Byzantine-robust Histogram Estimation in the Shuffle Model

Leixia Wang*
Renmin University of China
leixiawang@ruc.edu.cn

Qingqing Ye
Hong Kong Polytechnic University
qqing.ye@polyu.edu.hk

Haibo Hu
Hong Kong Polytechnic University
haibo.hu@polyu.edu.hk

Kai Huang
Hong Kong University of Science and Technology
ustkhuang@ust.hk

Xiaofeng Meng†
Renmin University of China
xfmeng@ruc.edu.cn

ABSTRACT

Local differential privacy (LDP), a common framework for collecting private data, has been criticized for its low data utility. To address this, the shuffle model, which introduces a shuffler between users and an analyzer to amplify the privacy guarantee, has been proposed. However, this model is even more vulnerable to poisoning attacks from Byzantine users, who can collude with each other and mix with honest users anonymously through shuffling. As such, conventional malicious user detection techniques cannot work. In this paper, we investigate histogram estimation in the shuffle model, where a set of colluding Byzantine users may launch data poisoning attacks. To enhance the system's robustness against poisoning attacks, we propose a Byzantine-robust shuffle model (BRSM), where the shuffler generates noises to remedy privacy damage, and the analyzer sanitizes the poisoned histogram to mitigate utility damage. Furthermore, by exploiting the fact that histograms are typically smooth, we propose two malicious bin detection and rectification protocols, namely, MDR and MDR*, to further improve the accuracy of histogram estimation. Extensive experiments demonstrate the effectiveness of our defense methods, which suppress the Byzantine attacks in a moderate degree.

PVLDB Reference Format:
Leixia Wang, Qingqing Ye, Haibo Hu, Kai Huang, and Xiaofeng Meng. Byzantine-robust Histogram Estimation in the Shuffle Model. PVLDB, 14(1): XXX-XXX, 2020.
doi:XX.XX/XXX.XX

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at URL_TO_YOUR_ARTIFACTS.

1 INTRODUCTION

Differential privacy (DP) has become a *de facto* privacy standard to protect users' privacy by enterprises or institutions. In particular, local differential privacy (LDP), in which users perturb their data locally before data collection, has been deployed by many IT giants, such as Google[22], Apple[45], and Microsoft[16]. Although it provides guaranteed privacy protection, LDP suffers from low data utility because of heavy data perturbation that overshadows the original data.

To enhance the utility, the shuffle model, also known as the Encode-Shuffle-Analyze framework, has been proposed [5]. This model introduces a shuffler between users and an analyzer to securely shuffle users' perturbed messages, amplifying privacy guarantee against the analyzer. The privacy amplification property allows users to reduce the perturbation for the same level of privacy. Thanks to its good balance between privacy and utility, and flexibility without a trusted curator, the shuffle model has sparked widespread interest.

Histogram estimation, which estimates the frequencies of a list of discretized values, is a fundamental estimation problem and has wide applications. For example, it can be used for demography, e.g., the population's age and income distribution, and for streaming data analysis, e.g., road traffic flow monitoring over 24 hours. However, histogram estimation has not been explored in the shuffle model. Fortunately, frequency estimation, which can act as a building block of histogram estimation, has been extensively investigated on its perturbation mechanisms and privacy amplification effects [1–3, 11, 12, 20, 25, 50]. Nonetheless, applying frequency estimation protocols for histogram is non-trivial, as the latter has unique characteristics, such as finer granularity (i.e., a large number of bins) and strong correlation between adjacent bins.

In addition, most existing studies on the shuffle model assume that all users are honest, which is not practical in real-world applications where some users may intentionally provide false data due to device compromise, dishonesty, and unwillingness to share data. Such malicious behavior is commonly known as a **poisoning attack** [7, 10, 32, 34, 37, 53]. The malicious users, who may further collude with each other, are often referred to as **Byzantine users.** Unfortunately, due to the following reasons, the shuffle model is particularly vulnerable to poisoning attacks from Byzantine users.

- Privacy damage: Since Byzantine users do not follow the given perturbation protocol, the randomness they should contribute is thus lost, which undermines the guaranteed desired privacy that relies on such randomness contributed by all users.
- Utility damage: Byzantine users can deliberately craft their output messages to inject poisoning data, resulting in malicious results. For example, they can manipulate their output to affect the statistical results of traffic flow, falsely indicating that a road is congested instead of clear. Or they

*Work partially done while at Hong Kong Polytechnic University.
†Corresponding author: Xiaofeng Meng.

Proceedings of the VLDB Endowment, Vol. 14, No. 1 ISSN 2150-8097.
doi:XX.XX/XXX.XX

can distort the actual income demographics by injecting extreme income data.
- Hard to detect: Byzantine users can easily masquerade as honest users because of the local perturbation and shuffling process. Since their outputs are mixed with honest users' perturbed outputs in an anonymous manner, it is challenging to distinguish malicious users from benign ones.

To our knowledge, Balcer et al. [2] is the only work that considers the impact of malicious users in the shuffle model. However, only the impact on privacy is addressed while neglecting the utility damage caused.

In this paper, we investigate histogram estimation in the shuffle model with the existence of a set of colluding Byzantine users launching poisoning attacks. We assume smooth histograms [36, 42], where the frequencies of adjacent bins do not vary much [43]. As for the threat model, we assume the Byzantine users can pry on the ordinary local perturbation procedure to disguise their poisoning outputs as randomized outputs. Furthermore, they are aware of the smoothness requirement of a histogram, and attempt to strike a balance between the attack goal and detection evasion.[1]

As a mitigation to the utility damage, we propose a Byzantine-robust shuffle model (BRSM) to enhance the system's robustness against poisons by leveraging the smoothness constraints of histograms. The shuffler here is responsible for guaranteeing the privacy, and it adds perturbed uniform noises before shuffling to replenish the damaged randomness caused by Byzantine users. The analyzer calibrates the bias from (unknown) honest users and the shuffler, and rectifies the poisoned histogram towards the original one to enhance the utility. In particular, based on the non-smoothness characteristics exhibited from poisoned values, we propose a malicious bin detection and rectification (MDR) protocol for the analyzer, and enhance it to MDR* protocol that can handle arbitrary degree of smoothness of the original histogram and privacy budget.

We demonstrate the effectiveness of our proposed defenses through extensive experiments and reach the conclusion that: *the utility after defense improves as the severity of the attack increases. The more severe the destruction of the utility an attack causes, the more apparent the non-smoothness characteristics exposed, and thus the higher degree of rectification achieved by our defense.* It suggests that we have achieved our goal to suppress the Byzantine attacks in a moderate degree.

In summary, our contributions are as follows:

- We are the first to study Byzantine users' poisoning attacks on histogram estimation in the shuffle model and analyze its impact on privacy and utility. We propose three attacks and theoretically model the trade-off between damaging utility and retaining histogram smoothness.
- We propose a BRSM framework to defend against these attacks while preserving privacy and mitigating utility damage. Within this framework, we present two protocols MDR and MDR*, which rectify the poisoned histogram and enhance the estimation accuracy.
- We perform extensive experiments on both synthetic and real-world datasets to demonstrate the effectiveness of our defense methods and reveal an intriguing relationship between attacks and defenses.

The remainder of this paper is organized as follows. Section 2 provides preliminaries and formally defines the problem. Section 3 introduces the threat model and three attacks and presents the trade-off between utility damage and histogram smoothness maintenance. Section 4 overviews the BRSM framework, while Section 5 and Section 6 elaborate on MDR and MDR*, respectively. Section 7 empirically evaluates the defense methods. Finally, we review related works in Section 8 and conclude this paper in Section 9.

[1] In other words, the attackers can go beyond the attack (e.g., maximal gain attack (MGA)[7]) that can severely undermine the estimation accuracy but is too easy to detect since it breaks the smoothness requirement of a histogram.

2 PRELIMINARIES AND PROBLEM DEFINITION

Table 1: Notations

Symbols	Description
N	the total number of users
α	the proportion of Byzantine users
α_{u}	the upper bound of the proportion of Byzantine users
m	the number of shuffler-padded noises
$[d]$	the public domain, $[d] = \{1, 2, \ldots, d\}$
$\mathbf{F}$	the actual histogram, $\mathbf{F} = (f_1, f_2, \ldots, f_d)$
$\hat{\mathbf{F}}$	the estimated histogram, $\hat{\mathbf{F}} = (\hat{f}_1, \hat{f}_2, \ldots, \hat{f}_d)$
$\bar{\mathbf{F}}$	the MDR-rectified histogram, $\bar{\mathbf{F}} = (\bar{f}_1, \bar{f}_2, \ldots, \bar{f}_d)$
$\tilde{\mathbf{F}}$	the MDR*-optimized histogram, $\tilde{\mathbf{F}} = (\tilde{f}_1, \tilde{f}_2, \ldots, \tilde{f}_d)$
$\mathcal{S}$	the function of simple moving average
$\mathbf{\Gamma}$	Byzantine users output $j \in [d]$ with probability $\gamma_j \in \mathbf{\Gamma}$
T_ξ	the threshold for identifying the malicious bins in MDR

2.1 Differential Privacy (DP)

DP, also known as central differential privacy [11], bounds the privacy of a randomization mechanism $\mathcal{M}$ on two neighboring databases $\mathbf{X} \simeq \mathbf{X}'$ with one record differing, i.e.,$x_i \neq x'_i, i \leq N$. It relies on a trusted curator to collect the private database and perform randomization. One pivotal property of DP is *post-processing invariance*, which ensures that any randomized post-processing on $\mathcal{M}$'s outputs still preserves (ϵ, δ)-DP.

Definition 1 (Differential Privacy [19]). *A randomization algorithm $\mathcal{M} : \mathbb{X}^N \to \mathbb{Z}$ is (ϵ, δ)-DP, where $\epsilon, \delta > 0$, iff for any two neighboring databases $\mathbf{X} \simeq \mathbf{X}' \in \mathbb{X}^N$ and any output $z \in \mathbb{Z}$,*

$$\Pr[\mathcal{M}(\mathbf{X}) = z] \leq e^{\epsilon} \Pr[\mathcal{M}(\mathbf{X}') = z] + \delta \tag{1}$$

2.2 Local Differential Privacy (LDP)

LDP allows users to perturb their data before reporting them to a central curator. This definition eliminates the need for a trusted curator and is widely used in industry [14, 16, 22].

Definition 2 (Local Differential Privacy [18]). *A randomization algorithm $\mathcal{R} : \mathbb{X} \to \mathbb{Y}$ is ϵ_l-LDP, where $\epsilon_l > 0$, iff for any pair of input records $x, x' \in \mathbb{X}$ and any output $y \in \mathbb{Y}$,*

$$\Pr[\mathcal{R}(x) = y] \leq e^{\epsilon_l} \Pr[\mathcal{R}(x') = y] \tag{2}$$

We review two typical perturbation mechanisms of LDP, which are also widely used in the shuffle model [3, 4, 20, 50].

Generalized Randomized Response (GRR) [48] is the generalization of randomized response [51]. In this mechanism, each user perturbs his value using Eq.(3), where $p = e^{\epsilon_l}/(e^{\epsilon_l} + d - 1)$, $q = 1/(e^{\epsilon_l} + d - 1)$, and $d = |\mathbb{X}|$ is the domain size. Because $p/q = e^{\epsilon_l}$, it satisfies ϵ_l-LDP. The analyzer collects these perturbed values and estimates the frequency $\hat{f}_x, x \in \mathbb{X}$ using Eq.(4), where n'_x is the collected count of x.

$$\Pr(\mathcal{R}(x) = y) = \begin{cases} p & \text{if } y = x \\ q & \text{if } y \neq x \end{cases} \tag{3}$$

$$\forall x \in \mathbb{X}, \quad \hat{f}_x = (n'_x - Nq)/(N(p - q)) \tag{4}$$

Unary Encoding (UE) [48] allows each user to encode his value x to a d-length bit vector $\mathbf{b}$, where only the x-th bit is 1, and the others are all 0s. The user perturbs each bit $b_j \in \{0, 1\}$, $j \leq d$ in $\mathbf{b}$ using Eq.(3) with $p = e^{\epsilon_l/2}/(e^{\epsilon_l/2} + 1)$ and $q = 1/(e^{\epsilon_l/2} + 1)$. Here, $p/q = e^{\epsilon_l/2}$. Because any change from x to x' will change at most two bits in $\mathbf{b}$, the total consumed privacy budget is ϵ_l. Then, the analyzer collects all these perturbed vectors, counts n'_x with $b_x = 1$, and estimates the frequency $\hat{f}_x$ using Eq.(4).

2.3 The Shuffle Model

Although LDP improves privacy more than DP, its error is $O(\sqrt{N})$ times that of DP because heavier perturbation arises in clients [8]. The shuffle model balances privacy and utility by adding a shuffling step over LDP, including the following parties:

- **Honest Users**: They follow the protocol's instructions to perturb their data honestly (as in LDP) and are willing to provide their data to the analyzer given an acceptable privacy guarantee.
- **Semi-honest Shuffler**: It collects values from all users, securely shuffles them without knowing concrete private values, and then transfers these anonymized and mixed data to the analyzer.
- **Untrusted Analyzer**: It receives all these shuffled values and strives to estimate accurate statistical results.

The core benefit of the shuffle model is privacy amplification. It ensures that users perturbing their data with ϵ_l-LDP can gain an (ϵ, δ)-DP guarantee against the analyzer, where $\epsilon < \epsilon_l$. This benefit relies on the perturbed value shuffling of all users. Next, we review the state-of-the-art privacy amplification theorem.

Theorem 1 (General Privacy Amplification[23]). *Given n users, if each user perturbs his value locally with ϵ_l-LDP, then for any $\delta \in (0, 1]$ and $\epsilon_l \leq \ln(\frac{n}{16\ln(2/\delta)})$, the shuffled messages satisfy (ϵ, δ)-DP, where*

$$\epsilon \leq \ln\left(1 + \frac{e^{\epsilon_l} - 1}{e^{\epsilon_l} + 1}\left(\frac{8\sqrt{e^{\epsilon_l}\ln(4/\delta)}}{\sqrt{n}} + \frac{8e^{\epsilon_l}}{n}\right)\right). \tag{5}$$

In this paper, we assume that honest users locally perturb their data by GRR or UE mechanism with a privacy budget ϵ_l, and the analyzer estimate each item's frequency by Eq.(4) with such an ϵ_l.

2.4 Smooth Histogram

The histogram, which consists of a list of discretized values along with their frequencies, is a discrete approximation of a continuous distribution. Therefore, we define the smoothness of a histogram by extending that from continuous numerical function[52]. Concretely, we use the forward difference to approximate the derivative [33], representing the slope of the line connecting the frequencies of two adjacent values j and $j + 1$. A histogram is smooth if the forward differences are consistent and bounded. Formally, we define ξ-smoothness as follows.

Definition 3 (ξ-smooth). *A histogram in domain $[d]$ is ξ-smooth if for any adjacent values j and $j + 1$, the absolute forward difference of their frequencies is bounded by ξ, i.e., $\left|\frac{f_j - f_{j+1}}{j-(j+1)}\right| = |f_j - f_{j+1}| \leq \xi$.*

We introduce a popular smoothing technique *simple moving average* (SMA) in Definition 4, which can smooth the histogram $\mathbf{F}$ to remove random noises. The smoothing step s determines the smoothing degree, where the larger the s, the smoother the histogram.

Definition 4 (Simple Moving Average (SMA)[29]). *Given domain $[d]$ and the smoothing step s, the SMA of f_j is $\mathcal{S}(f_j) = \frac{1}{2s+1}\sum_{z=j-s}^{j+s} f_z$ with $f_z = f_1$ if $j < 1$ and $f_z = f_d$ if $j > d$.*

2.5 Problem Definition

Without loss of generality, we assume that there are N users, with a proportion of α but no more than α_{u} Byzantine users, as shown in Fig. 1. Byzantine users launch attacks without knowing the original histogram, and the analyzer defends against attacks knowing the upper bound α_{u} of the Byzantine user proportion. Typically, we can assume that $\alpha_{\mathrm{u}} = 50\%$ without any background knowledge, indicating they will be no more than half of Byzantine users.

This paper focuses on histogram estimation in the shuffle model with the existence of a set of Byzantine users who can launch poisoning attacks. Given a public domain $[d] = \{1, 2, \ldots, d\}$ and each honest user having a discretized value $j \in [d]$, we use $\mathbf{F} = (f_1, f_2, \ldots, f_d)$ to denote the actual histogram of all honest users' data, and use $\hat{\mathbf{F}}$ to denote the estimated histogram on the analyzer side. Here, each bin corresponds to a value, and we use the terms "bin" and "value" interchangeably in the following text. We aim to devise a framework and corresponding methods to estimate a histogram over domain $[d]$ as accurately as possible while satisfying (ϵ, δ)-DP.

3 THREAT MODEL

In this section, we analyze the capabilities and goals of Byzantine users and reveal a trade-off between utility damage and smoothness maintenance.

3.1 Attackers' Capabilities

The *Byzantine users* in the shuffle model know the implementation of clients' encoding and perturbation steps and have no idea about the actual histogram. They can collude with each other and send any message at will, undermining the aggregated result.

We present the attacked shuffle model in Fig. 1 and formalize the behavior of Byzantine users. Without loss of generality, we assume that Byzantine users output each value $j \in [d]$ with probability γ_j, where $\mathbf{\Gamma} = \{\gamma_1, \gamma_2, \ldots, \gamma_d\}$ can be negotiated in advance by collusion. The malicious outputs are designed to have the same appearance and statistical characteristics as honest users' outputs to confuse the analyzer. *The same appearance* is achieved by picking values from

the encoded domain. *The same statistical characteristic* ensures that the probability distribution of the output satisfies $\sum_{j\in[d]}\gamma_j = C$ as in honest users' outputs, where C is a constant depending on different local randomizers. For example, if the randomizer is GRR, each Byzantine user outputs a value $j \in [d]$ with probability γ_j and ensures that $\sum_{j\in[d]}\gamma_j = 1$. However, if the randomizer is UE, each Byzantine user outputs a d-length bit vector $\mathbf{y}$ with each bit $y_j \sim Bernoulli(\gamma_j)$ and ensures that $\sum_{j\in[d]}\gamma_j = p + (d-1)q = (e^{\epsilon_l/2}+d-1)/(e^{\epsilon_l/2}+1)$, making the expected count of bit 1 in the output vector similar to that in the honest user's output.

By elaborate disguises of Byzantine users, the analyzer can no longer distinguish the outputs of Byzantine users from those of honest users in the shuffled messages. Byzantine users can then freely launch poisoning attacks against our systems, causing two kinds of damage:

- **Privacy damage**: By taking advantage of randomness in all perturbed outputs and shuffling, users can gain amplified privacy against the analyzer when there are no Byzantine users, resulting in a tighter ϵ that is inversely proportional to $\sqrt{N}$ (according to Theorem 1 with $n = N$). However, because Byzantine users will not follow the perturbation mechanism to contribute randomness, the privacy guarantee ϵ will be inversely proportional to $\sqrt{N(1-\alpha_u)}$, indicating a significant privacy loss.
- **Utility damage**: By injecting malicious values, Byzantine users will undermine the estimated frequency as in Eq.(6), leading to a significant bias as in Eq.(7).

$$\mathbb{E}(\hat{f}_j) = (1-\alpha)f_j + \alpha(\gamma_j - q)/(p-q) \tag{6}$$

$$Bias(\hat{f}_j) = \mathbb{E}(\hat{f}_j) - f_j = \alpha\left((\gamma_j - q)/(p-q) - f_j\right) \tag{7}$$

3.2 Attackers' Goals and Attacks

In practice, Byzantine users have various goals. "Naughty" Byzantine users with malicious hacker or business rival identities aim to destroy the estimated results. Byzantine users with speculator identities want to promote or suppress specific items to increase their gains. Attacks with the above goals will undermine the utility of estimated results. However, some cautious Byzantine users may attempt to maintain the original statistics of the histogram (e.g., smoothness) to ensure perfect stealthiness. These users could be extremely privacy-conscious individuals trying to participate in the system covertly in order to enjoy free services or rewards without disclosing any private information.

In this subsection, we propose two extreme attacks: **Maximal Loss Attack (MLA)** and **Absolute Smooth Attack (ASA)**, and a middle-ground: **Random Distribution Attack (RDA)**.

3.2.1 Maximal Loss Attack (MLA). We propose MLA to maximize the *utility loss* of the poisoned histogram. We define the *utility loss* as the square error of the estimated histogram, i.e., $\mathcal{L} = \|\hat{\mathbf{F}} - \mathbf{F}\|^2$. Given p, q, and C for specific randomizers, let $C_U = \sum_{j\in[d]} q^2/(p-q)^2 - 2qC/(p-q)^2 + q/(p-q)$. Then, we have

$$\mathbb{E}[\mathcal{L}] = \alpha^2\left(\sum_{j\in[d]}\frac{\gamma_j^2}{(p-q)^2} + \sum_{j\in[d]} f_j^2 - \sum_{j\in[d]}\frac{f_j\gamma_j}{p-q} + C_U\right). \tag{8}$$

To maximize the utility loss in Eq.(8), Byzantine users need to set appropriate $\mathbf{\Gamma}$. As α, p, q, d, and C are constants, and each f_j only relates to real histogram distribution, we can know that

$$\arg\max_{\mathbf{\Gamma}} \mathbb{E}[\mathcal{L}] = \arg\max_{\mathbf{\Gamma}}\left(\sum_{j\in[d]}\frac{\gamma_j^2}{(p-q)^2} - \sum_{j\in[d]}\frac{f_j\gamma_j}{p-q}\right) \tag{9}$$

Since $0 < p-q < 1$, the first item $\sum_{j\in[d]}\gamma_j^2/(p-q)^2$ dominates Eq.(9). Therefore, Byzantine users can alternatively optimize Eq.(10) while subject to the constraints on Γ. We exhibit MLA in Theorem 2.

$$\arg\max_{\mathbf{\Gamma}} \mathbb{E}[\mathcal{L}] \approx \arg\max_{\mathbf{\Gamma}} \sum_{j\in[d]}\frac{\gamma_j^2}{(p-q)^2} = \arg\max_{\Gamma}\sum_{j\in[d]}\gamma_j^2 \tag{10}$$
$$s.t. \quad \sum_{j\in[d]}\gamma_j = C, \quad \forall j, 0 \le \gamma_j \le 1$$

Theorem 2 (Maximal Loss Attack (MLA)). *Given $C=1$ for GRR and $C = (e^{\epsilon_l/2}+d-1)/(e^{\epsilon_l/2}+1)$ for UE, Byzantine users negotiate $\mathbf{\Gamma}$ in advance and randomly set $\lfloor C\rfloor$ 1s, $d-\lceil C\rceil$ 0s, and one $\gamma_j = C - \lfloor C\rfloor$ in $\mathbf{\Gamma}$ if C is not an integer. Following $\mathbf{\Gamma}$, they disguise their outputs as benign outputs of GRR or UE and then report them to the shuffler.*

3.2.2 Absolute Smooth Attack (ASA). We propose ASA to keep the poisoned histogram ξ-smooth. First, we formalize the smoothness of the attacked histogram as the maximal absolute difference of adjacent frequencies, i.e., $\Delta_{\max}$ as in Eq.(11). According to Definition 3, the ξ-smoothness of the poisoned histogram is maintained if $\Delta_{\max} \le \xi$. Since for any j, $|f_j - f_{j+1}| \le \xi$, when $\max_{j\in[d]}|\gamma_j - \gamma_{j+1}| \le \xi(p-q)$, this smooth condition holds.

$$\begin{aligned}\Delta_{\max} &= \max_{j\in[d]}\left|\mathbb{E}(\hat{f}_j) - \mathbb{E}(\hat{f}_{j+1})\right| \\ &= \max_{j\in[d]}\left|(1-\alpha)(f_j - f_{j+1}) + \tfrac{\alpha}{p-q}(\gamma_j - \gamma_{j+1})\right|\end{aligned} \tag{11}$$

However, because Byzantine users have no idea about the actual smoothness ξ, it is difficult to design $\mathbf{\Gamma}$ to meet this condition exactly. Instead, they can minimize $\max_{j\in[d]}|\gamma_j - \gamma_{j+1}|$ as much as possible to ensure that $\max_{j\in[d]}|\gamma_j - \gamma_{j+1}| \le \xi(p-q)$. By solving Eq.(12), we derive ASA in Theorem 3.

$$\min\max_{j\in[d]}|\gamma_j - \gamma_{j+1}| \quad s.t. \sum_{j\in[d]}\gamma_j = C, \quad \forall j, 0 \le \gamma_j \le 1 \tag{12}$$

Theorem 3 (Absolute Smooth Attack (ASA)). *Given $C=1$ for GRR and $C = (e^{\epsilon_l/2}+d-1)/(e^{\epsilon_l/2}+1)$ for UE, Byzantine users set $\mathbf{\Gamma} = \{C/d\}^d$, that is $\forall j \in [d]$, $\gamma_j = C/d$. Then, they generate GRR-liked or UE-liked messages with probability distribution $\mathbf{\Gamma}$.*

3.2.3 Random Distribution Attack (RDA). Beyond careful planning, Byzantine users may adopt a middle-ground strategy by randomly generating their output distribution $\mathbf{\Gamma}$. The two extreme attacks MLA and ASA, can be special cases of RDA, since their required $\mathbf{\Gamma}$ can be generated in RDA with a certain probability. Specifically, in RDA, each γ_j for $j \in [d]$ follows $Bernoulli(p_j)$, where p_j is a random variable. To preserve their stealthiness, they also should normalize Γ to be subject to constraints of appearance and statistical characteristics described in Section 3.1.

3.3 Trade-off between Utility Damage and Smoothness Maintenance

Both regular perturbation and the utility-damaging attack may break the smoothness of a histogram. The influence of perturbation

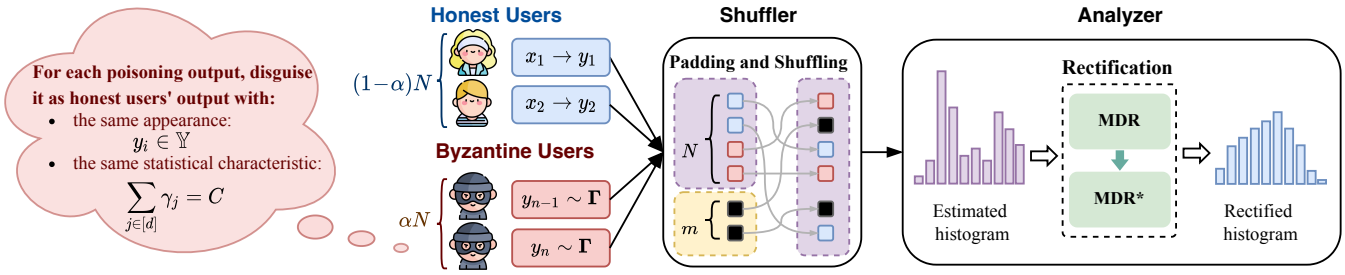

Figure 1: Byzantine-robust Shuffle Model (BRSM)

on smoothness is significantly smaller than that of attacks because privacy amplification of shuffling allows user to perturb their value with a giant ϵ_l. We then discover a trade-off between utility damage and the smoothness of the poisoned histogram.

- **Pursuing the maximal utility loss will break smoothness.** In MLA, when C is an integer, $\max_{j\in[d]} |\gamma_j - \gamma_{j+1}| = 1$; otherwise, $\max_{j\in[d]} |\gamma_j - \gamma_{j+1}| = \max\{C - \lfloor C \rfloor, 1 - C + \lfloor C \rfloor\} \geq 1/2$. Then, if $\alpha/2 \geq \xi$, then $\Delta_{\max} \geq \alpha \max_{j\in[d]} |\gamma_j - \gamma_{j+1}|/(p-q) \geq \xi$, the smoothness will be broken. This condition can be easily met with enough Byzantine users because ξ is usually a negligible value.
- **Keeping the absolute smoothness will minimize utility damage.** When Byzantine users launch ASA, the resulting utility loss is exactly the minimum value of Eq.(10).

Motivated by this trade-off, we can detect malicious bins by checking their non-smoothness. It is an open problem for Byzantine users to either strengthen utility damage or maintain smooth results, depending on their goals and cautiousness.

4 BRSM: DEFENSE FRAMEWORK

In this section, we propose the Byzantine-robust shuffle model (BRSM), as shown in Fig. 1, to defend against poisoning attacks by Byzantine users. It consists of two core components, i.e., **the privacy defense** by the shuffler and **the utility defense** by the analyzer. We describe these two defense strategies in the following, then depict the overall model at length.

4.1 Privacy Defense by Shuffler

In the shuffle model, the privacy guarantee (including the privacy amplification effect) depends on the randomness contributed by the shuffler and all users. However, as Byzantine users may not follow a given protocol faithfully, the overall randomness will be thus destroyed. To compensate for the randomness of at most $N\alpha_u$ Byzantine users, we delegate the shuffler to add m pieces of random noise to users' outputs before shuffling, so that we can preserve amplified privacy and defend against privacy undermining. In order to thoroughly mix the shuffler-padded noises with users' outputs, the shuffler also perturbs each uniform noise following the local perturbation mechanism before padding. The following Theorem 4 (with the proof in Appendix A.2 in the full version [46]) shows how to select an appropriate m to defend against the privacy damage. The twofold randomness introduced by uniform noises and perturbation makes m significantly less than $N\alpha_u$, the maximal number of Byzantine users.

Theorem 4 (Privacy Guarantee). *When $m = N\alpha_u \frac{d}{2e^{\epsilon_l}+d-2}$, Theorem 1 holds with $n = N$, and the shuffled messages satisfy (ϵ,δ)- DP.*

To eliminate these additional noises' influence, the shuffler sends the value m together with all shuffled messages to the analyzer. Because the value of m only relies on the upper bound of the Byzantine user proportion and not other private information, the exposure of m will not leak any additional privacy. With all the information, the analyzer can estimate each bin's frequency by Eq.(13), where n_j denotes the noisy count of j.

$$\hat{f}_j = \left(n_j - (N+m)q\right)/N(p-q) - m/Nd \tag{13}$$

This equation calibrates the bias introduced by both the shuffler's noises and honest users' perturbation. We demonstrate the utility of this estimation in Theorem 5 with the proof presented in Appendix A.3 in [46]. This calibrated result will be unbiased when Byzantine users abandon poisoning attacks to comply with perturbation protocols as honest users do.

Theorem 5 (Utility Guarantee). *The expectation and the variance of the calibrated frequency by Eq.(13) are presented in Eq.(14) and Eq.(15), respectively, where p and q are specified for different randomizers in Section 2.2.*

$$\mathbb{E}(\hat{f}_j) = (1-\alpha)f_j + \alpha(\gamma_j - q)/(p-q) \tag{14}$$

$$\begin{aligned}\mathrm{Var}(\hat{f}_j) &= (N(1-\alpha)f_j + m/d)(1-p-q)/N^2(p-q) \\ &+(N(1-\alpha)+m)(1-q)q/N^2(p-q)^2 + \alpha\gamma_j(1-\gamma_j)/N(p-q)^2\end{aligned} \tag{15}$$

4.2 Utility Defense by Analyzer

We deliver the primary mission of utility defense to the analyzer. Intuitively, a poisoned histogram will exhibit different characteristics from a benign histogram, which can be used to detect and rectify the poisoned histogram, to mitigate the utility damage. An actual histogram meets *three constraints*: **(1) the histogram is smooth, (2) the sum of frequencies is 1, and (3) the range of each frequency is** $[0, 1]$**.** However, all these constraints are vulnerable to poisoning attacks, especially the smoothness constraint, which contradicts utility damage. Therefore, we can detect malicious bins by examining their deviation from the constraints and mitigate their impact by rectifying them to satisfy the three constraints.

Utilizing this basic idea, we propose a baseline method, MDR, for detecting non-smoothed and malicious bins and rectifying their frequencies using nearby benign bins. However, because the real smoothness value ξ is unknown, MDR manually sets a detection threshold to determine which level of non-smoothness is malicious.

MDR with this threshold is simple and effective for most cases but is not general for those with inconstant smoothness. To tackle this, an optimization protocol, MDR*, is designed to pursue a more accurate result by avoiding under-smoothed or over-smoothed rectified histograms and focusing on truth-close histograms. We elaborate on these two methods in Sections 5 and 6.

4.3 Byzantine-robust Shuffle Model (BRSM)

An overview of BRSM is presented in Fig. 1. Unlike the traditional shuffle model, some malicious users sneak into BRSM and submit the poisoning data while disguising their outputs. The shuffler, responsible for the privacy, not only shuffles but also adds m perturbed uniform random noises to messages received from all users before shuffling them. The analyzer estimates the histogram from these shuffled messages and takes charge of mitigating utility damage introduced by Byzantine users. It uses either MDR or its optimization MDR* to rectify the poisoned histogram.

Concretely, we present the pseudocode of BRSM in Algorithm 1. Initially, the shuffler computes the number of padded noises m according to Theorem 4 (line 1). The analyzer specifies the perturbation mechanism $\mathcal{R}$ (i.e., GRR or UE) and computes ϵ_l (line 2). The ϵ_l is required for local perturbations and benefits from the privacy amplification in Theorem 1. Next, the honest users perturb their values with $\mathcal{R}$ (line 3), while Byzantine users negotiate their output distribution $\mathbf{\Gamma}$ and disguise their output as normal perturbed (line 5). After that, as we depicted in the framework, the shuffler pads noises to the received messages and shuffle them to guarantee privacy (lines 7∼9). The analyzer collects these messages and m to estimate the histogram and rectify them via MDR and MDR* (lines 12∼13). It is noteworthy that all analyzer's rectification steps preserve (ϵ, δ)-DP of results because of the post-processing invariance of DP.

Algorithm 1: Byzantine-robust Shuffle Model (BRSM)

Input: ϵ and α_{u} for analyzer; items $j \in [d]$ for honest users
Output: Rectified histogram $\tilde{\mathbf{F}}$
`// Initialization`
1 Shuffler calculates m based on Theorem 4;
2 Analyzer calculates ϵ_l based on Theorem 1 and broadcasts ϵ_l and perturbation mechanism $\mathcal{R}$ to the shuffler and all users ;
`// Honest user side`
3 Perturb his value x_i with ϵ_l to y_i, i.e., $y_i = \mathcal{R}(x_i)$;
4 Send y_i to the shuffler;
`// Byzantine user side`
5 Generate y_i with the probability $\mathbf{\Gamma}$;
6 Send y_i to the shuffler;
`// Shuffler side`
7 Generate m uniform random noises $\mathbf{R} = \{r_1, r_2, \ldots, r_m\} \in [d]^m$;
8 Perturb $\mathbf{R}$ to $\mathbf{R'} = \{\mathcal{R}(r_1), \mathcal{R}(r_2), \ldots, \mathcal{R}(r_m)\}$;
9 Securely shuffle $\mathbf{Y} \cup \mathbf{R'}$ to $\mathbf{Y'}$;
10 Send $\mathbf{Y'}$ and the number m to the analyzer;
`// Analyzer side`
11 For each $j \in [d]$, estimate $\hat{f}_j$ with Eq.(13) ;
12 MDR: Rectify the estimated $\hat{\mathbf{F}}$ to $\bar{\mathbf{F}}$;
13 MDR*: Find an optimal result $\tilde{\mathbf{F}}$ based on MDR;

5 MDR: A BASELINE PROTOCOL FOR UTILITY DEFENSE

This section introduces the MDR protocol, which utilizes smoothness to detect and eliminate the negative impact of malicious bins. We first present a solution for malicious bin detection and rebuilding, and further introduce pre-processing and post-processing steps to handle more general cases. These procedures mitigate the influence of poisoning attacks and ensure that the rectified histogram meets the constraints of a true histogram.

5.1 Malicious Bin Detection and Rebuilding

In typical scenarios, Byzantine users send values with a non-uniform probability $\mathbf{\Gamma}$ and cause varying corruption degrees on bins. For clarity, we classify the bins into *malicious* and *benign* categories. The malicious bins are heavily corrupted compared to others and can break the smoothness characteristic, requiring rectification. The benign bins, while slightly damaged, still maintain the smoothness characteristic partially, and their frequencies are close to the original ones, which can be used to rectify the malicious bins.

Basic Detection and Rebuilding: We measure the degree of smoothness of each bin and identify them as either malicious or benign. To do this, we construct a smooth histogram $\bar{\mathbf{F}}$ using SMA (described in Definition 4) on the estimated histogram $\hat{\mathbf{F}}$. If the frequency $\hat{f}_j$ of bin $j \in [d]$ deviate significantly from its smoothed value, i.e., $|\hat{f}_j - \bar{f}_j| \geq T_{\xi}$, we label j as malicious; otherwise, we label j as benign. T_{ξ} is a critical threshold that determines the degree of malicious bin detection, and we set it in the next subsection. Furthermore, we set the smoothing parameter $s = 1$ in SMA to avoid over-smoothing the unknown histogram.

To mitigate the negative effect of malicious bins, we discard their heavily corrupted frequencies $\hat{f}_j$ and replace them with the average frequency of their nearest benign neighbors, i.e., $(\hat{f}_l + \hat{f}_r)/2$, where l and r denote the left and right nearest benign bins to bin j, respectively. For an edge bin without a left (resp. right) benign neighbor, we assume that $\hat{f}_l$ (resp. $\hat{f}_r$) is equal to the frequency of the nearest right (resp. left) benign bin of this edge bin. Because the original histogram is ξ-smooth, when ξ is negligibly small, the frequencies of the adjacent bins should be very close, and this average can approximate $\hat{f}_j$ well. We show an example in the steps ①∼② of Fig. 2 (a), and depict the procedure in lines 3∼5 of Algorithm 2, where I_j is a label, and $I_j = 0$ indicates bin j is malicious, and $I_j = 1$ otherwise.

Two Types of Errors: However, since the smoothed histogram is constructed based on the poisoned histogram $\hat{\mathbf{F}}$, which may not accurately reflect the true distribution, the bin j next to a malicious bin may be misidentified, resulting in two types of errors. We refer to the benign label as a positive result and the malicious label as a negative result. The *false negative error* occurs when a benign bin j is misidentified as malicious, causing its frequency to be replaced with the average with more bias. The *false positive error* occurs when a malicious bin j is misidentified as benign, causing the heavily poisoned frequency to be retained.

An Iterative Mechanism: To address these errors, we perform detection and rebuilding iteratively. The intuition behind it is that, even with some errors, some malicious bins can be significantly

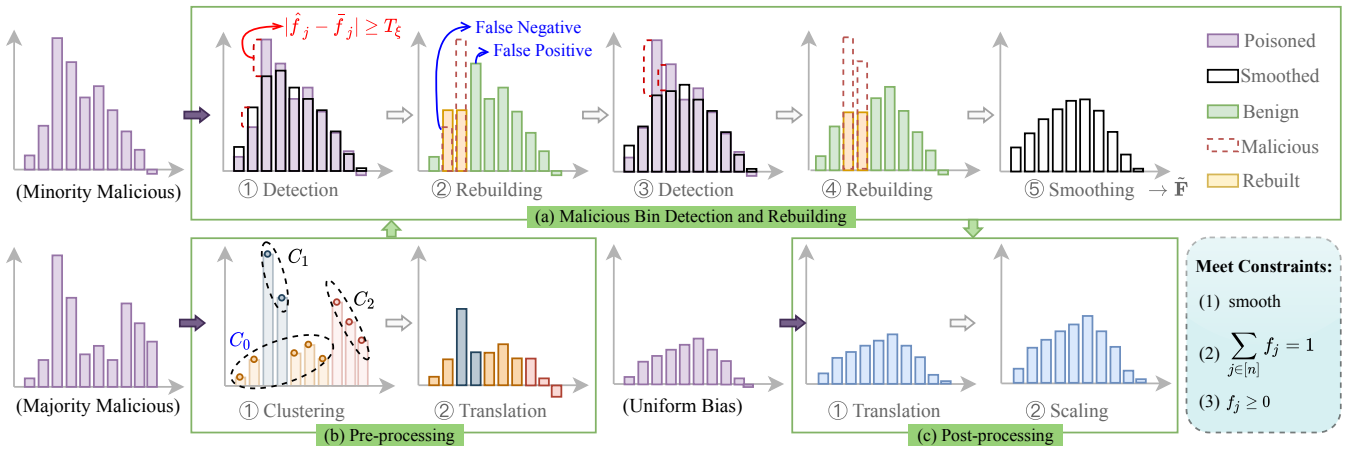

Figure 2: An Example of MDR

Algorithm 2: Detection and Rebuilding (DR)

Input: the estimated histogram $\hat{\mathbf{F}}$, the threshold T_ξ, T_N
Output: the rectified histogram $\bar{\mathbf{F}} = \{\bar{f}_1, \bar{f}_2, \ldots, \bar{f}_d\}$

Let labels $\mathbf{I}^{(0)} \in \{1\}^d$, $\mathbf{I}^{(1)} = \phi$, $\bar{\mathbf{F}} = \hat{\mathbf{F}}$ and $t_1 = 1$;
while $\mathbf{I}^{(t_1-1)} \neq \mathbf{I}^{(t_1)}$ *or* $t_1 \leq T_\mathrm{N}$ **do**
  Compute smoothed histogram $\bar{\mathbf{F}} \leftarrow S(\bar{\mathbf{F}})$;
  $I_j^{(t_1)} = 0$ if $|\hat{f}_j - \bar{f}_j| \geq T_\xi$, otherwise $I_j^{(t_1)} = 1$;
  Rebuild $\bar{f}_j$ with label $I_j^{(t_1-1)} = 0$ to $(\hat{f}_l + \hat{f}_r)/2$;
  $t_1 = t_1 + 1$;

rectified, leading to a more accurate rebuilt histogram. Based on this rebuilt histogram, we can derive a more accurate smooth histogram via SMA, enabling us to identify the malicious bins more accurately. Better detection and rebuilding can mutually promote each other, and thus some misidentified bins can be corrected during the iteration, as exhibited in Fig. 2(a). We continue the iteration until the labels $\mathbf{I}$ of bins are no longer changed or the maximum number of iterations T_N is reached. Finally, we smooth the last rebuilt histogram to obtain $\bar{\mathbf{F}}$ as the output.

We show this iterative procedure in Algorithm 2, and analyze the converge conditions in Theorem 6 with proof in Appendix A.4 in the full version [46]. The theorem shows that the output will converge to an accurate result with all bins labeled correctly when the number of malicious bins is less than $\lfloor (d-1)/3 \rfloor$.

Theorem 6. *Let χ_j, $|\chi|^b_{\max}$, $|\chi|^c_{\min}$, and n_c denote the noise added by perturbation and an attack, the noise with maximal absolute value for benign bins, the noise with minimal absolute value for malicious bins, and the number of malicious bins, respectively. When both conditions $11|\chi|^b_{\max} + d\xi \leq |\chi|^c_{\min}$, $n_c \leq \lfloor \frac{d-1}{3} \rfloor$ and $2|\chi|^b_{\max} + \frac{d}{6}\xi \leq T_\xi \leq \frac{1}{6}|\chi|^c_{\min} - \frac{1}{2}|\chi|^b_{\max} - \frac{d}{18}\xi$ hold, all bins will be labeled correctly after iterations and converge to an accurate histogram.*

5.2 The Choice of T_ξ

In MDR, T_ξ is a critical parameter for the precision of malicious bin detection and the smoothness of the rectified histogram. When T_ξ comes to smaller, MDR is more likely to identify more malicious bins, leading to a smoother rectified histogram and vice versa. We aim to find an appropriate value of T_ξ to filter all malicious bins with un-smoothness caused by the poisoning attack as much as possible. The lower bound of T_ξ revealed in Theorem 6 achieves this goal. However, without knowing the corresponding parameters $|\chi|^b_{\max}$, $|\chi|^c_{\min}$, and ξ, we cannot compute this value directly.

Alternatively, we propose a simple but effective approximation for it. We make two critical assumptions: the true histogram is sufficiently smooth ($\xi \to 0$), and the identified benign bins rarely have biases ($\psi_j^B \to 0$). Let each bin's noise $\chi_j = \psi_j^B + \psi_j^P$, where ψ_j^B is the noise from Byzantine users' input, and ψ_j^P is the noise from honest users and the shuffler's perturbation. Thus, we can simplify the range of T_ξ revealed in Theorem 6 as $T_\xi \geq 2|\psi^P|_{\max}$, where $|\psi^P|_{\max}$ represents the maximal value of all $|\psi_j^P|$. Prior researches [31, 49] have shown that ψ_j^P follows a Gaussian distribution $\mathcal{N}(0, \sigma^2)$ with $\sigma^2 = \mathrm{Var}(\hat{f}_j - f_j) \approx q(1-q)/n(p-q)^2$, where p and q are analyzed in Section 2.2. Therefore, we can approximate $|\psi^P|_{\max} = \sigma\Phi^{-1}(1-\beta/2)$ with confidence at least $1-\beta$, where Φ^{-1} is the inverse cumulative distribution function of $\mathcal{N}(0,1)$.

5.3 Pre-processing and Post-processing

To tackle the limitation of malicious bin detection and rebuilding that it works for minority malicious bins with heavily corrupted frequencies, we propose the pre- and post-processing steps to handle more general cases, with pseudocode shown in Algorithm 3.

Pre-processing: To handle cases where the majority of bins are malicious, we propose a pre-processing step. Because the un-smoothness of malicious bins comes from the different degrees of poisoning on adjacent bins, we can cluster the bins into classes, where the bins with similar poisoned biases are clustered together. Then, through aligning bins in different classes of histograms by translation, we can derive an approximate smooth histogram, eliminating the influence of the different degrees of poisoning. Specifically, we use HDBSCAN [6, 41], a hierarchical density-based clustering method. The bins are encoded as the normalized points $(j/d, \hat{f}_j/\sum \hat{f}_j)$ and clustered as $C = \{C_0, C_1, \ldots, C_K\}$. We always

Algorithm 3: MDR

Input: the estimated histogram $\hat{\mathbf{F}}$, the threshold T_ξ, T_{N}
Output: the rectified histogram $\bar{\mathbf{F}} = \{\bar{f}_1, \bar{f}_2, \ldots, \bar{f}_d\}$
// Pre-processing:Clustering and Translation
1 Let $t_0 = 1$ and $\hat{\mathbf{F}}^{(0)} = \phi, \hat{\mathbf{F}}^{(1)} = \hat{\mathbf{F}}$;
2 **while** $\hat{\mathbf{F}}^{(t_0-1)} \neq \hat{\mathbf{F}}^{(t_0)}$ **do**
3 Cluster $\hat{\mathbf{F}}^{(t_0)}$ to $C^{(t_0)} = \{C_0^{(t_0)}, C_1^{(t_0)}, \ldots, C_K^{(t_0)}\}$;
4 Set one with most values as the reference cluster $C_*^{(t_0)}$;
5 **for** $C_k^{(t_0)} \in C^{(t_0)} \backslash C_*^{(t_0)}$ **do**
6 Compute $\Delta^{(t_0)}$ between $C_k^{(t_0)}$ and $C_*^{(t_0)}$;
7 Translation: $\forall \hat{f}_j^{(t_0)} \in C_k, \hat{f}_j^{(t_0)} \leftarrow \hat{f}_j^{(t_0)} - \hat{\Delta}^{(t_0)}$;
8 $t_0 = t_0 + 1$;
// Malicious Bin Detection and Rebuilding
9 $\bar{\mathbf{F}} = \mathrm{DR}(\hat{\mathbf{F}}, T_\xi, T_{\mathrm{N}})$;
// Post-processing:Normalization
10 **if** $\bar{f}_{\min} < 0$ **then**
11 $\forall j \in [d], \bar{f}_j = \bar{f}_j - \bar{f}_{\min}$;
12 $\forall j \in [d], \bar{f}_j = \bar{f}_j / \sum_{j\in[d]} \bar{f}_j$;
13 **return** $\bar{\mathbf{F}}$;

choose the class C_* with the most points as the reference and translate the bins in other classes to align with it iteratively. The translation means we shift the bins in C_k up or down to eliminate the minimal frequency difference Δ between two adjacent bins from the C_* and C_k, respectively. Finally, we derive an approximated smooth histogram. We depict an example in Fig. 2 (b), describe the procedure in lines 3~7 of Algorithm 3 with the superscript (t_o) representing the number of iteration.

Note that the pre-processing step may not completely eliminate un-smoothness in all malicious bins, and some of them still require rectification by the malicious bin detection and rebuilding mechanism. Additionally, aligning bins to a reference class with biases may introduce uniform biases for each bin. Fortunately, these biases can be eliminated by normalization in our post-processing step.

Post-processing: The uniform biases of bins, which may be generated in the prior procedure or poisoned by Byzantine users (e.g., ASA), can be seen as an affine transformation of the true histogram. Thus, through normalization, we can enable the histogram to meet constraints (2) and (3) and mitigate these biases. Notably, not all of the many techniques [49] used to normalize a histogram are applicable, as we must preserve the smoothness of the histogram derived in the previous steps. To do this, we first subtract the minimal frequency $\bar{f}_{\min}$ from all frequencies, if $\bar{f}_{\min} < 0$, to translate them to values above 0 and scale them so that their sum is 1 (lines 11~12), as exampled in Fig. 2 (c).

6 MDR*: AN OPTIMIZED PROTOCOL

MDR, which relies on a given T_ξ, is effective for smooth enough cases but is not general enough. In practice, the smoothness of the original histogram can vary, and the assumption that the smoothness $\xi \to 0$ may not hold, making MDR ineffective. To tackle this, we propose an optimized protocol, MDR*.

6.1 Design Rationale

Intuitively, we should pick a threshold T_ξ in MDR according to the histogram smoothness. Without knowing the exact smoothness, we can alternatively enumerate κ possible thresholds $\mathbf{T} = \{T_1, T_2, \ldots, T_\kappa\}$ and generate candidates $\mathcal{F} = \{\bar{\mathbf{F}}^1, \bar{\mathbf{F}}^2, \ldots, \bar{\mathbf{F}}^\kappa\}$ through MDR. Some over-smoothed and under-smoothed phenomena may occur in these candidates. When the selected threshold T_k is too small for the original smoothness, the significant peaks in the original histogram may be flattened, resulting in *over-smoothed*. On the contrary, if the selected threshold T_k is too large, the noise poisoned by Byzantine users cannot be detected and rectified, resulting in *under-smoothed*.

Instead of picking a candidate directly, we adopt a compromise strategy. We assign a weight w_k to each candidate $\bar{\mathbf{F}}^k, k \in [\kappa]$, and adopt the weighted average frequencies of candidates, to alleviate the bias from extremely over-smoothed or under-smoothed histograms. The candidate closer to the true histogram $\tilde{\mathbf{F}}$ should have more weight in the averaged result and vice versa. To do this, we minimize the weighted deviations from the candidates to $\tilde{\mathbf{F}}$ and solve for the optimal histogram $\tilde{\mathbf{F}}$. We measure the deviation using the square difference. The optimization problem is presented in Eq.(16), where $S(\tilde{f}_j)$ is the smoothed value of $\tilde{f}_j$ by SMA.

$$\min \sum_{k\in[\kappa]} w_k \sum_{j\in[d]} (\bar{f}_j^k - \tilde{f}_j)^2 + \beta \sum_{j\in[d]} (\tilde{f}_j - S(\tilde{f}_j))^2 \tag{16}$$

$$s.t. \sum_{k\in[\kappa]} \exp(-w_k) = 1 \tag{17}$$

$$\sum_{j\in[d]} \tilde{f}_j = 1 \tag{18}$$

$$\forall j \in [d], \tilde{f}_j \geq 0 \tag{19}$$

To bound the range of w_k, we use the constraint in Eq.(17) inspired by [35]. This constraint is convex; it scales the range of w_k to $(0, \infty)$, amplifying the influence of the deviation between $\bar{\mathbf{F}}^k$ and $\tilde{\mathbf{F}}$ on w_k, which we will demonstrate in Section 6.2.1. To ensure both smoothness and frequency constraints, we introduce a penalty for smoothness in the second term of Eq.(16) and constraints for frequencies in Eqs. (18) and (19). The smoothness penalty is the square difference between frequencies and their smoothed values [15]. We set the coefficient $\beta = \sum_{k\in[\kappa]} w_k$ to make the weighted deviations and the smoothness penalty in Eq.(16) equally important.

6.2 Solving Optimization Problem

To solve this optimization problem, we use two-layer iterations. We use the block coordinate descent method in the outer layer iteration to update $\tilde{\mathbf{F}} = \{\tilde{f}_1, \tilde{f}_2, \ldots, \tilde{f}_d\}$ and $\mathbf{W} = \{w_1, w_2, \ldots, w_\kappa\}$ alternatively, reducing the objective function jointly. In each outer layer iteration, we fix one variable between $\tilde{\mathbf{F}}$ and $\mathbf{W}$ and calculate the optimal solution for the other variable. Because we cannot obtain the closed solution directly when we update $\tilde{\mathbf{F}}$, we derive an approximate optimal solution by an inner layer iteration. This algorithm, is presented in Algorithm 4, with details in what follows.

6.2.1 Update of $\mathbf{W}$. First, we fix $\tilde{\mathbf{F}}$ and calculate the optimal weights $\mathbf{W}$. In the beginning, given the candidates $\mathcal{F}$, we can set $\tilde{\mathbf{F}}$ as the average frequencies of $\mathcal{F}$. For simplicity, let $\mathscr{D}^k$ =

$\sum_{j\in[d]}(\bar{f}_j^k - \tilde{f}_j)^2$ and $\mathscr{D}_s = \sum_{j\in[d]}(\tilde{f}_j - S(\tilde{f}_j))^2$. Based on the Lagrange multiplier, we derive that the optimal w_k for any $k \in [\kappa]$ is

$$w_k = -\ln \frac{\mathscr{D}^k + \mathscr{D}_s}{\sum_{k\in[\kappa]} \mathscr{D}^k + \kappa \mathscr{D}_s}. \tag{20}$$

By ignoring $\mathscr{D}_s$, which is uncorrelated with $\bar{\mathbf{F}}^k$, the negative log function amplifies the normalized deviation $\mathscr{D}^k$ from $[0, 1]$ to $(0, \infty)$. $\bar{\mathbf{F}}^k$, which has a slight deviation from $\tilde{\mathbf{F}}$, will have considerable weight w_k and dominate the aggregated result.

6.2.2 Update of $\tilde{\mathbf{F}}$. Next, we can fix $\mathbf{W}$ to update the frequencies $\tilde{\mathbf{F}}$. However, this optimization is non-trivial with inequality constraints and a smoothness penalty. Thus, we transform it to an approximate optimization problem in Eq.(21) and solve it by an iterative procedure, where notations with superscript (t) denote the value in round t. When the objective function converges, the value of $\tilde{\mathbf{F}}$ tends to be stable, i.e., $\tilde{f}_j^{(t-1)}$ has almost the same value as $\tilde{f}_j^{(t)}$, and Eq.(21) is equivalent to Eq.(16).

$$\min \sum_{k\in[\kappa]} w_k \sum_{j\in[d]} (\bar{f}_j^k - \tilde{f}_j^{(t)})^2 + \beta \sum_{j\in[d]} (\tilde{f}_j^{(t)} - S(\tilde{f}_j^{(t-1)}))^2 \tag{21}$$

$$s.t. \sum_{j\in[d]} \tilde{f}_j^{(t)} = 1 \tag{22}$$

$$\forall j \in [d], \tilde{f}_j^{(t)} \geq 0 \tag{23}$$

We update $\tilde{f}_j^{(t)}$ with $\tilde{f}_j^{(t-1)}$ iteratively to solve this problem. At the beginning, we set $\forall j \in [d], \tilde{f}_j^{(0)} = 0$, which is equivalent to considering the squared L2 norm of $\tilde{\mathbf{F}}^{(1)}$ as the smoothness penalty. Then, given $\tilde{f}_j^{(t-1)}$, Eq.(21) becomes a convex problem that can be solved with KKT conditions. The proof is presented in Appendix A.6 in the full version [46]. Let the domain of the solution includes two parts $[d] = D_0 \cup D_+$. We can determine that for $j \in D_0$, $\tilde{f}_j^{(t)} = 0$ holds, and for $j \in D_+$, we have

$$\tilde{f}_j^{(t)\prime} = \frac{\sum_{k\in[\kappa]} w_k \bar{f}_j^k + \beta S(\tilde{f}_j^{(t-1)})}{\sum_{k\in[\kappa]} w_k + \beta}, \tilde{f}_j^{(t)} = \tilde{f}_j^{(t)\prime} + \frac{1 - \sum_{j\in D_+} \tilde{f}_j^{(t)\prime}}{|D_+|}. \tag{24}$$

When $D_+ = [d]$, $\tilde{f}_j^{(t)\prime}$ is the exact optimal solution to Eq.(21) with the only constraint being Eq.(22). The several items $j \in D_0$ prompt this result to meet the condition in Eq.(23), allowing us to derive $\tilde{f}_j^{(t)}$. Therefore, without knowing the exact D_+, we can set $D_+ = [d]$ and $D_0 = \phi$ at the start and update the j with $\tilde{f}_j^{(t)} \leq 0$ to D_0 iteratively. This iterative process gradually improves our results to meet KKT conditions and finds the optimal result.

We iteratively update $\tilde{\mathbf{F}}$ and $\mathbf{W}$ until an optimized result is derived. The iteration stops when the change in the objective function is smaller than a threshold θ, which can be set to less than $1/N$.

6.3 Candidate Set Selection

In this protocol, candidate set selection is a critical step. As the analyzer may have no prior knowledge of the histogram to be estimated, our idea is to enumerate all possible thresholds and trim extreme values to generate candidates. Concretely, let $\Delta' = \{\delta'_1, \delta'_2, \ldots, \delta'_d\}$ denote the absolute differences between bins' frequencies in $\hat{\mathbf{F}}$ and their smoothed frequencies. Obviously, $T_k, k \in [\kappa]$ only works in the range $[\Delta'_{\min}, \Delta'_{\max}]$, where $\Delta'_{\min}$ (resp. $\Delta'_{\max}$) represents the minimal (resp. maximal) value in Δ'. Moreover, each T_k fed to MDR will return a rectified histogram as well as a label vector $\mathbf{I}$. The number of benign labels ($I_j = 1$) in $\mathbf{I}$, referred to as b, indicates how many benign bins are used for bin rebuilding by averaging, implying the degrees of smoothness and rectification of the histogram. Thus, we trim $[\Delta'_{\min}, \Delta'_{\max}]$ to (Δ'_l, Δ'_r) with $b \in [0.6d, 0.9d]$ for effective thresholds, and then we generate candidates from it.

Algorithm 4: MDR*

Input: the estimated histogram $\hat{\mathbf{F}}$, the feasible parameters $\mathbf{T} = \{T_1, T_2, \ldots, T_\kappa\}$, the threshold θ for convergence
Output: the rectified histogram $\tilde{\mathbf{F}}$

```
Obtain candidates F via MDR with T;
// Update W and F̃ in Eq.(16) iteratively
Initialize f̃_j = Σ_{k∈[κ]} f̄_j^k/κ for ∀j ∈ [d] and L=∞, L_0=−∞;
while |L − L_0| > θ do
    L_0 = L;
    Update W with Eq.(20);
    // Update F̃ via solving Eq.(21) iteratively
    Initialize t = 1, F̃^(t−1) = {0}^d and L_1 = −∞ ;
    while |L − L_1| > θ do
        Initialize D_0 = φ, D_+ = [d] − D_0, and let L_1 = L;
        Update F̃^(t) with Eq.(24);
        while ∃j ∈ [d], f̃_j^(t) < 0 do
            Add j with f̃_j^(t) < 0 to D_0, update D_+ = [d] − D_0;
            Update F̃^(t) with Eq.(24);
        Obtain the objective function L with Eq.(21);
        t = t + 1, F̃ = F̃^(t);
return F̃ ;
```

7 EXPERIMENTAL EVALUATIONS

In this section, based on the BRSM framework, we evaluate our defense methods on several datasets and attack methods to demonstrate their effectiveness.

7.1 Experimental Setting

We implement all experiments in Python 3.8 on a Linux server with an Intel® Xeon® E5-2603 1.70 GHz CPU and 96 G memory. The experimental configurations are as follows.

7.1.1 Experiments Design. We focus on the defense against poisoning attacks on the histogram estimation in the shuffle model and adopt GRR and UE as typical perturbation mechanisms. Based on the BRSM framework, we evaluate our rectification methods MDR and MDR*, namely **BRSM-MDR** and **BRSM-MDR***, respectively, under several attacks through the *MSE* in Eq.(25), where f_j and $\tilde{f}_j$ are the true and rectified frequencies, respectively.

$$MSE(\tilde{\mathbf{F}}) = \frac{1}{d} \sum_{j\in[d]} (\tilde{f}_j - f_j)^2 \tag{25}$$

We also compare BRSM-MDR and BRSM-MDR* with the normalization method Norm proposed by [7]. Norm is originally designed to defend against poisoning attacks for frequency estimation in

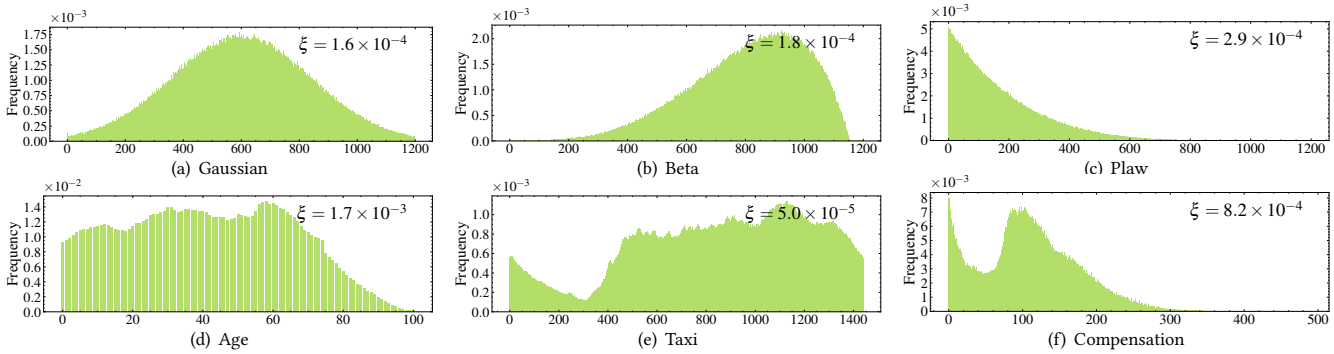

Figure 3: Histograms of Datasets

LDP and works by subtracting the minimal frequency from all frequencies and then normalizing them to sum to one. To adapt Norm to the shuffle model while maintaining privacy, we integrate it into the BRSM framework by replacing the rectification component. The resulting method is denoted by **BRSM-Norm**. Because we assume that the analyzer does not know detailed attack information, other sophisticated defenses in LDP focusing on specific attacks with assumptions of significant prior knowledge about attacks beyond the scope of this work. By default, we set the parameters $\epsilon = 0.8$, $\delta = 10^{-8}$, and the Byzantine user proportion $\alpha = 10\%$.

7.1.2 Attacks. We evaluate three proposed attacks, namely MLA, ASA, and RDA, and an existing one named Maximal Gain Attack (MGA) [7]. In MGA, Byzantine users try to maximize their gain on target values. We set the default target number as $d \times 2\%$.

7.1.3 Datasets. We utilize three synthetic datasets and three real ones. The three synthetic datasets are drawn from the **Gaussian** distribution $\mathcal{N}(d/2, d/5)$, the **beta**-binomial distribution $Beta(5, 2)$, and the **power-law (Plaw)** distribution with a probability density function $p(x) \propto x^4$. These data has all been discretized to the domain $[1200]$. The real datasets include age, taxi pick-up times, and total compensation data. The **Age** dataset samples 1/5 of the population from Canadian 2021 census data and includes 7,398,397 people in the age range $[0, 100]$. The **Taxi** dataset reports pick-up time from the New York taxi in January 2018 and contains 8,760,687 records in a day. The **Compensation** dataset is from the San Francisco controller's office database in 2015 and includes 608,900 city employees' records with compensation in $[10, 500] \times 1,000\$$. We plot the histograms of these datasets in Fig.3, and label their smoothness ξ at the upper-right corner in Fig.3.

7.2 Experimental Results

In this subsection, we evaluate the defense performance of our methods, present an important observation regarding the correlation between attack and defense, and then evaluate the impact of different parameters.

7.2.1 Overall Performance on Defense. We compare the overall performance of three methods (BRSM-MDR, BRSM-MDR*, BRSM-Norm) to defend against four types of attacks (MLA, ASA, RDA, and MGA) over six datasets in Fig. 4. **Overall, BRSM-MDR* performs the best with the lowest MSE to defend against attacks.** While in some cases, the result of BRSM-MDR* has a similar MSE of poisoned result, especially for ASA attack. This is because ASA injects uniformly distributed noise, which does not break but strengthens the estimated histogram's smoothness, as described in Section 3.3. However, this attack will maintain the distribution shape and not cause severe utility damage, which is tolerable. BRSM-Norm may work well against this attack in some cases, e.g., for Age datasets. It translates the estimated histogram to make the minimal frequency always zero, thereby counteracting the influence of uniform poisoning noises. But when the minimal frequency is larger than zero, like in Taxi dataset, it performs poorly. When there are too many zeros in the original histogram, BRSM-Norm is also no longer effective, as some values' frequencies after poisoning may remain zero. For other attacks, BRSM-Norm is almost useless.

On the other hand, we observe that both BRSM-MDR and BRSM-MDR* work well and have similar performance when the original histogram is smooth enough. Otherwise, their results may not be good (e.g., Fig. 4(c) and3(d)). For BRSM-MDR, this is because a given threshold could be too small, resulting in an over-smoothed result. Fortunately, BRSM-MDR* attempts to find an optimal rectified result with an appropriate threshold while avoiding the traps of under-smoothness or over-smoothness. It finally achieves a better-rectified result against MLA and MGA or a comparable result to a poisoned one against RDA and ASA. This means that even in some applications where the histogram is not smooth enough, it has no hurt but benefits to adopt BRSM-MDR* for rectification.

7.2.2 Correlation between Attack and Defense. Based on the optimized method BRSM-MDR*, which is the best method we presented, we plot the *MSE*s of the attacked and the rectified results under various attacks and datasets in Fig. 5. The colored short lines represent the *MSE*s of various attacks. The truncated bars represent the range of result errors after attacking, with (i.e., in yellow) or without (i.e., in purple) BRSM-MDR* rectification. As discussed in Section 3.2, MLA and ASA approach the maximal and the minimal utility damages, respectively, while other attacks, such as MGA and RDA, lie between them. Interestingly, we make the following observation:

The utility losses (i.e., MSEs) from attack and defense are almost inversely proportional. The larger utility loss by an attack, the more significant this loss is rectified by BRSM-MDR. It makes sense that the more aggressive the attack, the more apparent characteristics will*

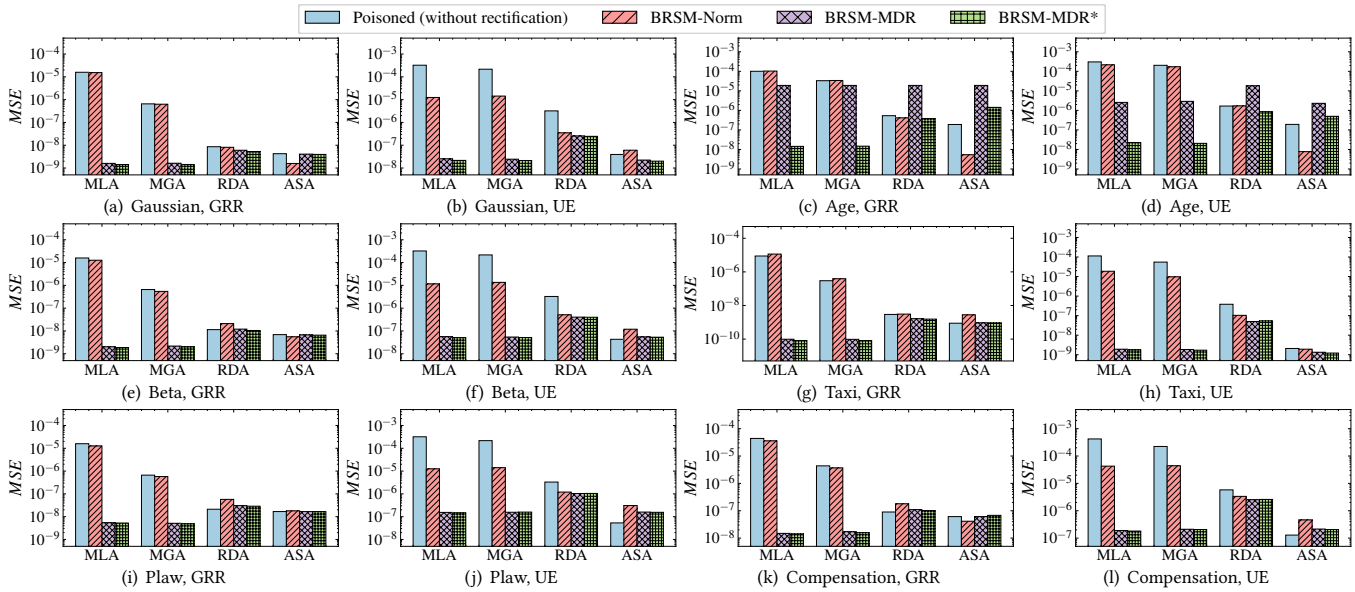

Figure 4: *MSE*s of Poisoned and Rectified Results

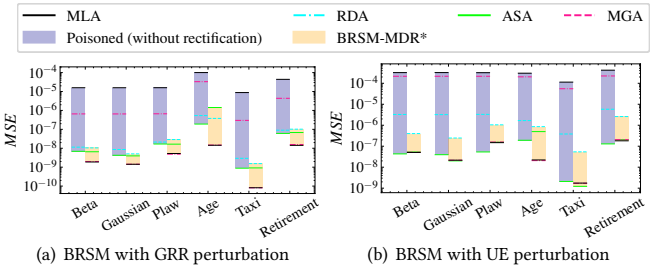

Figure 5: Correlation between Attack and Defense

expose and the more likely it is to be accurately detected and rectified. Therefore, attackers are advised not to be eager for instant success and quick profits when facing intelligent defenses.

Additionally, this result shows the stable and robust performance of BRSM-MDR*. The method can be applied to various datasets and can consistently suppress the high and large-scale error range from various attacks to a relatively small range.

7.2.3 Impact of Parameters. To evaluate the impact of different parameters on the performance of the BRSM framework, we conduct a set of experiments on a real dataset Taxi with GRR as the local perturbation mechanism.

Impact of Privacy Parameter ϵ. Figs.6 (a)~(d) show MSEs of poisoned and rectified results with increasing ϵ from 0.1 to 1.6, under four attacks, respectively. Overall, the biases caused by Byzantine users decrease with increasing ϵ, which is consistent with Eq.(7). For BRSM-MDR and BRSM-MDR*, as a large ϵ leads to a minor perturbation variance σ and a smoother estimated result, both of them benefit from it and produce a small MSE. When ϵ becomes small, the perturbation by honest users will lead to non-negligible noises. Our framework BRSM can also reduce this noise via smoothing, lowering the MSEs of attacks.

Impact of Byzantine Proportion α. Figs.6 (e)~(h) show MSEs of all poisoned and rectified results while increasing the Byzantine user proportion α from 5% to 50%, for four attacks. BRSM-MDR and BRSM-MDR* perform the best in most cases. With the increasing α, more Byzantine users participating would cause more destruction in results. The BRSM-Norm method for ASA is an exception, which sets the minimum frequency to zero and exhibits a slightly decreased MSE as α increases. However, for the Taxi dataset, which has a non-zero minimal frequency, BRSM-Norm can produce even greater additional MSE than that poisoned by ASA when α is less than 20%. In other situations, the BRSM-Norm method can rectify a histogram significantly shifted upwards by ASA with numerous Byzantine users, leading to a relatively small MSE.

Impact of Smoothness ξ. Figs.6 (i)~(l) show MSEs of poisoned and rectified results with increasing ξ for four attacks, where a smaller ξ indicates a higher degree of smoothness. The result shows that the smoother the original histogram is, the better-rectified result we will have. In the Taxi dataset, the various ξ from 0.00005 to 0.00027 are generated by varying the time interval of a bin from 1 to 5 minutes. It is worth noting that the MSE of BRSM-MDR with a fixed threshold T_ξ increases significantly when ξ reaches a certain value. In contrast, BRSM-MDR* is more stable when confronted with varying smoothness and always performs the best.

8 RELATED WORK

DP and LDP have been extensively studied in the literatures [14, 16, 18, 22, 54, 55] to protect individual privacy. To bridge the gap of privacy and utility of DP and LDP, the shuffle model [5] was proposed. Many studies on the shuffle model focused on either analyzing the privacy amplification effect [3, 11, 21, 23, 38] or improving the model's utility for various data collection and analysis tasks, such as frequency estimation [1–3, 11, 12, 20, 25, 50], value summation [3, 4, 24, 26, 27], vector summation [44, 47], subgraph

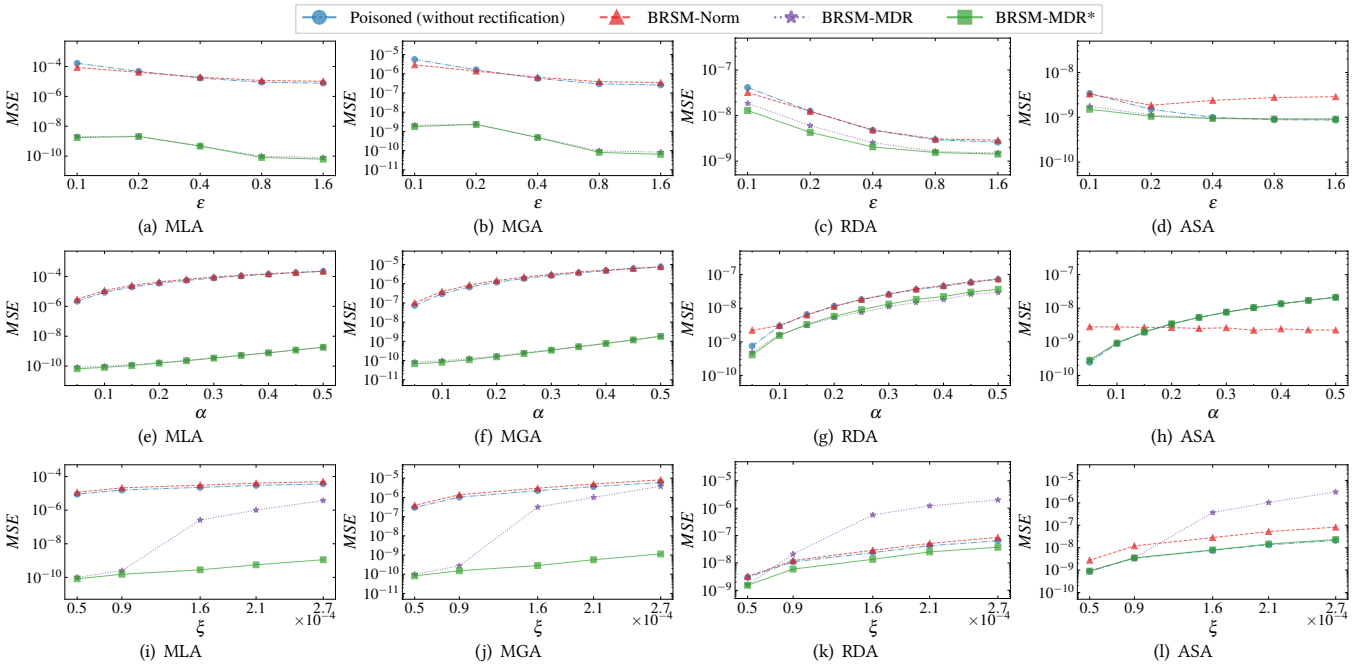

Figure 6: Results of BRSM by Varying Parameters ϵ, α, ξ on Taxi Dataset

counting [30], stochastic convex optimization [9], reinforcement learning [13], and federated learning [28, 40].

However, most works in the shuffle model did not consider the possibility of malicious users and focused solely on malicious behaviors of the shuffler and the analyzer, investigating their compromise, or collusion [5, 38, 50]. Only Balcer et al. [2] has recognized that malicious users could evade their responsibility for randomizing messages and proposed the robust shuffle privacy that only accounts for randomness from honest users.

In basic scenarios of LDP, the poisoning attacks, launched by malicious or Byzantine users and threatening the utility, have been studied. A theoretical study by Cheu et al. [10] proposed the input-manipulation and output-manipulation attacks, demonstrating their inevitability for local randomizers. Kato et al. [32] defended against the output-manipulation attack using verifiable randomization mechanisms with MPC. However, massive communication and computation costs make this method inapplicable for large-scale data collection scenarios. Moreover, researchers have considered different poisoning goals and discussed their defenses. Cao et al. [7] proposed the MGA for frequency estimation and heavy hitter identification tasks to maximize the gains of targeted items. They proposed frequent itemset mining-based and conditional probability-based attacker detection methods against MGA when the analyzer knows the number of targets and the normalization method (Norm) against attacks with random noises poisoned. Further, Wu et al. [53] investigated the maximal gain attack (M2GA) for key-value data and proposed one-class classifier-based and anomaly score-based attacker detection methods with the assumption that the analyzer knows the exact proportion of attackers. Li et al. [34] proposed the fine-grained data poisoning attack to control the estimated mean and variance to the intended values and designed a clustering-based defense only effective for cases with only a few attackers among users. Du et al. [17] proposed an opportunistic-and-colluding threat model to bias the estimated mean to one side and developed a general multi-group Differential Aggregation Protocol (DAP) to defend against it. However, most defenses in LDP rely on additional assumptions that may be impractical, such as the accessibility of attack patterns to the analyzer. Defending against poisoning attacks based on realistic assumptions is so critical and challenging, especially in the shuffle model, where the attacks are more powerful because Byzantine users can be anonymous to undermine both privacy and utility.

9 CONCLUSION

This paper investigates histogram estimation in the shuffle model, where a set of Byzantine users may launch data poisoning attacks. For the sake of a Byzantine-robust estimation, a BRSM framework is proposed to eliminate privacy damage and mitigate utility loss. A trade-off from the perspective of Byzantine users was discovered between strengthening the utility damage and evading been detected by maintaining the histogram smoothness. Inspired by this, we presented MDR to rectify the poisoned histogram and optimized it to MDR* to fit the complex practical cases with different levels of smoothness in original histograms. The extensive experiments demonstrated the effectiveness of the defenses, and it was concluded that more aggressive attacks are easier to detect and rectify. A critical direction for future work is to defend against poisoning attacks for more general scenarios and tasks, such as privately graph data collection and analysis.

REFERENCES

[1] Victor Balcer and Albert Cheu. 2020. Separating Local & Shuffled Differential Privacy via Histograms. In *1st Conference on Information-Theoretic Cryptography, ITC 2020, June 17-19, 2020, Boston, MA, USA (LIPIcs)*, Yael Tauman Kalai, Adam D. Smith, and Daniel Wichs (Eds.), Vol. 163. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 1:1–1:14.

[2] Victor Balcer, Albert Cheu, Matthew Joseph, and Jieming Mao. 2021. Connecting Robust Shuffle Privacy and Pan-Privacy. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, Dániel Marx (Ed.). SIAM, 2384–2403.

[3] Borja Balle, James Bell, Adrià Gascón, and Kobbi Nissim. 2019. The Privacy Blanket of the Shuffle Model. In *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part II (Lecture Notes in Computer Science)*, Alexandra Boldyreva and Daniele Micciancio (Eds.), Vol. 11693. Springer, 638–667.

[4] Borja Balle, James Bell, Adrià Gascón, and Kobbi Nissim. 2020. Private Summation in the Multi-Message Shuffle Model. In *CCS '20: 2020 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, USA, November 9-13, 2020*, Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna (Eds.). ACM, 657–676.

[5] Andrea Bittau, Úlfar Erlingsson, Petros Maniatis, Ilya Mironov, Ananth Raghunathan, David Lie, Mitch Rudominer, Ushasree Kode, Julien Tinnés, and Bernhard Seefeld. 2017. Prochlo: Strong Privacy for Analytics in the Crowd. In *Proceedings of the 26th Symposium on Operating Systems Principles, Shanghai, China, October 28-31, 2017*. ACM, 441–459.

[6] Ricardo J. G. B. Campello, Davoud Moulavi, and Jörg Sander. 2013. Density-Based Clustering Based on Hierarchical Density Estimates. In *Advances in Knowledge Discovery and Data Mining, 17th Pacific-Asia Conference, PAKDD 2013, Gold Coast, Australia, April 14-17, 2013, Proceedings, Part II (Lecture Notes in Computer Science)*, Jian Pei, Vincent S. Tseng, Longbing Cao, Hiroshi Motoda, and Guandong Xu (Eds.), Vol. 7819. Springer, 160–172.

[7] Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. 2021. Data Poisoning Attacks to Local Differential Privacy Protocols. In *30th USENIX Security Symposium, USENIX Security 2021, August 11-13, 2021*, Michael Bailey and Rachel Greenstadt (Eds.). USENIX Association, 947–964.

[8] TH Hubert Chan, Elaine Shi, and Dawn Song. 2012. Optimal lower bound for differentially private multi-party aggregation. In *Algorithms–ESA 2012: 20th Annual European Symposium, Ljubljana, Slovenia, September 10-12, 2012. Proceedings 20*. Springer, 277–288.

[9] Albert Cheu, Matthew Joseph, Jieming Mao, and Binghui Peng. 2022. Shuffle Private Stochastic Convex Optimization. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

[10] Albert Cheu, Adam D. Smith, and Jonathan R. Ullman. 2021. Manipulation Attacks in Local Differential Privacy. In *42nd IEEE Symposium on Security and Privacy, SP 2021, San Francisco, CA, USA, 24-27 May 2021*. IEEE, 883–900.

[11] Albert Cheu, Adam D. Smith, Jonathan R. Ullman, David Zeber, and Maxim Zhilyaev. 2019. Distributed Differential Privacy via Shuffling. In *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part I (Lecture Notes in Computer Science)*, Yuval Ishai and Vincent Rijmen (Eds.), Vol. 11476. Springer, 375–403.

[12] Albert Cheu and Maxim Zhilyaev. 2022. Differentially Private Histograms in the Shuffle Model from Fake Users. In *43rd IEEE Symposium on Security and Privacy, SP 2022, San Francisco, CA, USA, May 22-26, 2022*. IEEE, 440–457.

[13] Sayak Ray Chowdhury and Xingyu Zhou. 2022. Shuffle Private Linear Contextual Bandits. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA (Proceedings of Machine Learning Research)*, Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato (Eds.), Vol. 162. PMLR, 3984–4009.

[14] Graham Cormode, Somesh Jha, Tejas Kulkarni, Ninghui Li, Divesh Srivastava, and Tianhao Wang. 2018. Privacy at Scale: Local Differential Privacy in Practice. In *Proceedings of the 2018 International Conference on Management of Data, SIGMOD Conference 2018, Houston, TX, USA, June 10-15, 2018*, Gautam Das, Christopher M. Jermaine, and Philip A. Bernstein (Eds.). ACM, 1655–1658.

[15] Clayton V Deutsch. 1996. Constrained smoothing of histograms and scatterplots with simulated annealing. *Technometrics* 38, 3 (1996), 266–274.

[16] Bolin Ding, Janardhan Kulkarni, and Sergey Yekhanin. 2017. Collecting Telemetry Data Privately. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (Eds.). 3571–3580.

[17] Rong Du, Qingqing Ye, Yue Fu, Haibo Hu, Jin Li, Chengfang Fang, and Jie Shi. 2023. Differential Aggregation against General Colluding Attackers. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*. IEEE.

[18] John C. Duchi, Michael I. Jordan, and Martin J. Wainwright. 2013. Local Privacy and Statistical Minimax Rates. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*. IEEE Computer Society, 429–438.

[19] Cynthia Dwork and Aaron Roth. 2014. The Algorithmic Foundations of Differential Privacy. *Found. Trends Theor. Comput. Sci.* 9, 3-4 (2014), 11–64.

[20] Úlfar Erlingsson, Vitaly Feldman, Ilya Mironov, Ananth Raghunathan, Shuang Song, Kunal Talwar, and Abhradeep Thakurta. 2020. Encode, Shuffle, Analyze Privacy Revisited: Formalizations and Empirical Evaluation. *CoRR* abs/2001.03618 (2020).

[21] Úlfar Erlingsson, Vitaly Feldman, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Abhradeep Thakurta. 2019. Amplification by Shuffling: From Local to Central Differential Privacy via Anonymity. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, Timothy M. Chan (Ed.). SIAM, 2468–2479.

[22] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. 2014. RAPPOR: Randomized Aggregatable Privacy-Preserving Ordinal Response. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, AZ, USA, November 3-7, 2014*, Gail-Joon Ahn, Moti Yung, and Ninghui Li (Eds.). ACM, 1054–1067.

[23] Vitaly Feldman, Audra McMillan, and Kunal Talwar. 2021. Hiding Among the Clones: A Simple and Nearly Optimal Analysis of Privacy Amplification by Shuffling. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*. IEEE, 954–964.

[24] Badih Ghazi, Noah Golowich, Ravi Kumar, Pasin Manurangsi, Rasmus Pagh, and Ameya Velingker. 2020. Pure Differentially Private Summation from Anonymous Messages. In *1st Conference on Information-Theoretic Cryptography, ITC 2020, June 17-19, 2020, Boston, MA, USA (LIPIcs)*, Yael Tauman Kalai, Adam D. Smith, and Daniel Wichs (Eds.), Vol. 163. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 15:1–15:23.

[25] Badih Ghazi, Noah Golowich, Ravi Kumar, Rasmus Pagh, and Ameya Velingker. 2021. On the Power of Multiple Anonymous Messages: Frequency Estimation and Selection in the Shuffle Model of Differential Privacy. In *Advances in Cryptology - EUROCRYPT 2021 - 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17-21, 2021, Proceedings, Part III (Lecture Notes in Computer Science)*, Anne Canteaut and François-Xavier Standaert (Eds.), Vol. 12698. Springer, 463–488.

[26] Badih Ghazi, Ravi Kumar, Pasin Manurangsi, Rasmus Pagh, and Amer Sinha. 2021. Differentially Private Aggregation in the Shuffle Model: Almost Central Accuracy in Almost a Single Message. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event (Proceedings of Machine Learning Research)*, Marina Meila and Tong Zhang (Eds.), Vol. 139. PMLR, 3692–3701.

[27] Badih Ghazi, Pasin Manurangsi, Rasmus Pagh, and Ameya Velingker. 2020. Private Aggregation from Fewer Anonymous Messages. In *Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020, Proceedings, Part II (Lecture Notes in Computer Science)*, Anne Canteaut and Yuval Ishai (Eds.), Vol. 12106. Springer, 798–827.

[28] Antonious M. Girgis, Deepesh Data, Suhas N. Diggavi, Peter Kairouz, and Ananda Theertha Suresh. 2021. Shuffled Model of Federated Learning: Privacy, Accuracy and Communication Trade-Offs. *IEEE J. Sel. Areas Inf. Theory* 2, 1 (2021), 464–478.

[29] Rob J Hyndman. 2011. Moving Averages.

[30] Jacob Imola, Takao Murakami, and Kamalika Chaudhuri. 2022. Differentially Private Triangle and 4-Cycle Counting in the Shuffle Model. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, CCS 2022, Los Angeles, CA, USA, November 7-11, 2022*, Heng Yin, Angelos Stavrou, Cas Cremers, and Elaine Shi (Eds.). ACM, 1505–1519.

[31] Jinyuan Jia and Neil Zhenqiang Gong. 2019. Calibrate: Frequency Estimation and Heavy Hitter Identification with Local Differential Privacy via Incorporating Prior Knowledge. In *2019 IEEE Conference on Computer Communications, INFOCOM 2019, Paris, France, April 29 - May 2, 2019*. IEEE, 2008–2016.

[32] Fumiyuki Kato, Yang Cao, and Masatoshi Yoshikawa. 2021. Preventing Manipulation Attack in Local Differential Privacy Using Verifiable Randomization Mechanism. In *Data and Applications Security and Privacy XXXV - 35th Annual IFIP WG 11.3 Conference, DBSec 2021, Calgary, Canada, July 19-20, 2021, Proceedings (Lecture Notes in Computer Science)*, Ken Barker and Kambiz Ghazinour (Eds.), Vol. 12840. Springer, 43–60.

[33] Qingkai Kong, Timmy Siauw, and Alexandre Bayen. 2020. *Python Programming and Numerical Methods: A Guide for Engineers and Scientists*. Academic Press.

[34] Xiaoguang Li, Neil Zhenqiang Gong, Ninghui Li, Wenhai Sun, and Hui Li. 2022. Fine-grained Poisoning Attacks to Local Differential Privacy Protocols for Mean and Variance Estimation. *arXiv preprint arXiv:2205.11782* (2022).

[35] Yaliang Li, Qi Li, Jing Gao, Lu Su, Bo Zhao, Wei Fan, and Jiawei Han. 2016. Conflicts to Harmony: A Framework for Resolving Conflicts in Heterogeneous Data by Truth Discovery. *IEEE Trans. Knowl. Data Eng.* 28, 8 (2016), 1986–1999.

[36] Zitao Li, Tianhao Wang, Milan Lopuhaä-Zwakenberg, Ninghui Li, and Boris Skoric. 2020. Estimating Numerical Distributions under Local Differential Privacy. In *Proceedings of the 2020 International Conference on Management of Data, SIGMOD*

Conference 2020, online conference [Portland, OR, USA], June 14-19, 2020, David Maier, Rachel Pottinger, AnHai Doan, Wang-Chiew Tan, Abdussalam Alawini, and Hung Q. Ngo (Eds.). ACM, 621–635.

[37] Zhetao Li, Zhirun Zheng, Suiming Guo, Bin Guo, Fu Xiao, and Kui Ren. 2022. Disguised as Privacy: Data Poisoning Attacks against Differentially Private Crowdsensing Systems. *IEEE Transactions on Mobile Computing* (2022).

[38] Seng Pei Liew, Tsubasa Takahashi, Shun Takagi, Fumiyuki Kato, Yang Cao, and Masatoshi Yoshikawa. 2022. Network Shuffling: Privacy Amplification via Random Walks. In *SIGMOD '22: International Conference on Management of Data, Philadelphia, PA, USA, June 12 - 17, 2022*, Zachary Ives, Angela Bonifati, and Amr El Abbadi (Eds.). ACM, 773–787.

[39] HaoYang Liu, Jiang Hu, Yongfeng Li, and Zaiwen Wen. 2020. *Optimization: Modeling, Algorithm and Theory (in Chinese)* (3 ed.). Higher Education Press. 195–202 pages.

[40] Ruixuan Liu, Yang Cao, Hong Chen, Ruoyang Guo, and Masatoshi Yoshikawa. 2021. FLAME: Differentially Private Federated Learning in the Shuffle Model. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*. AAAI Press, 8688–8696.

[41] Leland McInnes, John Healy, and Steve Astels. 2017. hdbscan: Hierarchical density based clustering. *J. Open Source Softw.* 2, 11 (2017), 205.

[42] Douglas Nychka. 1990. Some properties of adding a smoothing step to the EM algorithm. *Statistics & probability letters* 9, 2 (1990), 187–193.

[43] Karl Pearson. 1895. X. Contributions to the mathematical theory of evolution.—II. Skew variation in homogeneous material. *Philosophical Transactions of the Royal Society of London.(A.)* 186 (1895), 343–414.

[44] Mary Scott, Graham Cormode, and Carsten Maple. 2022. Aggregation and Transformation of Vector-Valued Messages in the Shuffle Model of Differential Privacy. *IEEE Trans. Inf. Forensics Secur.* 17 (2022), 612–627.

[45] Apple Differential Privacy Team. 2017. Learning with Privacy at Scale. (2017). https://machinelearning.apple.com/research/learning-with-privacy-at-scale

[46] Leixia Wang, Qingqing Ye, Haibo Hu, Kai Huang, and Xiaofeng Meng. 2023. Byzantine-robust Histogram Estimation in the Shuffle Model (A Full version). (2023). https://github.com/LeixiaWang/BRSM.

[47] Shaowei Wang, Jin Li, Yuqiu Qian, Jiachun Du, Wenqing Lin, and Wei Yang. 2021. Hiding Numerical Vectors in Local Private and Shuffled Messages. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, Zhi-Hua Zhou (Ed.). ijcai.org, 3706–3712.

[48] Tianhao Wang, Jeremiah Blocki, Ninghui Li, and Somesh Jha. 2017. Locally Differentially Private Protocols for Frequency Estimation. In *26th USENIX Security Symposium, USENIX Security 2017, Vancouver, BC, Canada, August 16-18, 2017*, Engin Kirda and Thomas Ristenpart (Eds.). USENIX Association, 729–745.

[49] Tianhao Wang, Milan Lopuhaä-Zwakenberg, Zitao Li, Boris Skoric, and Ninghui Li. 2020. Locally Differentially Private Frequency Estimation with Consistency. In *27th Annual Network and Distributed System Security Symposium, NDSS 2020, San Diego, California, USA, February 23-26, 2020*. The Internet Society.

[50] Tianhao Wang, Min Xu, Bolin Ding, Jingren Zhou, Cheng Hong, Zhicong Huang, Ninghui Li, and Somesh Jha. 2020. Improving Utility and Security of the Shuffler-based Differential Privacy. *Proc. VLDB Endow.* 13, 13 (2020), 3545–3558.

[51] Stanley L Warner. 1965. Randomized response: A survey technique for eliminating evasive answer bias. *J. Amer. Statist. Assoc.* 60, 309 (1965), 63–69.

[52] Eric W. Weisstein. 2022. Smooth Function. [EB/OL]. https://mathworld.wolfram.com/SmoothFunction.html Accessed June 21, 2022.

[53] Yongji Wu, Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. 2022. Poisoning Attacks to Local Differential Privacy Protocols for Key-Value Data. In *31st USENIX Security Symposium, USENIX Security 2022, Boston, MA, USA, August 10-12, 2022*, Kevin R. B. Butler and Kurt Thomas (Eds.). USENIX Association, 519–536.

[54] Qingqing Ye, Haibo Hu, Man Ho Au, Xiaofeng Meng, and Xiaokui Xiao. 2020. Towards locally differentially private generic graph metric estimation. In *2020 IEEE 36th International Conference on Data Engineering (ICDE)*. IEEE, 1922–1925.

[55] Qingqing Ye, Haibo Hu, Xiaofeng Meng, and Huadi Zheng. 2019. PrivKV: Key-value data collection with local differential privacy. In *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 317–331.

A APPENDIX

A.1 Proof of Theorem 2

Proof. Because the objective function and the inequality constraints are convex, and the equality constraint is an affine function, the original problem is convex, so the KKT conditions are necessary and sufficient for the problem Eq.(10) [39]. We define the Lagrange function in Eq.(26).

$$\begin{aligned}&\mathcal{L}(\gamma_1,\gamma_2,\ldots,\gamma_m)\\&=-\sum_{j\in[d]}\gamma_j^2+\lambda(\sum_{j\in[d]}\gamma_j-C)+\sum_{j\in[d]}\mu_j(-\gamma_j)+\sum_{j\in[d]}\nu_j(\gamma_j-1)\end{aligned}\tag{26}$$

Then we get KKT conditions as follows.

$$\begin{cases}\forall j:\frac{\partial\mathcal{L}(\gamma_1,\gamma_2,\ldots,\gamma_m)}{\partial\gamma_j}=-2\gamma_j+\lambda-\mu_j+\nu_j=0 & (27a)\\ \sum_{j\in[d]}\gamma_j=C & (27b)\\ \forall j:0\le\gamma_j\le 1 & (27c)\\ \forall j:\mu_j\gamma_j=0 & (27d)\\ \forall j:\nu_j(\gamma_j-1)=0 & (27e)\\ \forall j:\mu_j,\nu_j\ge 0 & (27f)\end{cases}$$

According to Eq.(27a) and Eq.(27b), we have

$$\gamma_j=\frac{1}{2}(\lambda-\mu_j+\nu_j)\tag{28}$$

$$\lambda=\frac{1}{d}(2C+\sum_{j\in[d]}\mu_j-\sum_{j\in[d]}\nu_j)\tag{29}$$

Let the domain $[d]=D_0\cup D_1\cup D_+$.

- For $j\in D_0$, we have $\gamma_j=0$, $\nu_j=0$, and $\mu_j=\lambda>0$.
- For $j\in D_1$, we have $\gamma_j=1$, $\mu_j=0$, and $\nu_j=2-\lambda>0$.
- For $j\in D_+$, we have $\mu_j=0$, $\nu_j=0$, and $0<\gamma_j=\frac{\lambda}{2}<1$.

Then, we have

$$\lambda=\frac{1}{d}(2C+|D_0|\lambda-|D_1|(2-\lambda))\quad\Rightarrow\quad|D_1|=(2C-|D_+|\lambda)/2$$

Taking values in D_0, D_1 and D_+ into the objective function, we derive

$$\begin{aligned}&\sum_{j\in[d]}\gamma_j^2\\&=|D_+|(\frac{\lambda}{2})^2+|D_1|&(30)\\&=|D_+|(\frac{\lambda}{2})^2+C-\frac{\lambda}{2}|D_+|\\&=C+(\frac{\lambda^2}{4}-\frac{\lambda}{2})|D_+|.&(31)\end{aligned}$$

Because $0<\lambda<2$, $\frac{\lambda^2}{4}-\frac{\lambda}{2}<0$. Eq.(31) gets maximal value when $|D_+|$ as small as possible. So, when C is an integer, we have $|D_+|=0$, $|D_1|=C$ and $|D_0|=d-C$. When C is a numeric, we have $|D_+|=1$, $|D_1|=\lfloor C\rfloor$, $|D_0|=d-\lceil C\rceil$, and for $j\in D_+$, $\gamma_j=C-\lfloor C\rfloor$.

In conclusion, Eq.(10) obtains maximal value when $[d]=D_0\cup D_1\cup D_+$, $|D_1|=\lfloor C\rfloor$, $|D_+|=\lceil C\rceil-\lfloor C\rfloor$, and $|D_0|=d-\lceil C\rceil$, where $\forall j\in D_1,\gamma_j=1;\forall j\in D_+,\gamma_j=C-\lfloor C\rfloor$; and $\forall j\in D_0,\gamma_j=0$. □

A.2 Proof of Theorem 4

Except for the general privacy amplification theorem, there is a tighter privacy amplification theorem, Theorem 7[23], for the GRR mechanism. This subsection proves the appropriate m for cases where these two theorems hold separately, shown in Theorem 8.

Theorem 7 (Privacy Amplification for GRR[23]). *Given n honest users, if each user perturbs his value using GRR with ϵ_l-LDP, then for any $\delta\in(0,1]$ and $\epsilon_l\le\ln(\frac{n}{16\ln(2/\delta)})$, the shuffled messages satisfy (ϵ,δ)-DP, where*

$$\epsilon\le\ln\left(1+(e^{\epsilon_l}-1)\left(\frac{4\sqrt{2(d+1)\ln(4/\delta)}}{\sqrt{(e^{\epsilon_l}+d-1)dn}}+\frac{4(d+1)}{dn}\right)\right).\tag{32}$$

Theorem 8 (Expanded Privacy Guarantee). *Assuming there are N users and at most T_α Byzantine users in proportion, Theorem 1 holds when $m=NT_\alpha\frac{d}{2e^{\epsilon_l}+d-2}$, and Theorem 7 holds when $m=NT_\alpha\frac{d}{e^{\epsilon_l}+d-1}$.*

Proof. Because the analyzer's behaviors are post-processing of the shuffled messages, which does not influence privacy, We focus on the privacy of the shuffled messages. Suppose the neighboring databases $\mathbf{X}$ and $\mathbf{X}'$ consist of all users' values and differ on the first value, i.e., $x_1\neq x_1'$. Feldman et al. [23] propose that for any $x\in\mathbf{X}\backslash\{x_1,x_1'\}$, the randomized value with ϵ_l-LDP can be decomposed as Eq.(33), where $\mathcal{R}$ represents the output distribution of randomization, and LO refers to a "left-over" distribution.

$$\mathcal{R}(x)=\frac{e^{-\epsilon_l}}{2}\mathcal{R}(x_1)+\frac{e^{-\epsilon_l}}{2}\mathcal{R}(x_1')+(1-e^{-\epsilon_l})LO(x)\tag{33}$$

Reviewing that without Byzantine users, after shuffling, there're $\mu=Ne^{-\epsilon_l}$ messages in the expectation that are $\mathcal{R}(x_1)$ or $\mathcal{R}(x_1')$, independent of the truthful value x. Measuring the uncertainty brought by these messages by Chernoff bound and Hoeffding's inequality, Feldman et al. derive Lemma 3.5 in [23]. Further considering the privacy amplification of outputting its input for x_1 and x_1', they derive Theorem 1.

In BRSM, to remedy the privacy that may be undermined due to Byzantine users, the shuffler generates some perturbed uniform noises. we can decompose the perturbed uniform random $\mathcal{R}(r)\in\mathbf{R}$ as follows.

$$\begin{aligned}\mathcal{R}(r)&=\sum_{x\in[d]}\frac{1}{d}\mathcal{R}(x)\\&=\frac{1}{d}\mathcal{R}(x_1)+\frac{1}{d}\mathcal{R}(x_1')+\sum_{x\in[d]\backslash\{x_1,x_1'\}}\mathcal{R}(x)\\&=\frac{1}{d}\mathcal{R}(x_1)+\frac{1}{d}\mathcal{R}(x_1')+\sum_{x\in[d]\backslash\{x_1,x_1'\}}\frac{e^{-\epsilon_l}}{2}\mathcal{R}(x_1)+\frac{e^{-\epsilon_l}}{2}\mathcal{R}(x_1')\\&\quad+(1-e^{-\epsilon_l})LO(x)\\&=\left(\frac{1}{d}+\frac{(d-2)e^{-\epsilon_l}}{2d}\right)\mathcal{R}(x_1)+\left(\frac{1}{d}+\frac{(d-2)e^{-\epsilon_l}}{2d}\right)\mathcal{R}(x_1')\\&\quad+\sum_{x\in[d]\backslash\{x_1,x_1'\}}\left(1-\frac{2}{d}-\frac{(d-2)e^{-\epsilon_l}}{d}\right)LO(x)\end{aligned}$$

To ensure Theorem 1 still holds for the case that there're at most $T_\alpha N$ Byzantine users, we have to ensure that the mixed values of $(1-T_\alpha)N$ honest users' outputs and m perturbed uniform randoms have the same number of data-independent values as μ. Here,

the number of data-independent values $\mathcal{R}(x_1)$ and $\mathcal{R}(x_1')$ follows Bernoulli trials, and the expectation is

$$\mu' = N(1-T_\alpha)e^{-\epsilon_l} + m\left(\frac{2}{d} + \frac{(d-2)e^{-\epsilon_l}}{d}\right).$$

Let $\mu = \mu'$, we get the solution

$$m = NT_\alpha \frac{d}{2e^{\epsilon_l} + d - 2}.$$

For GRR, [23] propose a tighter amplified privacy bound, Theorem 7. In its proof, each user's output $\mathcal{R}(x)$ is decomposed as Eq.(34), where $\theta = \frac{d}{(e^{\epsilon_l}+d-1)(d+1)}$, $\mathbb{1}_x$ denotes the probability of always outputting x, and $U([d])$ represents the uniform distribution on $[d]$.

$$\mathcal{R}(x) = \theta\mathbb{1}_{x_1} + \theta\mathbb{1}_{x_1'} + \theta\mathbb{1}_{x_1}U([d]) + (1-3\theta)LO(x) \tag{34}$$

Similarly, we decompose $\mathcal{R}(r) \in \mathbf{R}'$ as follows.

$$\begin{aligned}\mathcal{R}(r) &= \frac{1}{d+1}\sum_{x\in[d]}\mathbb{1}_x + \frac{1}{d+1}U([d]) \\ &= \frac{1}{d+1}\mathbb{1}_{x_1} + \frac{1}{d+1}\mathbb{1}_{x_1'} + \frac{1}{d+1}U([d]) \\ &\quad + \sum_{x\in[d]\setminus\{x_1,x_1'\}}\left(1-\frac{3}{d+1}\right)LO(x)\end{aligned}$$

According to the proof of Corollary 4.2 in [23], to maintain this theorem in Byzantine cases, we must keep the expectation of outputting $\mathcal{R}(x_1)$ or $\mathcal{R}(x_1')$ the same as no Byzantine users, i.e.,

$$\frac{d}{(d+1)(e^{\epsilon_l}+d-1)}N = \frac{d}{(d+1)(e^{\epsilon_l}+d-1)}N(1-T_\alpha) + m\frac{1}{d+1}.$$

Then, we get $m = NT_\alpha\frac{d}{e^{\epsilon_l}+d-1}$. □

A.3 Proof of Theorem 5

Proof. For each $j \in [d]$,

$$\begin{aligned}\mathbb{E}(\hat{f}_j) &= \mathbb{E}\left(\frac{n_j-(N+m)q}{N(p-q)} - \frac{m}{Nd}\right) \\ &= \frac{N(1-\alpha)\left(f_j p + (1-f_j)q\right) + N\alpha\gamma_j + m(p/d + (1-1/d)q)}{N(p-q)} \\ &\quad - \frac{(N+m)q}{N(p-q)} - \frac{m}{Nq} \\ &= (1-\alpha)f_j + \alpha\frac{\gamma_j - q}{p-q}\end{aligned}$$

$$\begin{aligned}\mathrm{Var}(\hat{f}_j) &= \mathrm{Var}\left(\frac{n_j-(N+m)q}{N(p-q)} - \frac{m}{Nd}\right) \\ &= \frac{\mathrm{Var}(n_j)}{N^2(p-q)^2} \\ &= \frac{(N(1-\alpha)f_j + m/d)p(1-p) + (N(1-\alpha)(1-f_j)}{N^2(p-q)^2} \\ &\quad + \frac{m(1-1/d))q(1-q) + N\alpha\gamma_j(1-\gamma_j)}{N^2(p-q)^2} \\ &= \frac{(N(1-\alpha)f_j + m/d)(1-p-q)}{N^2(p-q)} + \frac{(N(1-\alpha)+m)(1-q)q}{N^2(p-q)^2} \\ &\quad + \frac{\alpha\gamma_j(1-\gamma_j)}{N(p-q)^2}\end{aligned}$$

□

A.4 Proof of Theorem 6

Proof. Let χ_j denote the noise added by perturbation and attacks. We have $\hat{f}_j = f_j + \chi_j$ and $\mathbb{E}[\chi_j] = \alpha\left((\gamma_j - q)/(p-q) - f_j\right)$.

Firstly, we calculate $|\hat{f}_j - \bar{f}_j|$ in round t_1, which has various values with various labels of bin j and its neighbors. We analyze the eight situations at length and summarize them in Table 2, where $\Delta_j^s = f_j - f_{j+1}$ and $-\xi \le \Delta_j^s \le \xi$. For clarity, we temporarily use $\dot{\mathbf{F}}$ to represent the rebuilt frequencies before smoothing derived in line 5 in Algorithm 3.

Table 2: $|\hat{f}_j - \bar{f}_j|$ in Different Cases

$I_j^{(t_1-1)}$	$I_{j-1}^{(t_1-1)}$	$I_{j+1}^{(t_1-1)}$	$\lvert\hat{f}_j - \bar{f}_j\rvert$
1	1	1	$\lvert\frac{2}{3}\chi_j - \frac{1}{3}\chi_l - \frac{1}{3}\chi_r - \frac{1}{3}\Delta_l^s + \frac{1}{3}\Delta_j^s\rvert$
	0	1	$\lvert\frac{1}{2}\chi_j - \frac{1}{6}\chi_l - \frac{1}{3}\chi_r - \frac{1}{6}\sum_{z=l}^{j-1}\Delta_z^s + \frac{1}{3}\Delta_j^s\rvert$
	1	0	$\lvert\frac{1}{2}\chi_j - \frac{1}{3}\chi_l - \frac{1}{6}\chi_r - \frac{1}{3}\Delta_l^s + \frac{1}{6}\sum_{z=j}^{r-1}\Delta_z^s\rvert$
	0	0	$\lvert\frac{1}{3}\chi_j - \frac{1}{6}\chi_l - \frac{1}{6}\chi_r - \frac{1}{6}\sum_{z=l}^{j-1}\Delta_z + \frac{1}{6}\sum_{z=j}^{r-1}\Delta_z^s\rvert$
0	1	1	$\lvert\chi_j - \frac{1}{2}\chi_l - \frac{1}{2}\chi_r - \Delta_l^s + \Delta_j^s\rvert$
	0	1	$\lvert\chi_j - \frac{1}{3}\chi_l - \frac{2}{3}\chi_r + \frac{2}{3}\Delta_j^s - \frac{1}{3}\sum_{z=l}^{j-1}\Delta_z^s\rvert$
	1	0	$\lvert\chi_j - \frac{2}{3}\chi_l - \frac{1}{3}\chi_r - \frac{2}{3}\Delta_l^s + \frac{1}{3}\sum_{z=j}^{r-1}\Delta_z^s\rvert$
	0	0	$\lvert\chi_j - \frac{1}{2}\chi_l - \frac{1}{2}\chi_r - \frac{1}{2}\sum_{z=l}^{j-1}\Delta_z^s + \frac{1}{2}\sum_{z=j}^{r-1}\Delta_z^s\rvert$

- When bin j is labeled as benign, i.e., $I_j^{(t_1-1)} = 1$ ($t_1 > 1$), its rebuilt frequency $\dot{f}_j$ in the t_1 round is still equal to $\hat{f}_j$. Then, $|\hat{f}_j - \bar{f}_j| = |\hat{f}_j - (\dot{f}_{j-1} + \dot{f}_{j+1} + \dot{f}_j)/3| = |2\hat{f}_j - \dot{f}_{j-1} - \dot{f}_{j+1}|/3$.
  If both $I_{j-1}^{(t_1-1)}$ and $I_{j+1}^{(t_1-1)}$ are 1s, i.e., $l = j-1$ and $r = j+1$, then

$$\begin{aligned}|\hat{f}_j - \bar{f}_j| &= |2\hat{f}_j - \hat{f}_{j-1} - \hat{f}_{j+1}|/3 \\ &= |\frac{2}{3}(f_j + \chi_j) - \frac{1}{3}(f_j + \Delta_{j-1}^s + \chi_{j-1}) - \frac{1}{3}(f_j - \Delta_j^s + \chi_{j+1})| \\ &= |\frac{2}{3}\chi_j - \frac{1}{3}\chi_{j-1} - \frac{1}{3}\chi_{j+1} - \frac{1}{3}\Delta_{j-1}^s + \frac{1}{3}\Delta_j^s| \\ &= |\frac{2}{3}\chi_j - \frac{1}{3}\chi_l - \frac{1}{3}\chi_r - \frac{1}{3}\Delta_l^s + \frac{1}{3}\Delta_j^s|.\end{aligned}$$

  If $I_{j-1}^{(t_1-1)} = 0$ and $I_{j+1}^{(t_1-1)} = 1$, i.e., $r = j+1$, then

$$\begin{aligned}|\hat{f}_j - \bar{f}_j| &= |2\hat{f}_j - (\hat{f}_l + \hat{f}_j)/2 - \hat{f}_{j+1}|/3 \\ &= |\frac{1}{2}(f_j + \chi_j) - \frac{1}{6}(f_j + \sum_{z=l}^{j-1}\Delta_z^s + \chi_l) - \frac{1}{3}(f_j - \Delta_j^s + \chi_{j+1})| \\ &= |\frac{1}{2}\chi_j - \frac{1}{6}\chi_l - \frac{1}{3}\chi_{j+1} - \frac{1}{6}\sum_{z=l}^{j-1}\Delta_z^s + \frac{1}{3}\Delta_j^s| \\ &= |\frac{1}{2}\chi_j - \frac{1}{6}\chi_l - \frac{1}{3}\chi_r - \frac{1}{6}\sum_{z=l}^{j-1}\Delta_z^s + \frac{1}{3}\Delta_j^s|.\end{aligned}$$

  Similarly, if $I_{j-1}^{(t_1-1)} = 1$ and $I_{j+1}^{(t_1-1)} = 0$, i.e., $l = j-1$, then

$$|\hat{f}_j - \bar{f}_j| = |\frac{1}{2}\chi_j - \frac{1}{3}\chi_{j-1} - \frac{1}{6}\chi_r - \frac{1}{3}\Delta_{j-1}^s + \frac{1}{6}\sum_{z=j}^{r-1}\Delta_z^s|$$

$$= |\frac{1}{2}\chi_j - \frac{1}{3}\chi_l - \frac{1}{6}\chi_r - \frac{1}{3}\Delta_l^s + \frac{1}{6}\sum_{z=j}^{r-1}\Delta_z^s|.$$

If both $I_{j-1}^{(t_1-1)}$ and $I_{j+1}^{(t_1-1)}$ are 0s, then

$$\begin{aligned}|\hat{f}_j - \bar{f}_j| &= |2\hat{f}_j - (\hat{f}_l + \hat{f}_j)/2 - (\hat{f}_j + \hat{f}_r)/2|/3 \\ &= |\frac{1}{3}(f_j - \chi_j) - \frac{1}{6}(f_j + \sum_{z=l}^{j-1}\Delta_z^s + \chi_l) - \frac{1}{6}(f_j - \sum_{z=j}^{r-1}\Delta_z^s + \chi_r)| \\ &= |\frac{1}{3}\chi_j - \frac{1}{6}\chi_l - \frac{1}{6}\chi_r - \frac{1}{6}\sum_{z=l}^{j-1}\Delta_z^s + \frac{1}{6}\sum_{z=j}^{r-1}\Delta_z^s|.\end{aligned}$$

- When bin j is labeled as maliciou, i.e., $I_j^{(t_1-1)} = 0$ ($t_1 > 1$), its frequency in the t_1 round will be rebuilt by its benign neighbor bins, i.e., $\dot{f}_j = (\hat{f}_l + \hat{f}_r)/2$. Then, $|\hat{f}_j - \bar{f}_j| = |\hat{f}_j - (\dot{f}_{j-1} + \dot{f}_{j+1} + \dot{f}_j)/3| = |\hat{f}_j - (\dot{f}_{j-1} + \dot{f}_{j+1})/3 - (\hat{f}_l + \hat{f}_r)/6|$.
  If both $I_{j-1}^{(t_1-1)}$ and $I_{j+1}^{(t_1-1)}$ are 1s, i.e., $l = j-1$ and $r = j+1$, then

$$\begin{aligned}|\hat{f}_j - \bar{f}_j| &= |\hat{f}_j - (\hat{f}_{j-1} + \hat{f}_{j+1})/3 - (\hat{f}_{j-1} + \hat{f}_{j+1})/6| \\ &= |f_j + \chi_j - (f_j + \Delta_{j-1}^s + \chi_{j-1} + f_j - \Delta_j^s + \chi_{j+1})/2| \\ &= |\chi_j - \frac{1}{2}\chi_{j-1} - \frac{1}{2}\chi_{j+1} - \Delta_{j-1}^s + \Delta_j^s| \\ &= |\chi_j - \frac{1}{2}\chi_l - \frac{1}{2}\chi_r - \Delta_l^s + \Delta_j^s|.\end{aligned}$$

  If $I_{j-1}^{(t_1-1)} = 0$ and $I_{j+1}^{(t_1-1)} = 1$, i.e., $r = j+1$, then

$$\begin{aligned}|\hat{f}_j - \bar{f}_j| &= |\hat{f}_j - ((\hat{f}_l + \hat{f}_r)/2 + \hat{f}_{j+1})/3 - (\hat{f}_l + \hat{f}_{j+1})/6| \\ &= |f_j + \chi_j - \frac{1}{3}(f_j + \sum_{z=l}^{j-1}\Delta_z^s + \chi_l) - \frac{2}{3}(f_j - \Delta_j^s + \chi_{j+1})| \\ &= |\chi_j - \frac{1}{3}\chi_l - \frac{2}{3}\chi_{j+1} + \frac{2}{3}\Delta_j^s - \frac{1}{3}\sum_{z=l}^{j-1}\Delta_z^s| \\ &= |\chi_j - \frac{1}{3}\chi_l - \frac{2}{3}\chi_r + \frac{2}{3}\Delta_j^s - \frac{1}{3}\sum_{z=l}^{j-1}\Delta_z^s|.\end{aligned}$$

  Similarly, if $I_{j-1}^{(t_1-1)} = 1$ and $I_{j+1}^{(t_1-1)} = 0$, i.e., $l = j-1$, then

$$\begin{aligned}|\hat{f}_j - \bar{f}_j| &= |\chi_j - \frac{2}{3}\chi_{j-1} - \frac{1}{3}\chi_r - \frac{2}{3}\Delta_{j-1}^s + \frac{1}{3}\sum_{z=j}^{r-1}\Delta_z^s| \\ &= |\chi_j - \frac{2}{3}\chi_l - \frac{1}{3}\chi_r - \frac{2}{3}\Delta_l^s + \frac{1}{3}\sum_{z=j}^{r-1}\Delta_z^s|.\end{aligned}$$

  If both $I_{j-1}^{(t_1-1)}$ and $I_{j+1}^{(t_1-1)}$ are 0s, then

$$\begin{aligned}|\hat{f}_j - \bar{f}_j| &= |\hat{f}_j - ((\hat{f}_l + \hat{f}_r)/2 + (\hat{f}_l + \hat{f}_r)/2)/3 - (\hat{f}_l + \hat{f}_r)/6| \\ &= |f_j + \chi_j - \frac{1}{2}(f_j + \sum_{z=l}^{j-1}\Delta_z^s + \chi_l) - \frac{1}{2}(f + j - \sum_{z=j}^{r-1}\Delta_z^s + \chi_r)| \\ &= |\chi_j - \frac{1}{2}\chi_l - \frac{1}{2}\chi_r - \frac{1}{2}\sum_{l}^{j-1}\Delta_z^s + \frac{1}{2}\sum_{z=j}^{r-1}\Delta_z^s|.\end{aligned}$$

Let $11|\chi|_{\max}^b + d\xi \le |\chi|_{\min}^c$ and $2|\chi|_{\max}^b + \frac{d}{6}\xi \le T_\xi \le \frac{1}{6}|\chi|_{\min}^c - \frac{1}{2}|\chi|_{\max}^b - \frac{d}{18}\xi$. When l and r are true positive, $\frac{1}{3}(|\chi_j| - |\chi|_{\max}^b) - \frac{d}{18}\xi \le |\hat{f}_j - \bar{f}_j| \le |\chi_j| + |\chi|_{\max}^b + \frac{d}{6}\xi$. Because $|\chi_j| + |\chi|_{\max}^b + \frac{d}{6}\xi \le T_\xi$ for benign bin j and $\frac{1}{3}(|\chi_j| - |\chi|_{\max}^b) - \frac{d}{18}\xi \ge T_\xi$ for malicious bin j, the bin j can be labeled correctly. When one of l and r is false positive, $|\hat{f}_j - \bar{f}_j| \ge \frac{1}{6}|\chi|_{\min}^c - \frac{1}{2}|\chi|_{\max}^b - \frac{d}{18}\xi$. Because $\frac{1}{6}|\chi|_{\min}^c - \frac{1}{2}|\chi|_{\max}^b - \frac{d}{18}\xi \ge T_\xi$, we will always label bin j as malicious, leading to false negative errors.

Based on this analysis, we discover that one malicious bin can pollute at most two benign bins' tags, and two benign bins can correctly label at least one bin between them. As such, when the number of malicious bins $n_c \le \lfloor \frac{d-1}{3} \rfloor$, i.e., $3n_c + 1 \le d$, cccording to the pigeonhole principle, in a certain round t_1^*, at least one bin j^* is genuinely benign, and all others are identified as malicious. In this case, there is no false positive error but only false negative errors. In the following rounds, relying on the benign neighbor bins l and r, which are truly positive, these bins with false negative errors can be re-identified as benign gradually. The number of true benign bins will increase with iteration until all bins are labeled correctly, leading to an accurate rectified histogram with approximate values for malicious bins and estimation values for benign bins. □

A.5 Proof of Section 6.2.1

In this subsection, we solve the optimization problem in Eq.(35).

Proof. For brevity, we simplify this optimization problem for $\mathbf{W}$ in Eq.(35) with $\mathscr{D}^k = \sum_{j\in[d]}(\tilde{f}_j^k - \tilde{f})_j^2$ and $\mathscr{D}_s = \sum_{j\in[d]}(\tilde{f}_j - (\tilde{f}_{j-1} + \tilde{f}_j + \tilde{f}_{j+1})/3)^2$.

$$\min \sum_{k\in[\kappa]} w_k\mathscr{D}^k + \sum_{k\in[\kappa]} w_k\mathscr{D}_s, \quad s.t. \sum_{k\in[\kappa]} \exp(-w_k) = 1 \tag{35}$$

Firstly, we construct the Lagrange function $\mathcal{L}(\mathbf{W})$.

$$\mathcal{L}(\mathbf{W}) = \sum_{k\in[\kappa]} w_k\mathscr{D}^k + \sum_{k\in[\kappa]} w_k\mathscr{D}_s + \lambda(\sum_{k\in[\kappa]} \exp(-w_k) - 1)$$

Then, we can get the optimal solution by solving the following equations.

$$\begin{cases}\frac{\partial\mathcal{L}(\mathbf{W})}{\partial w_k} = \mathscr{D}^k + \mathscr{D}_s - \lambda\exp(-w_k) = 0 \\ \sum_{k\in[\kappa]}\exp(-w_k) = 1\end{cases}$$

And the optimal solution is $w_k = -\ln\frac{\mathscr{D}^k + \mathscr{D}_s}{\sum_{k\in[\kappa]}\mathscr{D}^k + \kappa\mathscr{D}_s}$. □

A.6 Proof of Section 6.2.2

In this subsection, we solve the optimization problem in Eq.(21).

Proof. Fixing $\tilde{f}_j^{(t-1)}$, Eq.(21) is convex for $\tilde{f}_j^{(t)}$.

Firstly, we construct the Lagrange function $\mathcal{L}(\tilde{\mathbf{F}}^{(t)})$.

$$\begin{aligned}\mathcal{L}(\tilde{\mathbf{F}}^{(t)}) = &\sum_{k\in[\kappa]} w_k \sum_{j\in[d]}(\tilde{f}_j^{(t)} - \tilde{f}_j^k)^2 + \beta\sum_{j\in[d]}(\tilde{f}_j^{(t)} - S(\tilde{f}_j^{(t-1)}))^2 \\ &+ \lambda(\sum_{j\in[d]}\tilde{f}_j^{(t)} - 1) + \sum_{j\in[d]}\mu_j(-\tilde{f}_k^{(t)})\end{aligned}$$

Then, we can get KKT conditions as follows.

$$\begin{cases} \frac{\partial \mathcal{L}(\tilde{\mathbf{F}}^{(t)})}{\partial \tilde{f}_j^{(t)}} \\ = \sum_{k\in[\kappa]} 2w_k(\tilde{f}_j^{(t)} - \bar{f}_j^k) + 2\beta(\tilde{f}_j^{(t)} - S(\tilde{f}_j^{(t-1)})) + \lambda - \mu_j \\ = 0 & (37a) \\ \sum_{j\in[d]} \tilde{f}_j^{(t)} = 1 & (37b) \\ \forall j \in [d], \tilde{f}_j^{(t)} \geq 0 & (37c) \\ \forall j \in [d], \mu_j(-\tilde{f}_k^{(t)}) = 0 & (37d) \\ \forall j \in [d], \mu_j \geq 0 & (37e) \end{cases}$$

Solving Eq.(37a), we get

$$\tilde{f}_j^{(t)} = \frac{\sum_{k\in[\kappa]} w_k \bar{f}_j^k + \beta S(\tilde{f}_j^{(t-1)})}{\sum_{k\in[\kappa]} w_k + \beta} + \frac{\mu_j - \lambda}{2(\sum_{k\in[\kappa]} w_k + \beta)}.$$

Let $[d] = D_0 \cup D_+$. When $j \in D_0$, $\tilde{f}_j^{(t)} = 0$, we have

$$\mu_j = 2(\sum_{k\in[\kappa]} w_k(-\bar{f}_j^k) + \beta(-S(\tilde{f}_j^{(t-1)}))) + \lambda > 0.$$

When $j \in D_+$, $\mu_j = 0$, we have

$$\tilde{f}_j^{(t)} = \frac{\sum_{k\in[\kappa]} w_k \bar{f}_j^k + \beta S(\tilde{f}_j^{(t-1)})}{\sum_{k\in[\kappa]} w_k + \beta} + \frac{-\lambda}{2(\sum_{k\in[\kappa]} w_k + \beta)} > 0.$$

According to Eq.(37b), we have $\sum_{j\in D_+} \tilde{f}_j^{(t)} = 1$, then

$$\lambda = \frac{2\sum_{j\in D_+}(\sum_{k\in[\kappa} w_k \bar{f}_j^k + \beta S(\tilde{f}_j^{(t-1)})) - 2(\sum_{k\in[\kappa]} w_k + \beta)}{|D_+|}.$$

Therefore, for $j \in D_+$,

$$\tilde{f}_j^{(t)} = \frac{\sum_{k\in[\kappa]} w_k \bar{f}_j^k + \beta S(\tilde{f}_j^{(t-1)})}{\sum_{k\in[\kappa]} w_k + \beta} + \frac{1 - \sum_{j\in D_+} \frac{\sum_{k\in[\kappa]} w_k \bar{f}_j^k + \beta S(\tilde{f}_j^{(t-1)})}{\sum_{k\in[\kappa]} w_k + \beta}}{|D_+|}.$$

Since we don't know the exact D_+, at the start, we can assume $D_+ = [d]$ and $D_0 = \phi$ and compute $\tilde{\mathbf{F}}^{(t)}$. Then, we add j with $\tilde{f}_j^{(t)} < 0$ to D_0 and recompute $\tilde{\mathbf{F}}^{(t)}$ iteratively until $\tilde{\mathbf{F}}^{(t)}$ satisfies constraints in Eq.(23). Here, for j updated to D_0, its original misclassification about $j \in D_+$ causes $\mu_j = 0$. When we set its negative frequency $\tilde{f}_j^{(t)}$ to 0, the corresponding updated λ increases, leading to $\mu_j > 0$. Following the iteration, we can find the optimal solution satisfies KKT conditions. □