n = (d+1)t+1$. This proof can be easily extended to hold for the general case $n > \max\{3, d+1\}\cdot t$. Assume that dt vectors are placed at coordinates $x + \varepsilon \cdot u_i, i \in \{1, \ldots, d\}$, where ε is a small constant and t vectors placed at each coordinate. The remaining $n-dt$ vectors are placed at $(0, \ldots, 0)$. Assume that these $n-dt$ vectors include t Byzantine vectors. Observe that such a construction is always possible since $n > (d+1)t$.

In [24], it was shown that the safe area of such a construction results in a single point $(0, \ldots, 0)$. Note that the non-faulty centroid is located in $td/(n-t)$, and the radius of the centroid ball is $t/(2(n-t))$. Thus, the approximation of the centroid is $2d$ in this example. □

A.4 Federated learning in peer-to-peer networks

The results presented in this paper also hold for federated learning in synchronous peer-to-peer networks when $n > 3t$. In the peer-to-peer setting, there is no trusted server. Instead, the clients communicate with each other in a fully-connected network by sending messages. The aggregation step by the server is replaced by an exact Byzantine agreement algorithm that makes sure that the clients agree on the same aggregation vector. The lower bounds presented in Section 3.1 and 3.2 trivially extend to this distributed setting, as they are presented for a stronger setting in which the clients do not receive different sets of vectors as it is possible in a peer-to-peer setting. On the other hand, interactive consistency protocols [54, 55] from distributed computing allow the clients to agree on the same set of vectors. Thus, each client can apply the presented aggregation algorithms locally. Since the algorithms are deterministic, all clients output identical vectors after Byzantine agreement.

B Empirical evaluation

In the practical evaluation, we differentiate between the two federated learning variants where the model parameters or the gradients are exchanged. We consider n clients, where each client $i \in [n]$ has access to its own data that follows an unknown distribution $\mathcal{D}_i$. Let $F_i(x)$ be the local loss function of client i with respect to model parameter x. The objective is

$$\arg\min_{x\in\mathbb{R}^d} F(x), \quad \text{where} \quad F(x) = \frac{1}{n}\sum_{i=1}^{n} F_i(x)$$

The training is executed in rounds. We differentiate between the following two settings:

FedSGD In each round r, a client locally computes the gradient $g_i(x_r) = \nabla F_i(x_r)$ on its dataset. It then sends $g_i(x_r)$ to the server. The server upgrades the global model by aggregating the gradients $x_{r+1} \leftarrow x_r - \eta \frac{1}{n} \sum_{i=1}^{n} g_i(x_r)$, where η is a fixed learning rate, and sends the new model to the clients for the next round.

FedAvg In each round r, a client locally updates its model parameters (possibly multiple times) $x_{r+1}^i \leftarrow x_r^i - \eta g_i(x_r)$. It then shares its model parameter x_{r+1}^i with the server. The server aggregates the model parameters $x_{r+1} \leftarrow \sum_{i=1}^{n} x_{r+1}^i$ and shares the new model with the clients.

The aggregation algorithm in the definition of *FedSGD* and *FedAvg* is an unweighted average of the vectors. For the experiments, we replace this aggregation step with one of the aggregation algorithms presented in Section 3.2. These aggregation algorithms are summarized below.

Aggregation algorithms We implemented the following aggregation algorithms for comparison:

- **Center of** $\mathrm{Ball}_{\mathrm{cov}}(\mathrm{S}_{\mathrm{Cent}})$**:** This algorithm computes all possible centroids on subsets of $n-t$ vectors and outputs the center of $\mathrm{Ball}_{\mathrm{cov}}(\mathrm{S}_{\mathrm{Cent}})$. The algorithm achieves a 1-approximation of the centroid and satisfies weak validity.
- **MDA [20]:** This algorithm computes a subset of $n - t$ vectors with the smallest diameter and outputs the centroid of this subset. The algorithm achieves a 2-approximation of the centroid and satisfies strong validity.
- **Box Algorithm [24]:** This algorithm computes the intersection of TTH and CH, and outputs the center of this intersection. In [24], it was shown that such an intersection is non-empty for $n > 3t$. The algorithm achieves a $2\sqrt{d}$-approximation of the centroid and satisfies box validity.

We do not implement the algorithm based on the *safe area* (see Lemma A.9), since this algorithm only works in scenarios where $n > (d + 1)t$.

B.1 Distributed mean-estimation

We first study the distributed mean-estimation problem with synthetic data. This experiment removes the optimization dynamics of FedSGD and FedAvg and focuses only on the aggregation step. This controlled setting allows us to illustrate the theoretical tradeoff between validity and centroid approximation before evaluating the same aggregation rules in full federated learning experiments. In the current setup, honest vectors are generated synthetically using the Dirichlet distribution. We set $n = 15$ and $f = 7$. We consider the fall of empires (FOE), a little is enough (ALIE) and the signflip attacks and compare the Box algorithm and MDA with the plain mean of all inputs as a baseline. We additionally test different attack severities, where the gradient of the Byzantine client is multiplied by a factor. For evaluation, we measure the approximation ratio $\frac{|g - Cent^*|_2}{Rad}$ and the distance between the output and the trusted box (TH).

Figures 5, 6 and 7 illustrate the distributed mean estimation with mild, moderate and strong heterogeneous data. Accross all scenarios, the Box algorithm remains inside the TH for all tested attack severities, while the mean increasingly violates this. For extreme attacks severities, MDA successfully removes Byzantine clients from the minimum diameter subset and computes the true centroid. Intermediate and less severe attacks blend in with the true input nodes, hence produce a larger approximation ratio and the output may lie outside the TH. In this scenario, the Box algorithm provides a better approximation.

Interestingly, the approximation ratio of MDA under ALIE attack with severity 0.5 and 1 is high throughout different heterogeneities. However the output is inside the TH. Upon further investigation, we discovered that MDA in this case does not output the true centroid, however the output is inside the TH.

These results illustrate that centroid approximation and validity notions capture different aspects of robustness and that stronger validity constraints can improve stability when the attacks have lower severity.

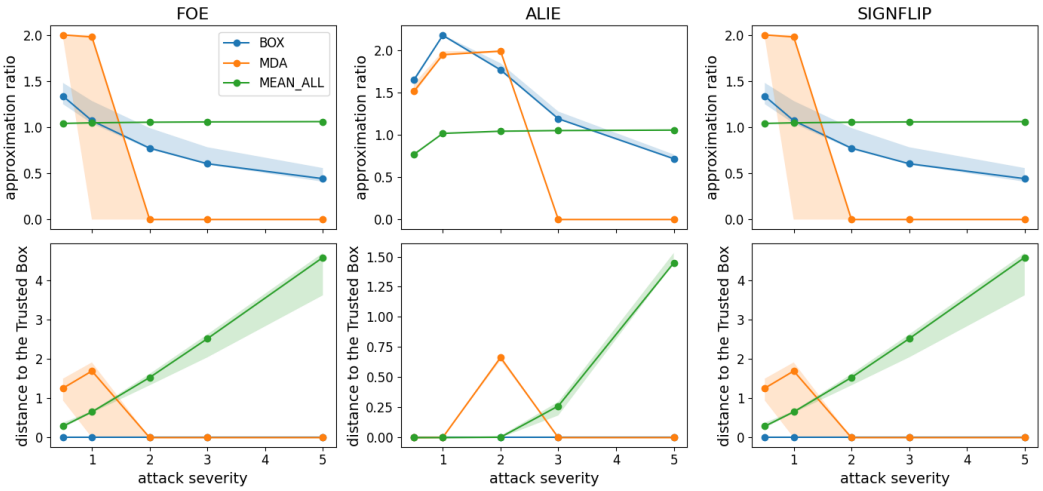


Figure 5: Distributed mean-estimation with mildly heterogeneous data

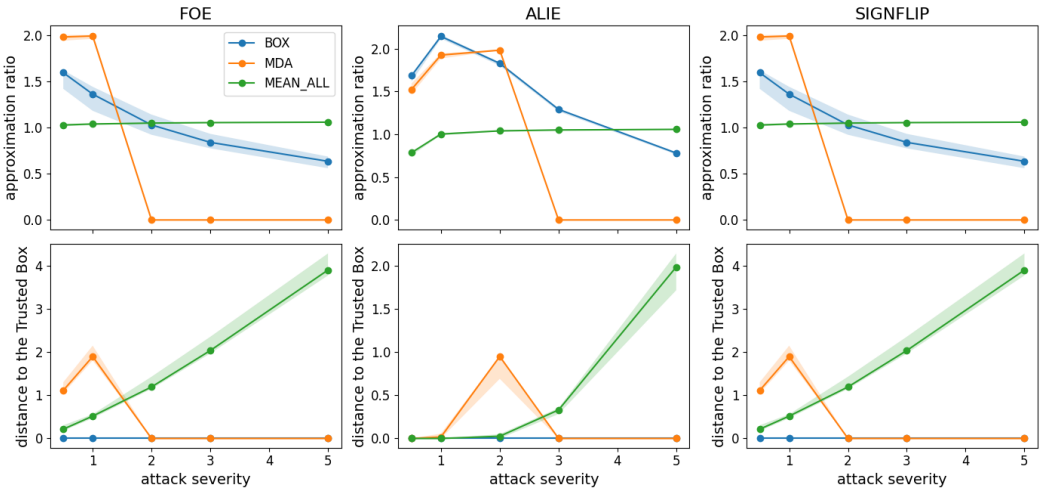


Figure 6: Distributed mean-estimation with moderately heterogeneous data

B.2 Experimental setup

We implement a client/server federated learning model for solving classification tasks in Python using the Tensorflow library. The models are evaluated on the MNIST dataset from Kaggle[1]. The dataset contains 42,000 images of handwritten digit in JPEG format which are labeled, and each class of the data is kept in a separate folder. We consider a setting with 30 clients and assume that a constant fraction of them are Byzantine. We use $f < n/3$ to denote the actual number of Byzantine clients present in the system. To simulate data heterogeneity in our experiments, we consider the Dirichlet distribution with parameter α [56], as done in [21, 57, 58]. Parameter α indicates the level of heterogeneity sampled by clients' datasets. Smaller values of α indicate a more heterogeneous setting, where a client likely owns data only from a very few classes. In line with [58], we consider three values for α: $\alpha = 1$ representing mild heterogeneity, $\alpha = 0.5$ representing moderate heterogeneity, and $\alpha = 0.1$ representing strong heterogeneity. We repeat the experiments three times and show the variability using shaded error bands. Additionally, we consider the extreme heterogeneous case, where the data is sorted by classes and distributed among clients such that each client possesses up to

[1] `https://www.kaggle.com/datasets/scolianni/mnistasjpg`

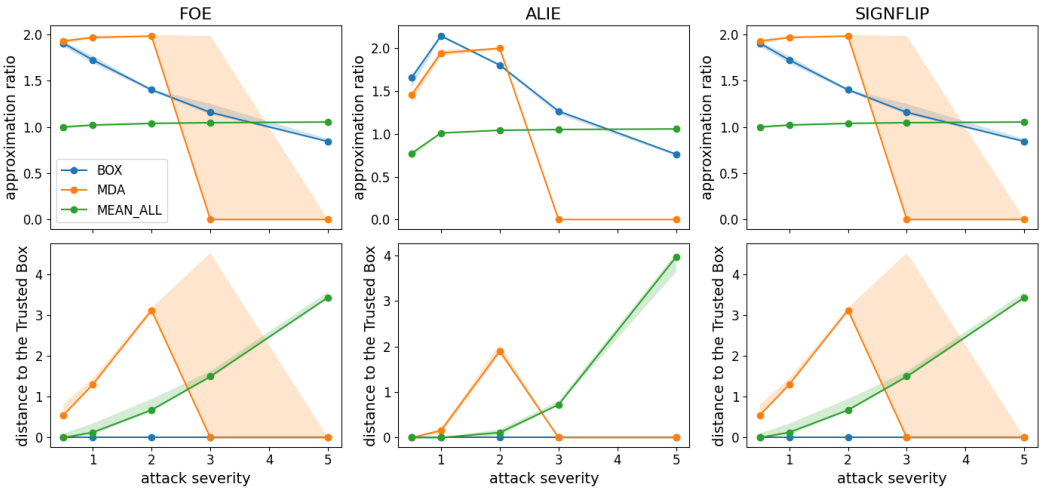


Figure 7: Distributed mean-estimation with strongly heterogeneous data

two different classes of data. Note that in Section 4 we considered a setting with $n = 100$ clients and strong heterogeneity with $\alpha = 0.2$. That is because the data distribution with $\alpha = 0.1$ created clients with less images than what is required to sample a batch, so training these clients is not possible.

The underlying neural network for solving the image classification task is a MultiLayer Perceptron (MLP) with 3 layers. The learning rate is set to $\eta = 0.01$ and the decay is calculated with respect to the number of global communication rounds (epochs), i.e. $decay = \frac{\eta}{rounds}$. The batch size is set to 32.

Byzantine behavior in federated learning has been extensively studied in the literature, and the attacks has been categorized into training-based and parameter-based attacks [59]. Training-based attacks, also known as data poisoning attacks, have been analyzed in [60, 61, 62]. In order to match our theoretical analysis and due to simplicity in implementation, our work considers parameter-based attacks [59, 63], where the adversary can alter only its messages, while the datasets and the training process remain unchanged.

Our experiments consist of the following attacks:

- **Fall of empires** (FOE): Byzantine clients compute the mean of honest nodes' input, reverse it and scale it by $\epsilon > 0$ [52]. We set $\epsilon = 1$.
- **A little is enough** (ALIE): Byzantine clients estimate mean μ and standard deviation σ of honest nodes and send $\mu - z \cdot \sigma$ to the server [64]. We set $z = 1$.
- **Sign flip** (SF): inspired by the signSGD algorithm [65, 66], the gradient of the faulty clients is multiplied by -1 and sent to the server [67]. This attack has been widely used in practical simulations [68, 69, 63, 70, 71].
- **Mimic**: Byzantine clients imitate one fixed honest client by simply sending its gradient to the server [72].

B.3 Experimental results

***FedSGD* setting:** Figure 8 shows how Box and MDA algorithms perform under the *mimic* attack with $f \in \{3, 6, 9\}$ Byzantine clients. When there are three adversarial clients present in the system, both MDA and Box algorithms converge and achieve up to 93% accuracy. Small accuracy differences appear between mild and extreme heterogeneous distributions, caused by the stronger heterogeneity. With $f = 6$, Box algorithm under all but extreme heterogeneous distribution converge. MDA with extremely heterogeneous datasets seems to converge after expressing instability with 88% accuracy. When the number of Byzantine clients is increased to $f = 9$, Box algorithm under extreme and strong heterogeneity struggles to converge. With mild and moderate heterogeneity, box algorithm converges

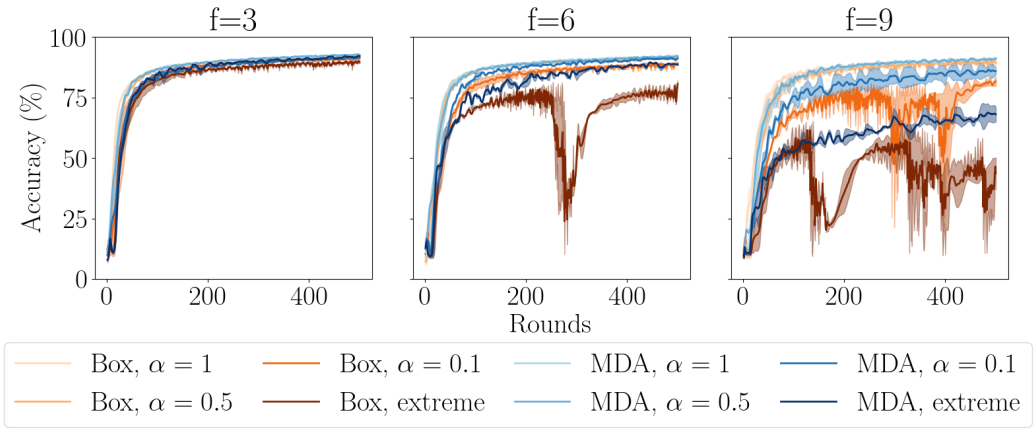


Figure 8: Mimic attack with $f = \{3, 6, 9\}$ in *FedSGD* setting under mild ($\alpha = 1$), moderate ($\alpha = 0.5$), strong ($\alpha = 0.1$) and extreme heterogeneity

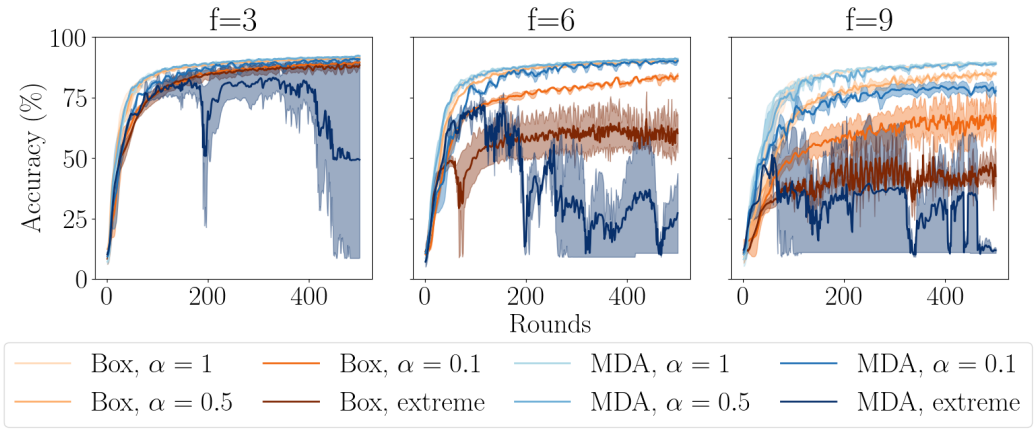


Figure 9: Sign flip attack with $f = \{3, 6, 9\}$ in *FedSGD* setting under mild ($\alpha = 1$), moderate ($\alpha = 0.5$), strong ($\alpha = 0.1$) and extreme heterogeneity

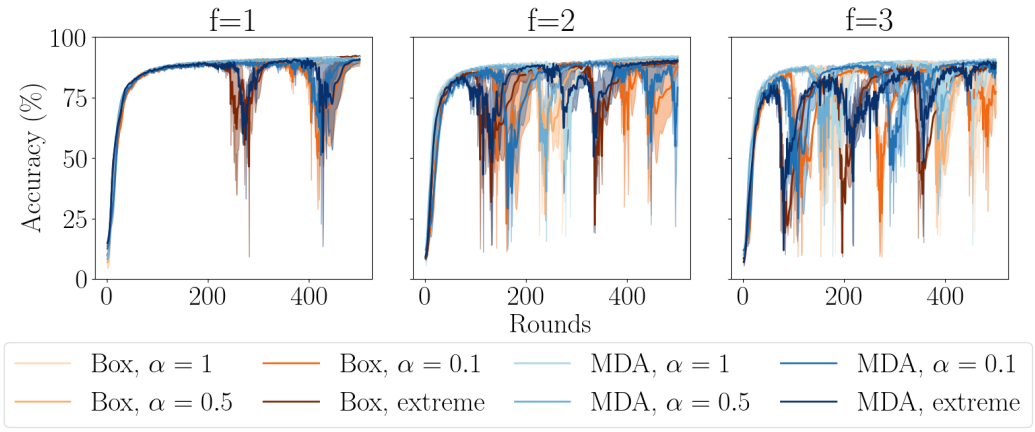


Figure 10: A little is enough attack with $f = \{1, 2, 3\}$ in *FedSGD* setting under mild ($\alpha = 1$), moderate ($\alpha = 0.5$), strong ($\alpha = 0.1$) and extreme heterogeneity

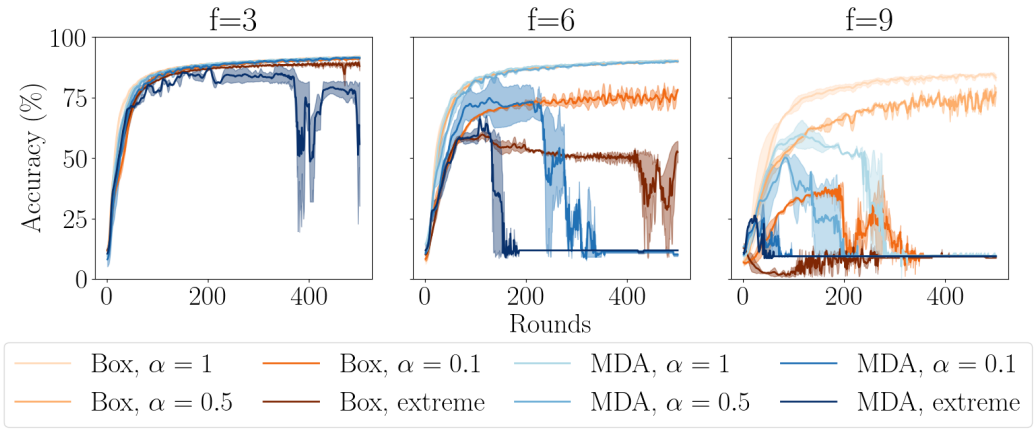


Figure 11: Fall of empires attack with $f = \{3, 6, 9\}$ in *FedSGD* setting under mild ($\alpha = 1$), moderate ($\alpha = 0.5$), strong ($\alpha = 0.1$) and extreme heterogeneity

and reaches over 90% accuracy. MDA is more resilient against the mimic attack, as it converges in mild and moderate heterogeneous setting with over 90% accuracy. In stronger heterogeneity setting, MDA is instable but converges achieving accuracy lower than 65%. Since there are 9 adversarial clients ($f = t = 9$) that imitate one honest client (in total 10 clients with the same input), the subset of $n - t$ nodes with the minimum diameter will always contain at least one of these clients. Furthermore, the trusted hyperbox removes t smallest and largest value in each dimension, most likely leaving in the majority of the Byzantine input. Hence, MDA is less influenced by the mimic attack than the Box algorithm.

Figure 9 illustrates the effect of *sign flip* attack on Box and MDA algorithms with $f \in \{3, 6, 9\}$ Byzantine clients. With $f = 3$, MDA and Box algorithms converge reaching 93% and 91% accuracy. In the extreme heterogeneous setting, MDA fails to converge. If the number of adversarial clients is $f = 6$, it can be observed that the Box algorithm is unstable under the extreme heterogeneous setting. With strong heterogeneity, the Box algorithm converges with lower accuracy than MDA, namely 85% compared to 90%. When $f = 9$, Box algorithm with strong and moderate heterogeneity converge achieving 62% and 84% accuracy. On the other hand, MDA converges reaching higher accuracy than the Box algorithm, which reflects to our theoretical results showing that MDA is a 2- and Box $2\sqrt{d}$-approximation of the centroid.

Figure 10 depicts performance of Box and MDA algorithms under the *a little is enough* attack with $f \in \{1, 2, 3\}$ adversarial clients. Firstly, we consider a lower number of Byzantine clients, as the attack already affects the system with $f = 2$ and $f = 3$. When there is one adversarial client in the system, both MDA and Box algorithm with mild and moderate heterogeneity converge and reach 92% accuracy. It can be observed that with stronger heterogeneities, both MDA and Box struggle to converge and sudden drops in accuracy occur at regular intervals (around every 250 rounds). With the increased number of Byzantine clients, all algorithms experience these drops. However, as heterogeneity increases, drops in accuracy become more frequent. For example, when $f = 3$, Box algorithm with extreme heterogeneity exhibits accuracy cliffs around every 100 rounds. Overall, the attack induces regular, heterogeneity-dependent accuracy drops that intensify with larger f. Both Box and MDA fail to maintain stable convergence under these settings. Note that the batch size is 32. Small batches (e.g., 32) increase variance in honest gradients, making the *a little is enough* and *fall of empires* attacks harder to detect and defend against [73].

Figure 11 shows how the MDA and the Box algorithms perform under the *fall of empires* attack [52] with $f \in \{3, 6, 9\}$ Byzantine clients. For $f = 3$, MDA and Box algorithm achieve up to 93% accuracy. However, under extreme heterogeneity, MDA does not converge, and the Box algorithm reaches up to 90% accuracy. For $f = 6$, MDA fails under extreme and strong heterogeneity, whereas Box still converges under strong heterogeneity with lower accuracy. For $f = 9$ Byzantine clients are present, MDA fails across all distributions. However, the Box algorithm converges under mild and moderate heterogeneity and achieves 82% and 75% accuracy, respectively. These results suggest that

there may be a trade-off between the centroid approximation and the different validity conditions, also in practice, which we plan to investigate in more detail in future work.

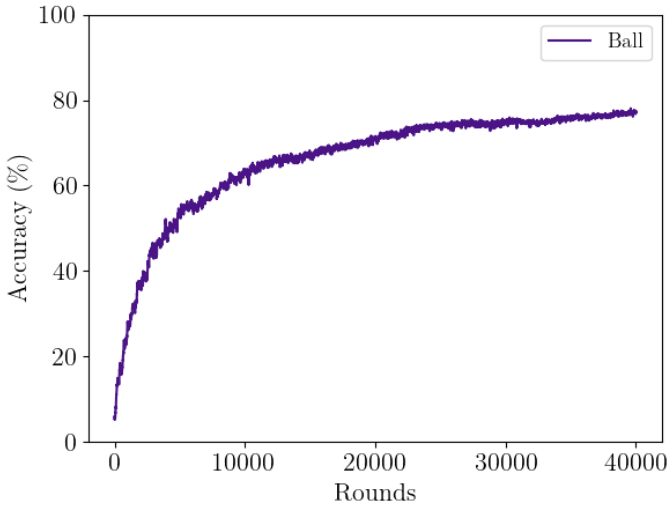


Figure 12: *FedSGD* setting with $\mathrm{Ball}_{\mathrm{cov}}(\mathrm{S}_{\mathrm{Cent}})$ algorithm

Figure 12 illustrates the Center of $\mathrm{Ball}_{\mathrm{cov}}(\mathrm{S}_{\mathrm{Cent}})$ algorithm in the *FedSGD* setting with no Byzantine behavior. It can be observed that after 40,000 rounds $\mathrm{Ball}_{\mathrm{cov}}(\mathrm{S}_{\mathrm{Cent}})$ algorithm reaches over 77%. The Center of $\mathrm{Ball}_{\mathrm{cov}}(\mathrm{S}_{\mathrm{Cent}})$ algorithm requires significantly more rounds than the MDA or the Box algorithm and is therefore not evaluated under Byzantine behavior and different heterogeneity distributions.

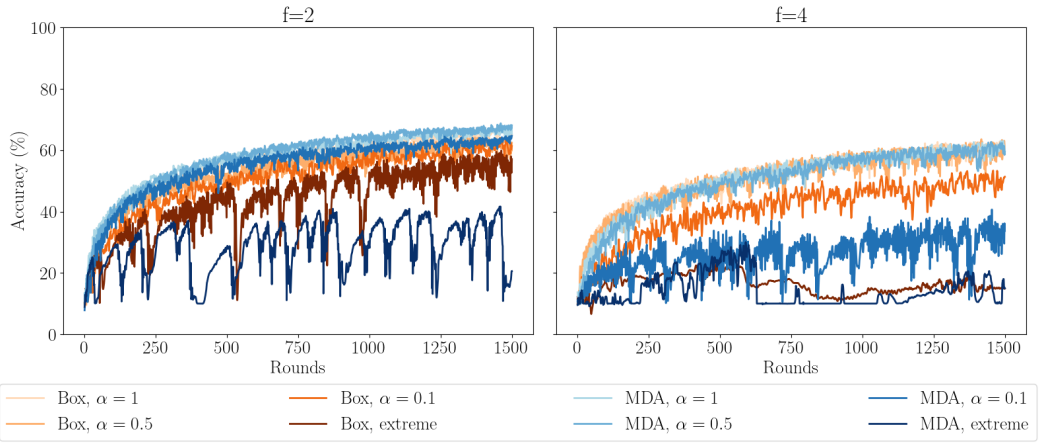


Figure 13: Fall of empires attack with $f = \{2, 4\}$ in *FedSGD* setting under mild ($\alpha = 1$), moderate ($\alpha = 0.5$), strong ($\alpha = 0.1$) and extreme heterogeneity on CifarNet

CifarNet: We additionally test the MDA and Box algorithm on CIFAR10 dataset, which has 60,000 32×32 color images in 10 classes, out of which 50,000 are training images and 10,000 test images. For the CIFAR10 dataset we implemented CifarNet, a medium-sized convolutional network with thousands of trainable parameters and the ability to capture spatial relationships in colored images. For the experiments, we assume $n = 20$. CIFAR10 is a more complex dataset than MNIST and the experiments require a larger number of training rounds. Hence, we were computationally limited and could test out 20 clients out of which $f = \{2, 4, 6\}$ are faulty.

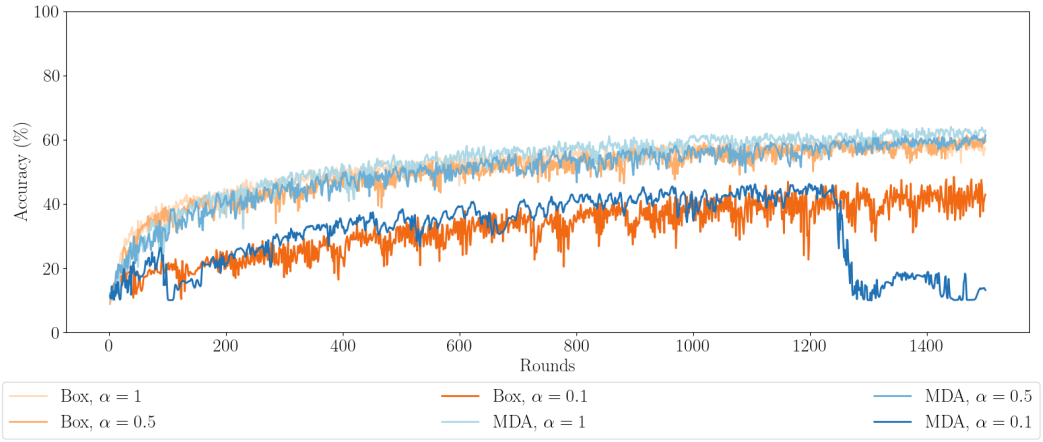


Figure 14: Sign flip attack with $f = 4$ in *FedSGD* setting under mild ($\alpha = 1$), moderate ($\alpha = 0.5$) and strong ($\alpha = 0.1$) heterogeneity on CifarNet

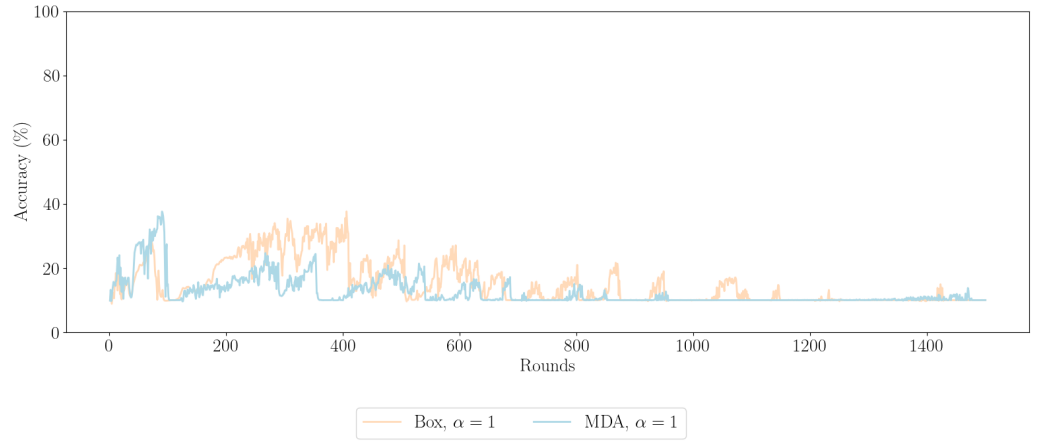


Figure 15: A little is enough attack with $f = 2$ in *FedSGD* setting under mild ($\alpha = 1$) heterogeneity on CifarNet

Figure 13 illustrates the fall of empires attack under 2 and 4 Byzantine clients. Overall, the accuracy of the models training on the CIFAR10 dataset is significantly lower than training on the MNIST dataset. Additionally, with the more heterogeneous setting, fluctuations in accuracy become more evident. Similar to the results in Figure 2, MDA approach achieves a slightly higher accuracy than the Box algorithm in less heterogeneous settings, which complies with the better approximation results of the centroid. However, with increased heterogeneity between the clients, the Box algorithm outperforms the MDA approach, as it is more robust and satisfies a stronger validity condition than MDA. Hence, we can conclude that stronger validity condition is responsible for providing a more robust solution against Byzantine attacks.

Similar can be concluded from the Figure 14, which depicts the sign flip attack with $f = 4$ under mild, moderate and strong heterogeneous data. Box algorithm converges with low accuracy, whereas MDA fails to converge with strong heterogeneity.

Figure 15 shows a little is enough attack with $f = 2$ with mild heterogeneity. In correspondence to Figure 10 with $f = 3$, MDA and Box fail to converge when 10% of the clients are Byzantine.

Figure 16 illustrates the mimic attack with $f = 6$. Box algorithm does not seem to converge under strong heterogeneity, which complies with the results from Figure 8 with $f = 9$ on MNIST dataset. MDA achieves higher accuracy and is more robust against the mimic attack.

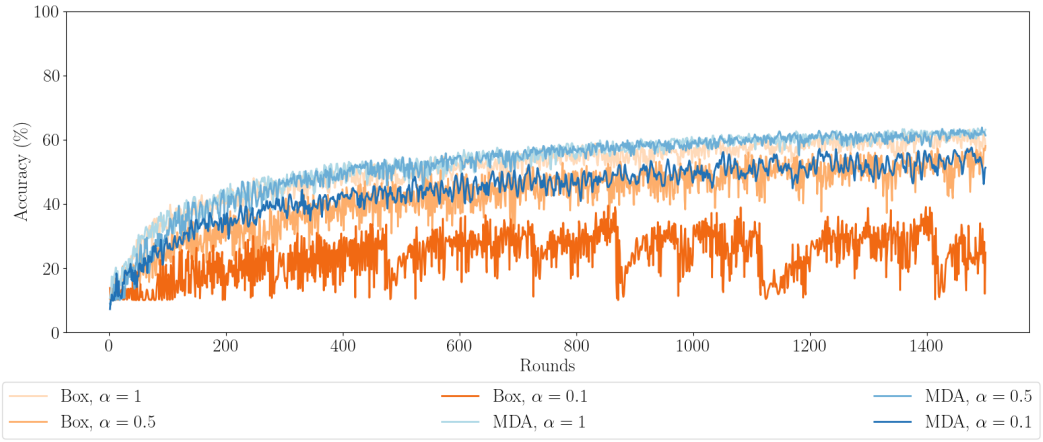


Figure 16: Mimic attack with $f = 6$ in *FedSGD* setting under mild ($\alpha = 1$), moderate ($\alpha = 0.5$) and strong ($\alpha = 0.1$) heterogeneity on CifarNet

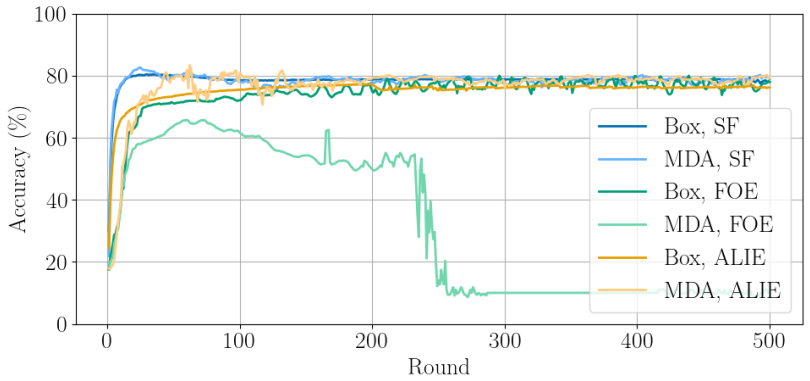


Figure 17: *FedAvg* setting with mild heterogeneous data distributions with $f = 1$

***FedAvg* setting:** Figure 17 illustrates *FedAvg* setting with mild heterogeneous data distribution. In this experiment, we set $f = 1$ and evaluate the algorithms on *sign flip* and *a little is enough*. Additionally, we lower the learning rate to $\eta = 0.001$, since the higher learning rate causes client drift in this setting. In the *sign flip* attack, MDA and Box converge achieving 80% and 78% accuracy, respectively. Nevertheless, MDA shows small differences of 3% in accuracy, whereas the Box algorithm converges smoothly.

In the *a little is enough* attack, accuracy drops slightly to 76% when using the Box algorithm. MDA reaches higher accuracy than Box algorithm (78%), but it is more unstable and shows small differences in accuracy, similar to the ones in the *sign flip* attack.

Under the *fall of empires* attack, Box algorithm converges and reaches 78% accuracy. However, MDA algorithm fails to converge. Compared to the MDA algorithm under *sign flip* or *a little is enough* attack, the *fall of empires* attack has a larger impact and prevents MDA from converging, similar to the results in the *FedSGD* setting.

In future, we intend to continue the empirical evaluation and test out *FedAvg* in different scenarios.

C Impact statement

Our work focuses on developing fault-tolerant federated learning models for clients that hold heterogeneous data. Our methods are designed to find a balance between including data outliers while

tolerating malicious client behavior. This is a theoretical study for which we do not foresee any negative impact. Our practical evaluation uses open-source datasets to evaluate the performance of our methods.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Abstract and introduction clearly state the paper's contribution and scope, including theoretical and experimental results.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The assumptions are explicitly mentioned in Section 2 and Appendix B, and a discussion is provided in the Contributions part of Section 1 and in Section 4. Limitations of the work are discussed in Section 5.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: All theorems, formulas and proofs are numbered and referenced. The assumptions are stated clearly in Section 2.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Section 4 and Appendix B provide all necessary information on the implementation. Additionally, we plan to release the source code in the final version of the paper.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We will release our source code for the final version of the paper.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: All information on the experimental setting can be found in Section 4 as well as in Appendix B.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The main experiments supporting the claims in Section 4 are repeated over five independent runs and report shaded error bands showing variability across runs. Some additional appendix experiments are exploratory single-run evaluations due to computational constraints.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Information can be found in Section 4 and in Appendix B.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conducted in the paper conforms in all aspects with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We provide a discussion on the impact of our work in Appendix C.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.

- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: This work uses available open-source datasets.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: See Section 4 and Appendix B.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We will release our source code. The information and details on implementation can be found in Appendix B

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.