A STUDY ON LOCAL DIFFERENTIAL PRIVACY UNDER ADVERSE CIRCUMSTANCES

RONG DU

PhD

The Hong Kong Polytechnic University

2025

The Hong Kong Polytechnic University
Department of Electrical and Electronic Engineering

A Study on Local Differential Privacy under Adverse Circumstances

Rong Du

A thesis submitted in partial fulfillment of the requirements for
the degree of Doctor of Philosophy

July 2024

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgment has been made in the text.

Signature: ____________________

Name of Student: Rong Du

Abstract

The rapid generation of information in the era of big data has made its analysis and the application of effective strategies increasingly essential across various fields, including business [97], healthcare [33], education [49], transportation [108], and public administration [66]. One method that has proven its immense potential for information gathering is crowdsourcing. However, the convenience of data collection through crowdsourcing also brings significant privacy concerns, particularly under adverse circumstances.

Recent years have witnessed numerous data breach incidents, highlighting the vulnerability of personal information in centralized databases. Notable examples include the Yahoo breaches in 2013 and 2014 affecting 3 billion users [4], the Facebook-Cambridge Analytica scandal impacting over 50 million users [2], the Equifax leak compromising 143 million consumers' data [6], and the Marriott International hotels data breach affecting up to 500 million guests [3]. These incidents underscore the pressing need for robust privacy-preserving mechanisms, especially in adverse data collection environments.

However, LDP faces significant challenges under adverse circumstances, particularly in three key areas: i) The curse of high dimensionality, which compromises aggregation accuracy. ii) Inefficient processing of sparse data with low-frequency values. iii) Vulnerability to Byzantine attacks that introduce poisoned data.

This thesis presents a comprehensive study on enhancing LDP under adverse condi-

tions, making the following contributions: i) We optimize privacy budget allocation among correlated attributes to improve utility in high-dimensional data scenarios. ii) For sparse data, we develop a novel approach using budget allocation and reinforcement learning to identify top-k values efficiently. iii) To combat Byzantine attacks, we establish robust LDP protocols that filter out poisoned data by analyzing varying user behaviors.

Our research advances the field of secure and efficient data analytics under LDP by introducing innovative privacy-preserving mechanisms designed to perform effectively in challenging environments. This study not only addresses current limitations but also provides a foundation for future research in improving LDP's resilience and applicability under adverse circumstances.

Publications Arising from the Thesis

1. R. Du, Q. Ye, , Y. Fu, H. Hu. "Distribution estimation under LDP against arbitrarily distributed attacks", submitted to *IEEE Transactions on Dependable and Secure Computing*, 2024.

2. R. Du, Q. Ye, Y. Fu, H. Hu and K. Huang. "Top-k Discovery under Local Differential Privacy: An Adaptive Sampling Approach", *IEEE Transactions on Dependable and Secure Computing*, 2024.

3. Y. Fu, Q. Ye, R. Du, and H. Hu. "Interactive Trimming against Evasive Online Data Manipulation Attacks: A Game-Theoretic Approach. In *IEEE International Conference on Data Engineering (ICDE)*, May 13-17, 2024, Netherlands. https://doi.org/10.48550/ARXIV.2403.10313

4. Y. Fu, Q. Ye, R. Du, and H. Hu. "Collecting Multi-type and Correlation-Constrained Streaming Sensor Data with Local Differential Privacy", *ACM Transactions on Sensor Networks*. https://doi.org/10.1145/3623637

5. R. Du, Q. Ye, Y. Fu, and H. Hu. "Collecting High-Dimensional and Correlation-Constrained Data with Local Differential Privacy", *Proc. of 18th IEEE International Conference on Sensing, Communication, and Networking (SECON)*, 2021. https://doi.org/10.1109/SECON52354.2021.9491591

6. R. Du, Q. Ye, Y. Fu, H. Hu, J. Li, C. Fang, and J. Shi. “Differential Aggregation against General Colluding Attackers” *IEEE International Conference on Data Engineering (ICDE)*, April 37, 2023, Anaheim, California. https://doi.org/10.1109/ICDE55515.2023.00169

7. Y. Fu, M. H. Au, R. Du, H. Hu, and D. Li. “Cloud Password Shield: A Secure Cloud-based Firewall against DDoS on Authentication Servers”, *(poster) Proc. of 40th IEEE International Conference on Distributed Computing Systems (ICDCS)* , July 8 10, 2020, Singapore. https://doi.org/10.1109/ICDCS47774.2020.00154

8. Y. Fu, Q. Ye, R. Du, and H. Hu. “Unified Proof of Work: Delegating and Solving Customized Computationally Bounded Problems in A Privacy-preserving Way”, *The 6th APWeb-WAIM International Joint Conference on Web and Big Data (APWeb-WAIM)*, August 11-13, 2022, China. https://doi.org/10.1007/978-3-031-25201-3_24.

9. R. Du, Q. Ye, Y. Fu, and H. Hu. “Privacy for Free: Leveraging Local Differential Privacy Perturbed Data from Multiple Services.”, VLDB2025, under revision.

10. R. Du, Q. Ye, Y. Fu, and H. Hu. “Leveraging Historical Perturbation for Data Stream Publication under Local Differential Privacy.”, Submitted to ICDE2025, under revision.

Acknowledgments

Looking back at my PhD studies, I realize that the experience has been both challenging and rewarding. As I approach the completion of my thesis, I would like to express my sincere gratitude and respect to the following individuals.

Firstly, I would like to express my heartfelt gratitude to my supervisor, Prof. Haibo Hu. He not only gave me this invaluable opportunity to pursue my doctoral studies but also offered me professional and systematic academic training. As a mentor, he has taught me how to conduct research properly and maintain rigor. His academic insights have guided me, making me realize the significance of my work and fostering my genuine love for research. Prof. Hu, who is diligent and dedicated, is a model for me. What I have learned from him has profoundly impacted me, and I hope to pass these lessons on to my students in the future.

I would also like to thank Dr. Qingqing Ye, who not only guided me academically but also provided emotional support. Whenever I was unsure about how to continue my studies, Dr. Ye could always keenly perceive my state of mind, proactively engaging in discussions with me and offering invaluable advice with a gentle demeanor. Dr. Ye is not only an excellent advisor but also a wonderful collaborator.

Additionally, I owe my thanks to every kind and intelligent colleague from the Astaple Laboratory, who offered me immense help. I still remember vividly, when I invited everyone to attend the rehearsal for my conference presentation, a room full of colleagues came to support me, which profoundly moved me. The proactive questions

and discussions during our group meetings have been incredibly beneficial to me. I am fortunate to be a part of this team, characterized by its passion for scientific pursuit and its spirit of kindness and assistance.

During emotional struggles and moments of anxiety, my dear friends Ms. Lina Zhou, Mr. Yu Zheng, Miss Yao Wang, Miss Duo Xu, Miss Yarong Wen and Miss Bowen Wang were always there for me. They discussed life and personal growth with me, helping me to overcome lifes challenges. I am deeply grateful for their companionship.

I also wish to thank my husband, Mr. Yue Fu, who is both my intimate partner and my work colleague. I am grateful for his support in my personal life and his mathematical insights in my academic work. I look forward to overcoming challenges and exploring the scientific world with him in the future.

Lastly, I would like to sincerely express my gratitude to the Hong Kong Polytechnic University for providing the invaluable resources and opportunities for my studies. I am also deeply thankful to my family and to every kind-hearted individual who has offered their assistance in my life.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Local differential privacy (LDP) has become popular for protecting personal data privacy, particularly in big data analysis. It has already been deployed in many real-life data collection systems, including Google Chrome [45, 48], iOS [109], and Windows 10 [35]. By perturbing data through mechanisms such as Laplace or the LDP frequency oracle protocol [17,28,39,64,114], LDP enables the data collector to aggregate statistical values (e.g., mean or frequency) using methods such as likelihood estimation [45] and regression [92], while providing deniability for users' private data. Users perturb their private data using mechanisms like Laplace noise addition or randomized response [43, 61] before transmission, enabling population statistics estimation without revealing individual accurate data.

In the era of big data, privacy-preserving technologies are essential to collect sensitive data and analyse their statistical features (e.g., frequency and mean) while preserving individual's data privacy. In an LDP protocol, users only provide perturbed data to collectors, who then estimate some statistics, e.g., mean and frequency, the two most fundamental from these data [8, 16, 41, 71, 84, 110, 114, 124]. However, almost all LDP works assume that users are at least semi-honest, that is, they honestly perturb and send data to the collector according to the protocol. Unfortunately, this assumption rarely holds in real-world scenarios — any large-scale data collection system cannot rule out the existence of Byzantine users [14, 87, 89, 90], who are malicious users that can collude among themselves to send fake values and influence the estimated statistics in their favor. In particular, the estimated mean has become a popular target in such attacks. For instance, Byzantine users have engaged in product rating fraud for e-commerce sellers to boost their sales [58, 76, 79, 102]. The New York Times reported businesses hiring workers on Mechanical Turk, an Amazon-owned crowdsourcing marketplace, to post fake 5-star Yelp reviews on their businesses [76].

The implementation of LDP faces significant challenges, particularly under adverse circumstances. This thesis focuses on three main aspects of these challenges:

1.1 The Utility of LDP in Complex Data Environments

The utility of LDP-protected data becomes critically important as data complexity and volume increase, especially in adverse scenarios. High-dimensional data, often derived from multiple sensors or attributes, pose unique challenges. For instance, datasets like the Wisconsin breast cancer diagnostic dataset [1] have over 30 attributes, with common correlations like temperature and humidity in a workspace. Ensuring ϵ-LDP for all attributes in such complex environments often leads to substantial utility loss due to the noise introduced by privacy mechanisms.

Another significant challenge lies in the problem of top-k item discovery under LDP, particularly in sparse data environments. Traditional LDP mechanisms struggle to achieve satisfactory performance in these scenarios due to users' varying numbers of items and vast item domains. Solutions like LDPMiner [88] and SVIM [115] attempt to address these issues but still face challenges in accurately determining padding lengths and avoiding biased frequency estimation.

1.2 The Byzantine Security of LDP

The security of LDP systems becomes particularly crucial in the presence of malicious users, a common occurrence in adverse data collection circumstances. These Byzantine users may attempt to manipulate statistical estimations by submitting adversarial data, potentially colluding and employing opportunistic strategies. The nature of LDP makes it challenging to identify such malicious actors, as the perturbation process provides plausible deniability for adversarial submissions.

As the privacy protection level increases (i.e., as ϵ becomes smaller), the potential impact of Byzantine users grows. They can inject adversarial data over a broader domain, potentially leading to disproportionately large estimation errors - a phenomenon known as the long-tail attack [135]. Traditional statistical methods for outlier removal, such as trimming, face limitations in the LDP context due to the lack of prior knowledge about data distributions and the risk of introducing bias by removing legitimate perturbed data.

1.3 Contributions and Thesis Organization

This thesis presents novel solutions to enhance both the utility and security of LDP systems under adverse circumstances. Our contributions, as presented in each chapter, are as follows:

Chapter 2: A comprehensive review of relevant literature, covering existing LDP protocols, reinforcement learning techniques (focusing on multi-armed bandits), and studies on Byzantine Attacks in privacy-preserving data collection.

Chapter 3: Introduction of fundamental concepts and preliminaries used throughout the thesis.

Chapter 4: Proposal of a relaxed LDP model, univariate dominance local differential privacy (UDLDP), designed for high-dimensional data in adverse environments. We introduce correlation-bounded perturbation (CBP) and its extension with sampling (CBPS) to optimize privacy budget allocation among correlated attributes.

Chapter 5: Development of adaptive sampling schemes (ARBS and ARBSF) based on multi-armed bandits for efficient top-k item discovery and frequency estimation in sparse data scenarios under LDP. We also present an optimization technique to reduce time complexity from $O(n)$ to $O(1)$.

Chapter 6: Introduction of a novel approach to mitigate Byzantine attacks in LDP systems. We present the multi-group Differential Aggregation Protocol (DAP), incorporating the Segmented Expectation-Maximization Filter (SEMF) and its extension, Distribution Estimation-EMF (DE-EMF), to enhance distribution estimation accuracy under adversarial conditions.

Chapter 7: Summary of the thesis outcomes and proposal of future research directions for improving LDP utility and security under adverse circumstances.

This thesis aims to advance the field of LDP by addressing the critical challenges posed by adverse data collection environments, thereby enhancing both the utility and security of privacy-preserving data analysis systems.

Chapter 2

Literature Review

2.1 Differential Privacy

Differential privacy (DP) [42], [43], [80] is a mathematical approach to quantizing privacy protection, typically through the appropriate use of Laplace [43], Gaussian [72], or geometric distributions [54] to randomize the results of statistical queries in interactive query-response systems. DP has been extensively studied in various fields, including theory analyses [17, 39, 64], data publication [28, 133], data publication [28, 133], machine learning [16], and systems [21]. DP works on the assumption that a trusted third-party server is used, but this is regarded as impractical in privacy aware crowdsourced systems.

2.2 Local Differential Privacy

Local Differential Privacy (LDP) [29, 39, 42, 43, 64, 80], a variant of DP, is a technique introduced in 1965, aiming to provide privacy guarantees for individual users in distributed systems, particularly in untrusted environments. The inception of this concept was marked by the proposal of the randomized response (RR) model [117], a simple perturbation technique by Warner.

Since its proposal, LDP has been extensively studied, applied, and successfully deployed in various industries and research fields. Google's Chrome was the first to deploy an extension of LDP known as RAPPOR [45], making it the first client-based practical privacy solution. This was followed by its implementation in Apple's iOS [105,109], Microsoft's Windows 10 [35], and research conducted by Samsung [84].

LDP has been widely applied in multiple areas, including but not limited to multi-attribute values estimation [37, 92], marginal release [31, 134], time series data release [127, 128], graph data collection [103, 125, 126], key-value data analysis [56, 129, 130], and private learning (machine learning) [136, 137].

2.2.1 Frequency and Mean Estimation for LDP

Mean and frequency estimations are commonly seen in LDP scenarios. Kairouz et al. [62] proposed k-RR, which is designed to be adaptive to a wider range of values than RR. Google, through the work of Erlingsson et al. [45], developed and implemented RAPPOR in Chrome. This system encodes user data into a Bloom filter and applies a randomized response to each bit of the filter, leading to more accurate decoding outcomes. However, due to the restriction on the false positive rate of the Bloom filter, it is necessarily sparse, leading to high communication costs.

To address these communication costs, Bassily and Smith [17] proposed a 1-bit protocol for frequency estimation. The parameter results were further improved with the OUE protocol proposed by Wang et al., which is significantly more accurate. They also designed the OLH protocol, which uses local hashing to provide better utility, effectively reducing the communication cost while maintaining the same variance as OLH.

Wang et al. [114] optimized the parameters of the basic RAPPOR by minimizing the variance in frequency estimation. Many studies have focused on complex data types and analysis tasks under LDP. For instance, Bassily and Smith proposed an asymptotically optimal solution for building succinct histograms over a large categorical domain under LDP, and Qin et al. [88] proposed a two-phase approach named LDP-Miner for estimating heavy hitters (items frequently possessed by users) in set-valued data with LDP, wherein each user can have any subset of an item domain of varying lengths.

Several mechanisms [8, 16, 35, 71] have also been proposed for frequency estimation under LDP. Among them, the most relevant work for estimating numerical data by using the EM algorithm in LDP is the SW mechanism [71]. However, the SW mechanism is not designed to combat Byzantine attacks and therefore cannot eliminate the impact of poison values. For mean estimation, Duchi et al. [41] propose a 1-bit mechanism, and Wang et al. [110] propose the Piecewise Mechanism, which are the state-of-the-art methods.

2.2.2 Multiple Attributes Collection for LDP

In addition to single attributes, a large amount of work has also studied the values of multiple attributes. The Lopub developed by Ren et al. [92] focuses on high-dimensional crowdsourced data publication. However, the communication costs of this approach are relative high which send a perturbed Bloom Filter. Du et al. [37] utilize correlations among attributes and design a more relaxed definition of LDP to achieve better utility. Piecewise mechanism [110] leverages probability density functions to perturb input values, thereby enhancing the precision of results, and subsequently utilizes OLH to collect multi-attribute values.

The most similar topic to the collection of multi-attribute values is the collection of set-value data. For the former, each user has d attributes, each attribute corresponding to a value. For the latter, each user has a private set containing a subset of the

d attributes, where the elements in the set are the attributes themselves rather than their values. The two most relevant works are LDPminer [88] and PSFO [115]. The goal of both methods is to identify heavy hitters and estimate the frequency of their corresponding values in extremely sparse set-value data. LDPMiner is a frequency publishing method targeted at heavy hitter queries. First, data are collected and the collector determines the heavy hitter set and returns it to the user. The user then sends data corresponding to some of the items in the set to the data collector. Based on the LDPMiner, Wang et al. [115] examined the same problem and proposed a more efficient framework PSFO to estimate both the frequent items and the frequent itemsets. In general, PSFO schemes employ padding and sampling techniques to mitigate the variance resulting from large domains. However, these techniques introduce bias, which has a padding length smaller than the length of the user's private set. Furthermore, the frequency estimation performs poorly due to the significant variance when the padding length exceeds that of most value sets. Hence, these methods may not be optimal for accurate frequency estimation.

2.3 Multi-armed Bandits

The classical MAB problem is a formulation of the exploration and exploitation dilemma inherent in reinforcement learning [22]. The MAB problem [19] , which involves decision-making under uncertainty, has been extensively studied for decades. MAB problem has found applications in numerous fields. such as online advertising [68], clinical trials [18], networking [22, 23], and pairwise ranking [9].

Most MAB studies focused on either (i) minimizing the regret (e.g., [12], [13], [53], [22], [10], [74]) through a tradeoff between the exploration and exploitation of arms or researching pure exploration problems, which are aimed at identifying one, or (ii) pure exploration problems (e.g., [10, 12, 13, 13, 22, 53, 74]) where minimizing the number of samples taken or identifying one or multiple best possible arms while satisfying specific conditions (e.g., within a fixed number of samples or with the least cost). This study focus on exploring the best reward with the given cost, which indicates that the problem is of the pure exploration type. There are already quite a few papers [37, 52, 63, 91, 104] that use MAB methods under the LDP system to achieve higher privacy protection. This thesis presents how Multi-Armed Bandit (MAB) methods can be employed to enhance the performance of the existing LDP perturbation protocol, RR, in achieving better utility for top-k estimation on set-value data.

LDP is a newly emerged technique to provide individual privacy guarantees for distributed users. In 1965, the concept of local privacy was firstly studied, and the randomized response (RR) model was firstly proposed by Warner [61]. To optimize the performance of perturbation algorithms, Kairouz et al. [62] introduced k-RR, which is adaptive to a universe of information-theoretic utility functions.

A fundamental goal of LDP functionality is frequency estimation. RAPPOR [45], which was proposed and well-employed into Chrome by Google, encodes users' data into a Bloom filter and then performs RR on each bit of Bloom filter, which enables the decoding result more accurate. However, the false positive rate of Bloom filter shall be restricted, thus the Bloom filter is necessary to be sparse, which renders the communication cost unsatisfactory. Bassily and Smith [17] proposed a 1-bit protocol for frequency estimation to optimize the communication cost. However, the data utility is still unsatisfactory. The parameter results are further optimized in the Optimized Unary Encoding (OUE) [45] protocol, which achieves significantly better accuracy. This literature also designed an OLH protocol, which provides much better accuracy, but still requires an $O(logn)$ communication cost. Note that all of the above methods focus on LDP on a single attribute. Another interesting problem in LDP is mean estimation over numerical data, which have been widely studied in literature [39] [40].

LDPMiner [16] is a frequency publishing method, which is targeted at heavy hitter queries. Firstly, the data collector collects data and determines the Heavy Hitter set, and returns it to the user. Then, the user sends data corresponding to some of the items in the set to the data collector. Ren et al. [92] develop the Lopub, which is focused on high-dimensional crowdsourced data publication. However, the communication cost of this approach is very high, since the transmission of every attribute is the size of a Bloom filter. PM and HM mechanisms [110] perturb the input value into a probability density function to get a better result accuracy.

2.4 Byzantine Attacks

The problems of the Byzantine attacks, that is, data poisoning attacks have recently been studied in many fields, such as crowdsourcing and crowdsensing scenarios [26,51], applications of Internet of Things [59,93], electric power grids [75] and machine learning algorithms [25, 46, 47]. However, combating Byzantine attacks in LDP protocols is a relatively new topic that has few state-of-the-art papers. Literature [30] figures out that LDP is vulnerable to manipulation attacks. With a small privacy budget or a large input domain, a few poisoned values can completely ruin the real distribution. To combat this kind of attack, sampling is an easy but effective approach. Literature [24] formulates the data poisoning attack as an optimization problem and proposes three attacking patterns to maximize their attacking effectiveness, and design some countermeasures accordingly. Literature [120] is the first attempt at poisoning attacks for key-value data in LDP protocols. They formulate an attack with two objectives, which are to simultaneously maximize the frequencies and mean values and to design two countermeasures against this attack. Literature [65] proposes a novel verifiable LDP protocol based on Multi-Party Computation (MPC) techniques. They propose a verifiable randomization mechanism in which the data collector can verify the completeness of executing an agreed randomization mechanism for every data provider. However, this method is only proposed for the categorical frequency oracles, such as kRR [62], OUE [114] and OLH [114] instead of mean and distribution estimation on numerical values. Recent research [100] on LDP protocols for frequency estimation presents two verifiable LDP protocols: VGRR and VOUE. These methods, however, necessitate that users provide the aggregator with additional information about the original data. In contrast, our proposed scheme eliminates the need for such extra information from users.

Chapter 3

Preliminary

3.1 Local Differential Privacy

An assumption of differential privacy (DP) is the existence of a trusted server, which is usually impractical in privacy-aware crowdsourced systems. To deal with this problem, local differential privacy (LDP) is proposed recently to provide a stringent privacy guarantee for crowdsourced systems when data contributors trust nobody but themselves. A mechanism $\mathcal{M}$ satisfies with ϵ-local differential privacy (ϵ-LDP), where $\epsilon \geq 0$, if and only if for any two data records S_1, S_2 and any possible output $T \in \text{Range}(\mathcal{M})$, the following condition holds:

$$\frac{P[\mathcal{M}(S_1) = T]}{P[\mathcal{M}(S_2) = T]} \leq e^{\epsilon} \tag{3.1}$$

This is a formal definition of LDP, where $\mathcal{M}$ is a non-deterministic perturbation algorithm that maps a certain input to an output with certain probability. The set of all possible outputs is called the value range of this perturbation algorithm. Since $P[\mathcal{M}(S_1) = T]$ is very close to $P[\mathcal{M}(S_2) = T]$, an adversary can not determine any individual's true answer from observation of their outputs.

3.2 Local Differential Privacy for High-dimensional Data

Although the definition on scalar data is clear, extension to high-dimensional cases remains non-trivial. A straightforward definition is to directly put the high-dimensional data into Equ. 3.1. Formally, let $X = \{b_1, b_2, ...b_n\}$ and $X' = \{{b_1}', {b_2}', ...{b_n}'\}$ denote any two records X and X' from the crowd-sourced dataset. A mechanism $\mathcal{M}$ satisfies with ϵ-LDP, if the following condition holds for any possible output $Y \in \text{Range}(\mathcal{M})$:

$$\begin{aligned}&\frac{Max(P[\mathcal{M}(X) = Y])}{Min(P[\mathcal{M}(X') = Y])}\\ =&\frac{Max(P[\mathcal{M}(b_1, ..., b_n) = Y])}{Min(P[\mathcal{M}(b_1', ..., b_n') = Y])}\\ \leq& e^{\epsilon}\end{aligned} \tag{3.2}$$

When each attribute b_n has binary value, for any two data records $B_1 = \{b_1 = 0, ..., b_h = 0, ..., b_k = 0\}$ and $B_2 = \{b_1 = 1, ..., b_h = 1, ..., b_k = 1\}$, the following inequality is required to be held for any given B_1, B_2 and output B':

$$\frac{P[\mathcal{M}(B_1) = B']}{P[\mathcal{M}(B_2) = B']} \leq e^{\epsilon} \tag{3.3}$$

From this definition, intuitively, an adversary seeing “00000” cannot tell whether the input is “00000” or “11111” due to the privacy provided by e^{ϵ}.

3.3 Randomized Response

Randomized response (RR) [117], which has been widely used in the “Yes or No” sensitive problem, is the most straightforward perturbation algorithm to guarantee LDP. Before answering the data collector, users flip a biased coin, send true answers with a certain probability q, and false answers with probability $1-q$. Due to the existing randomness, the data collector cannot tell the true value of any user. Particularly, for the binary case, a most widely used randomized response algorithm is given as follows:

$$b' = \begin{cases} b, & w.p.\ q \\ 1-b & w.p.\ 1-q \end{cases}$$

3.4 Multi-armed Bandit

The MAB (or bandits) is a powerful hypothesis generation and exploration technique. It can be considered as a collection of arms with real Bernoulli distributions $D = \{D_1, \ldots, D_n\}$. Let $\{p_1, \ldots, p_d\}$ be the mean values associated with these reward distributions. Each arm i has the value 1 with probability p_i and has the value 0 with probability $q_i = 1 - p_i$. The probability mass function f of this distribution, over a possible output z, is

$$f(z; p_i) = \begin{cases} p_i, & \text{if } z = 1 \\ q_i & \text{if } z = 0 \end{cases}$$

The gambler iteratively plays one arm per round and observes the associated reward. Finally, the decision-maker selects some arms to sample adaptively and thus maximizes her expected gain, identifying one or multiple arms that satisfy specific conditions, and minimizing the number of samples taken.

3.5 Hoeffding Bounds vs. Empirical Bernstein Bounds

Hoeffding bounds. Hoeffding's inequality, as proven by Wassily Hoeffding in 1963 [21], gives an upper bound for the likelihood that the sum of constrained independent random variables deviates from its expected value. Let $\{X_1, \ldots, X_t\}$ be real-valued i.i.d. random variables with the range R ($R = 1$ for binary data) and the mean μ, and let $\overline{X_t} = \frac{1}{t}\sum_{i=1}^{i=t} X_i$. With a probability of at least $1-\delta$, Hoeffding's inequality states that:

$$|\overline{X_t} - \mu| <= R\sqrt{\frac{log(2/\delta)}{2t}}, \tag{3.4}$$

Hoeffding's inequality has been extensively applied to online learning scenarios due to its generality. One limitation of the bound is that it scales linearly with the range R, but not with the variance of X_i. In cases where the variance bound is known and relatively small compared to the range, Bernstein's inequality is a more appropriate choice due to its better handling of small variance bounds. However, when there is little prior knowledge of variance bounds, Hoeffding's inequality is more practical.

Empirical Bernstein bounds. The empirical Bernstein bound [81] provides a tighter bound than Hoeffding's inequality, depending on the empirical standard deviation $\overline{\sigma}_t^2 = \frac{1}{t}\sum_{i=1}^{t}(X_i - \hat{x_t})^2$. The empirical Bernstein bound states that with a probability of at least $1-\delta$,

$$|Xt - \mu| \leq \overline{\sigma_t}\sqrt{\frac{2log(3/\delta)}{t}} + \frac{3Rlog(3/\delta)}{t}. \tag{3.5}$$

The rate of decline for the term associated with range R is t^{-1}, and it becomes insignificant as the variance increases. In contrast, the square root term is dependent on $\overline{\sigma_t}$ rather than R. Therefore, when $\overline{\sigma_t}$ is much smaller than R, the empirical Bernstein bound is much tighter than the Hoeffding bound.

3.6 Piecewise Mechanism (PM)

LDP has been widely adopted to estimate statistics from a large population of users. This thesis mainly focuses on state-of-the-art **Piecewise Mechanism (PM) [110]** on mean estimation of numerical values. As shown in Algorithm 1, given input value $v \in [-1, 1]$, the probability density function (PDF) of output $v' \in [-C, C]$ has two parts: domain $[l(v), r(v)]$ and domain $[-C, l(v)) \cup (r(v), C]$, where $C = \frac{e^{\epsilon/2}+1}{e^{\epsilon/2}-1}$, $l(v) = \frac{C+1}{2}v - \frac{C-1}{2}$ and $r(v) = l(v) + C - 1$. Given input v, the perturbed value is in range $[l(v), r(v)]$ with high probability and in range $[-C, l(v)) \cup (r(v), C]$ with low probability. Because value v' is an unbiased estimator of input value v, the data collector can use the mean of collected values as an unbiased estimator of the mean of input values.

Algorithm 1: Piecewise Mechanism

Input: Original value v and privacy budget ϵ

Output: Perturbed value v'

Sample x uniformly at random from [0,1]; **if** $x < \frac{e^{\epsilon/2}}{e^{\epsilon/2}+1}$ **then**

  Sample v' uniformly at random from $[l(v), r(v)]$ **else**

  Sample v' uniformly at random from $[-C, l(v)) \cup (r(v), C]$ **EndIf**

return v'

3.7 Square Wave Mechanism (SW)

LDP has been widely adopted to estimate statistics from a large population of users. This thesis mainly focuses on state-of-the-art **Square Wave Mechanism (SW) [71]** on distribution estimation of numerical values.

The main idea of SW is to increase the probability of output value that can provide more information about the input value. The data collector receives perturbed values from users and reconstructs the distribution over a discrete numerical domain. The bucketization step to discrete can be performed either before or after applying the randomization step. This chapter only describes the "bucketize before randomize" here. Each user processes a floating value in the domain [0,1] and generates a value in $[-S, 1+S]$. Assume the bucket size of input domain is d and output domain size $d' = d + 2b$ ($b = \lfloor \frac{\epsilon e^{\epsilon} - e^{\epsilon} + 1}{2e^{\epsilon}(e^{\epsilon} - 1 - \epsilon)} d \rfloor$), given an input value in bucket v, the randomized output can be expressed as following:

$$Pr[SW(v) = v'] = \begin{cases} p, & \text{if } |v - v'| \leq b, \\ q, & \text{otherwise.} \end{cases} \tag{3.6}$$

where $p = \frac{e^{\epsilon}}{(2b+1)e^{\epsilon}+d-1}$ and $q = \frac{1}{(2b+1)e^{\epsilon}+d-1}$. After reviving the perturbed data, the data collector aggregates the original distribution by using the Maximum Likelihood Estimation (MLE) [95], and reconstructs the distribution of original values. Note that SW can estimate both the mean value and the original data distribution.

3.8 Expectation Maximization

Given a set of observed values X in a statistical model, a straightforward approach to estimate an unknown parameter θ of it is to find the maximum likelihood estimation (MLE). Generally, there is a θ can be obtained by setting all first-order partial derivatives of the likelihood function l to zero and solve them. However, it is impossible to attain θ in this way where latent variables Z exist — the result will be a set of interlocking equations where the solution of θ needs the values of Z and vice versa. When one set of equations is substituted for the other, the result is an unsolvable equation. The expectation-maximization (EM) algorithm [11] can effectively find the MLE by performing expectation (E) steps and maximization (M) steps iteratively when there are latent variables.

E step produces a function $Q(\theta|\theta^t)$ that evaluates the log-likelihood expectation of θ given the current estimated parameters θ^t:

$$Q(\theta|\theta^t) = \mathbb{E}_{Z|X,\theta^t}[log(\theta; X, Z)] = \mathbb{E}_{Z|X,\theta^t}[log(X, Z|\theta)]$$

M step calculates parameters that maximize the expected log-likelihood obtained in the E step:

$$\theta^{t+1} = \arg\max_{\theta} Q(\theta|\theta^t).$$

Chapter 4

Collecting High-Dimensional and Correlation-Constrained Data with Local Differential Privacy

With the prosperity of Internet of Things and crowdsourcing, sensor readings have become an important source of data that drives new applications such as smart home, smart city, smart manufacturing, and telecare [5]. For example, during the recent COVID-19 pandemic, for a better epidemiological understanding, volunteer Hong Kong participants who are quarantined in their homes or hospitals have been wearing a device with built-in sensors on their upper arm 24 hours a day, through which data including their body temperatures, respiratory rates, blood oxygen levels, and heart rates are sent to a digital platform for real-time monitoring and analysis [107]. However, while data collected and shared between users and institutions can produce rich knowledge about the cyber-physical space, natural phenomena, and society, they also bring unprecedented privacy threats to the data providers. To address privacy concerns, local differential privacy (LDP) [28] [39] [64] is proposed as a stringent privacy guarantee for crowdsourced systems, in which sensitive data (particularly those associated with personal information) are collected in a perturbed manner to estimate their statistics, such as frequency or mean. The strength of perturbation is described and quantified by a privacy budget ϵ, which is typically acknowledged by data providers.

As both the type and complexity of sensors increase, data collected from users have been expanding with higher dimensions (or more "attributes" in the IoT terminology). For example, some popular medical datasets, such as the Wisconsin breast cancer diagnostic dataset, have over 30 attributes [15]. Further more, many of these attributes have correlations among themselves, such as the temperature and humidity in a working space. To satisfy ϵ-LDP for all attributes as a whole, this chapter must either use sampling [16] or the composition theorem of LDP [69] to partition the budget for all d dimensions. The former collects a single attribute from each user. Since the number of users is diluted by dimensions, this solution requires a large user population that is proportional to the dimensionality. The latter pessimistically assumes a complete correlation (i.e., full dependence) between any two attributes, which is equivalent to the collector repeatedly observing an attribute for d times. According to the composition theorem of LDP, each attribute must be perturbed with a smaller privacy budget (e.g., ϵ/d) so that the sum of all budgets is still ϵ. For high-dimensional data, this incurs a large amount of noise on each attribute and thus results in extremely poor data utility.

Our key observation is that the pessimistic assumption of a complete correlation between attributes is not necessary in practice, especially for those IoT data that come from different types of sensors that are intrinsically independent, for example, humidity sensor and luminance sensor in a working space. Even for those correlated attributes, the correlation often has an upper bound that can be derived from historical data or apriori knowledge. This chapte quantifies the degree of correlation and leverage it to effectively allocate the privacy budgets or sampling probabilities of all attributes. The objective is to achieve an optimized utility for statistics estimation (e.g., frequency count) in high-dimensional data. To summarize, the contributions of this chapter are three-folded.

- **Correlation Quantification.** This chapter presents a formal definition to quantify the degree of correlation between a pair of attributes from aprior knowledge or historical data. This lays the foundation of reducing perturbation in LDP for high dimensional data.
- **Univariate Dominance LDP (UDLDP).** To further address the low utility in high-dimensional LDP, this chapter develops a relaxed privacy model, namely, univariate dominance LDP, to allow the definition of LDP on a single attribute. This chapter then presents a correlation-bounded perturbation protocol (CBP) that satisfies UDLDP.
- **CBP with sampling (CBPS).** This chapter extend CBP to support sampling, a common technique in sensor networks and IoT, and present the best sampling strategy for all attributes to achieve the best data utility.

The rest of the chapter is organized as follows. Chapter 4.1 presents the problem definition. Chapter 4.2 presents the detailed CBP protocol. Chapter 4.3 studies CBP with sampling. Chapter 4.4 shows the experimental results, followed by a summary in Chapter 4.5.

4.1 Problem definition

This chapter studies the problem of privacy-preserving statistical estimation on data records from different users with n attributes. For brevity, this chapter first refrains ourselves on binary attributes (**multi-value cases** are discussed in a dedicated part in Chapter IV) and estimate the frequency of “1” for each attribute. Since these attributes have correlations among themselves, traditional LDP must use either sample users or partition the privacy budget ϵ for each attribute. Our main assumption is three-folded. First, not all attributes are correlated. From apriori knowledge and historical data, this chapter can identify attributes that are independent of the others, which form "clusters". Second, even in a cluster, the attributes are usually correlated partially, rather than completely dependent on one another. Hence, it is possible to assume an upper bound of their correlations and leverage it on privacy budget partition. Third, as this chapter studies high-dimensional data (i.e., large n), the notion of ϵ-LDP, where any two records in the high-dimensional space must not have drastic probability difference (i.e., one being very high and one being very low), becomes too stringent and sometimes unnecessary as most probabilities are negligible. To alleviate this issue, this chapter proposes a dimension-wise ϵ-LDP which dictates no two records have drastic probability difference **in each attribute**. The rest of this chapter gives the formal definitions of these notions.

4.1.1 Attribute Clusters

In practice, high-dimensional data are not necessarily related to each other. From apriori knowledge, this chapter can group all attributes into several clusters, the attributes in which are independent of those in the others. An example is shown in Fig. 4.1, which illustrates a sensor data record $(b_1, b_2, ..., b_m)$ collected from a room in a smart building system. Cluster C_1 contains three attributes from an air particle sensor, C_2 has only one attribute from a carbon monoxide concentration sensor, and C_λ has three attributes from a noxious gas sensor collecting indoor air pollution by decoration. According to domain knowledge of building systems [85] [121], attributes in different clusters, or *inter-cluster* attributes, are independent to each other.

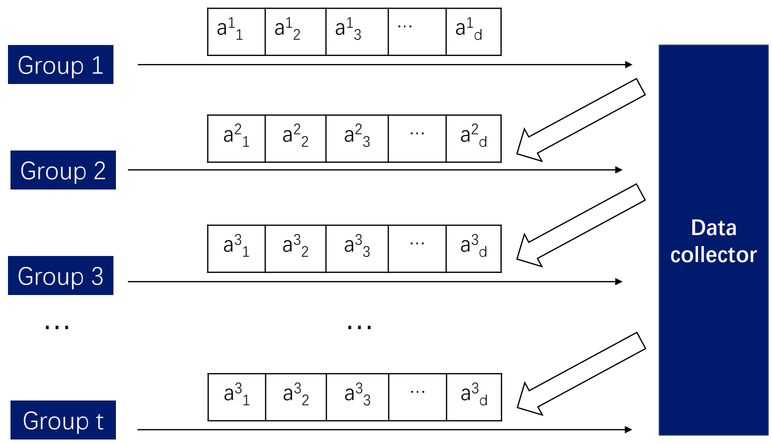


Figure 4.1: Attribute clusters in a smart building system

4.1.2 Intra-Cluster Attribute Correlation

To quantize the correlation between attributes inside a cluster, this chapter adopts the correlation coefficient η, which is based on the definition of covariance [94]. η is generally considered as an effective metric to measure how close two random variables X, Y are related to each other: $\eta = \frac{Cov(X,Y)}{\sqrt{Var(X)Var(Y)}}$. In our context, the vector X is constituted from the values of some attribute (say, A) provided by all the n users, thus, X_i denotes the value of some attribute A provided by the i-th user, which is also the i-th component of vector X. Likewise, vector Y denotes the same contents of some other attribute B.

Although the correlation coefficient can be derived from historical data, the above definition cannot calculate the perturbation possibility, as it only describes the relationship between two vectors as an entirety, which cannot answer such question as: "When the value of attribute A provided by a user is 1, what is the probability the value of attribute B from the same user being?"

To solve the problem, let's go back and see where the concept of covariance roots from. Notice that, covariance is essentially an inner product defined on a vector space, which differentiates two vectors by their included angles. In the definition of covariance, such inner product is defined as the expectation value of the multiplication of two random variables: $< X, Y >= E(XY) = \frac{\Sigma_1^n X_i Y_i}{n}$, which follows the form of standard inner product in Euclidean space and further defines the included angle of two vectors by its cosine value. This angle decides how correlated the two vectors are.

In fact, in this paradigm, the definition of inner product is not specifically requested, and the initial definition chooses the L_2 norm. Given another form of inner product, say, L_1 norm, the above metric remains, while the value of correlation coefficient varies. That is, if the inner product is given by $< X, Y >' = E(|X - Y|) = \frac{\Sigma_1^n |X_i - Y_i|}{n}$, the cosine value of the angle will be $\eta' = \frac{<X,Y>'}{\sqrt{<X,X>'<Y,Y>'}}$, which is another well-defined correlation coefficient of two variables.

For binary cases, $|X_i - Y_i|$ degrades to simple XOR, and η' degrades to $\frac{\Sigma_1^n X_i \oplus Y_i}{n}$. Let P_{ij} denote the degraded η', so the correlation of two binary attributes b_i and b_j in a cluster is formulated upon the probabilistic distribution of their values as below:

Definition 1. Intra-Cluster Attribute Correlation. *For attributes b_i and b_j which have correlation P_{ij}, b_i is equal to b_j with probability P_{ij}, and b_i differs from b_j with probability $1 - P_{ij}$.*

$$\begin{cases} b_i = b_j, & w.p.\ P_{ij} \\ b_i = 1 - b_j & w.p.\ 1 - P_{ij} \end{cases}$$

P_{ij} measures how probable the revealed value of one attribute can dictate the value of the other attribute. If $P_{ij} = 1$, b_i is completely correlated with b_j in a positive manner; if $P_{ij} = 0$, b_i is completely correlated with b_j in a negative manner. On the other hand, if $P_{ij} = 0.5$, b_i is independent of b_j.

In practical applications, it is more feasible to capture the correlation probability between two attributes by an interval, for example, "b_i is equal to b_j in the scope of probability $[0.7, 0.8]$". This leads to the following definition on the bounded correlation:

Definition 2. Bounded Intra-Cluster Attribute Correlation. *For attributes b_i and b_j which have correlation in the scope of $[P_1, P_2]$. Let $P_{ij} = Max(|P_1 - 0.5|, |P_2 - 0.5|)$, b_i is equal to (or differs from) b_j with probability bound P_{ij}, and b_i differs from (or equals to) b_j with probability bound $1 - P_{ij}$.*

In the rest of this chapter, P_{ij} denotes the **bounded correlation** unless otherwise stated.

4.1.3 Univariate Dominance LDP

Traditional ϵ-LDP is defined on all attributes, so that no two instances in the entire attribute space have distinctive probabilities (i.e., one being very high and the other being very low). However, as the number of attributes (i.e., dimensionality) increases, this privacy model becomes an overkill. Our key observation is that since the number of instances grows exponentially with dimensionality, this chapter can allow some instances to be more probable, as long as the number of these instances is still too large for the collector to infer the value of any attribute with high confidence. As such, this chapter proposes a relaxed model of LDP, namely, univariate dominance LDP, by ensuring not a single attribute has distinctive probabilities on any two values over its univariate distribution.

Definition 3. *ϵ-Univariate Dominance LDP.* *For any attribute b_i and any two inputs s and s', a perturbation algorithm $\mathcal{M}$ satisfies with UDLDP, if the following condition holds for any possible output Y from Range $\mathcal{M}$*

$$e^{-\epsilon} \leq \frac{P[\mathcal{M}(b_i = s) = Y]}{P[\mathcal{M}(b_i = s') = Y]} \leq e^{\epsilon}$$

Although UDLDP is a relaxed privacy model of LDP, when attributes are independent of each other (any P_{ij} is 0.5 in binary case), observation on one attribute becomes an independent event to that on any other attribute. Therefore, according to the parallel composition theorem [69], **an ϵ-UDLDP protocol on each attribute also satisfies ϵ-LDP in the entire high-dimensional space**. In the worst case, when all $p_{ij} = 0.5$, ϵ-UDLDP degenerates to the original ϵ-LDP.

4.2 CBP: Correlation-bounded Perturbation Protocol

This chapter describes the details of the correlation-bounded perturbation protocol to satisfy ϵ UDLDP. The protocol consists of a correlation bounding algorithm, a user perturbation algorithm, and a collector calibration algorithm.

The system model is described in Fig. 5.1. First, based on historical data, the collector partitions the attributes into independent clusters and calculates the bound of correlation and perturbation probability Q_t in each cluster t (step ①). Second, each user receives these Q_t, perturbs her high-dimensional data vector accordingly, and sends it to the data collector (step ②). Finally, the data collector calibrates those data and estimates the statistics (i.e., frequency count) of each attribute (step ③). In what follows, this chapter presents the detailed algorithm in each step.

4.2.1 Calculating Perturbation Probability

As the first step, the data collector calculates the perturbation probability Q_t in each cluster C_t according to apriori knowledge and historical data on each attribute. For ease of presentation, in what follows, this chapter assumes only one cluster and calculate its perturbation probability. This chapter starts with a two-dimensional

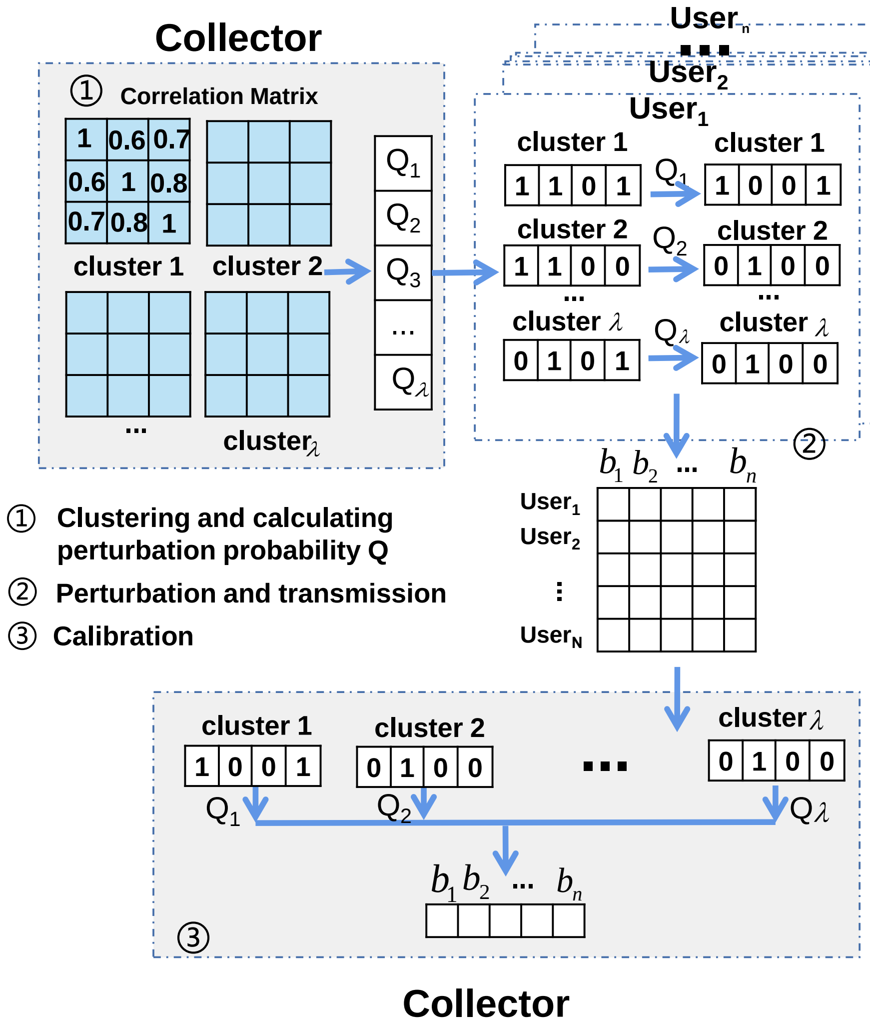


Figure 4.2: Correlation-bounded Perturbation Protocol

case and then generalize to multi-dimensional and multi-value cases.

Two-dimensional case. When the perturbation probability is fixed to q_i, the perturbation algorithm of attribute b_i becomes:

$$b_i' = \begin{cases} b_i, & w.p.\ q_i \\ 1 - b_i & w.p.\ 1 - q_i \end{cases}$$

To derive a q_i that satisfies definition 3, this chapter firstly needs to deduct the probability of $b_i = s (s \in \{0, 1\})$ upon knowing the output $y(b_1', ..., b_i', ..., b_n')$. In fact, the probability of $b_i = s$ is determined by a series of observations on b'_i and every $b'_j(j \neq i)$. For observation on b'_i, it is called **direct observation**, as it determines b_i directly. For observations on $b'_j(j \neq i)$, they are called **indirect observations**, as they determine b_j, from which b_i can be deducted using the correlations P_{ij}. The overall effect of all the observations shall determine q_i.

In the case $n = 2$, only one direct observation and one indirect observation are involved. For the direct observation on b'_i, b_i can be determined by q_i, that is, the probability of $b_i = s$ equals to q_i. As discussed in the definition of correlation and UDLDP, observation on the other perturbed attribute b'_j results in an indirect observation on b_i, as b_i can be deducted from b_j and their correlation P_{ij}. Let $E_{b_i=s}^{b'_j=s}$ denote the indirect observation on $b'_j = s$ for $b_i = s$, which can be written as:

$$\begin{aligned}
&P(E_{b_i=s}^{b'_j=s}) = P(b'_j = s|b_i = s)\\
=&P(b_j = b_i, b'_j = b_j|b_i = s) + P(b_j \neq b_i, b'_j \neq b_j|b_i = s)\\
=&P(b_j = b_i)P(b'_j = b_j|b_i = s) + P(b_j \neq b_i)P(b'_j \neq b_j|b_i = s)\\
=&P_{ij}q_i + (1 - P_{ij})(1 - q_i).
\end{aligned} \tag{4.1}$$

On the basis of Equ. 4.1, the overall effect of the two observations can be calculated from the Bayes formula. For simplicity, let $M_{b_i=s}^1$ denote $b_i = s$ judged by the direct observation, and $M_{b_i=s}^2$ denote $b_i = s$ judged by the overall effect of the two observations, which can be calculated from:

$$\begin{aligned}
&P(M_{b_i=s}^2) = P(M_{b_i=s}^1|b'_j = s)\\
=&\frac{P(M_{b_i=s}^1)P(E_{b_i=s}^{b'_j=s})}{P(M_{b_i=s}^1)P(E_{b_i=s}^{b'_j=s}) + (1 - P(M_{b_i=s}^1))P(E_{b_i=\bar{s}}^{b'_j=s})}
\end{aligned}$$

Likewise, $P(M_{b_i=\bar{s}}^2)$ can be calculated. Inserting the observations on b'_i into Definition 3, the following lemma is obtained:

Lemma 1. *For any attribute b_i, the algorithm satisfies ϵ-UDLDP in 2-dimension, if the perturbation probability q_i satisfies*

$$e^{-\epsilon} \leq \frac{P(M^2_{b_i=s})}{P(M^2_{b_i=\bar{s}})} \leq e^{\epsilon}$$

Multi-dimensional case. Likewise, the perturbation probabilities $q_i (i \in \{1, 2, ..., n\})$ in multi-dimensional cases can be calculated. Let $M^k_{b_i=s}$ denote the event that $b_i = s$ upon the overall effect of the first k observations. For each attribute b_i, there is one direct observation b'_i and $n-1$ indirect observations $b'_j (j \neq i)$. Similar to the 2D case, the probability $b_i = s$ is judged by the overall effect of the n observations from which a perturbation q_i can be calculated. Since $P(M^k_{b_i=s})$ can be deducted with $P(M^{k-1}_{b_i=s})$ and $P(b'_j = s)$ (b'_j should be the k-th observation), the calculation of the perturbation probability is summarized in the following theorem:

Theorem 1. *For any attribute b_i, the algorithm satisfies with ϵ-UDLDP in n-dimension, if the perturbation probability q_i satisfies*

$$e^{-\epsilon} \leq \frac{P(M^n_{b_i=s})}{P(M^n_{b_i=\bar{s}})} \leq e^{\epsilon} \tag{4.2}$$

where

$$\begin{aligned} P(M^k_{b_i=s}) &= P(M^{k-1}_{b_i=s} | b'_j = s) \\ = &\frac{P(M^{k-1}_{b_i=s}) P(E^{b'_j=s}_{b_i=s})}{P(M^{k-1}_{b_i=s}) P(E^{b'_j=s}_{b_i=s}) + (1 - P(M^{k-1}_{b_i=s})) P(E^{b'_j=s}_{b_i=\bar{s}})} \end{aligned} \tag{4.3}$$

for $k \in \{2, ..., n\}$.

Bounded Perturbation Probability. Different attribute b_i can result in different solution of q_i according to Lemma 1. Although the precise perturbation probabilities can be calculated, the involved inequalities can be computationally heavy, or worse, may have no solution when the dimension goes high.

To simplify the calculation, the perturbation probabilities on each attribute can be fixed at the same value.

As such, the minimum value among all the calculated q_i is chosen as the worst case to guarantee the given privacy budget ϵ. In this way, the above inequalities can be greatly simplified and lead to a bounded perturbation probability Q_0:

$$Q_0 = min\{q_1, q_2, ..., q_n\}$$

If a series of differential e_i^ϵ is obtained by substituting Q_0 into Equ. 4.2, the maximum one in e_i^ϵ meets the user-acknowledged privacy budget ϵ, which is also guaranteed for the rest.

Note that, Equ. 4.3 can be degraded to a completely related (or independent) situation between attributes. When all attributes are completely related, the maximum value of $P(M^n_{b_i=s})$ will be Q_0^n and the minimum value of $P(M^n_{b_i=\bar{s}})$ will be $(1-Q_0)^n$, which is equivalent to the usual case where the privacy budget is allocated at $\frac{\epsilon}{n}$ for each attribute. When all attributes are independent, the maximum value of $P(M^n_{b_i=s})$ will be Q_0 and the minimum value of $P(M^n_{b_i=\bar{s}})$ will be $1-Q_0$. In such case, each attribute is allocated with a privacy budgets ϵ.

Multi-Value case. The methodology of CBP is not limited to binary values. A correlation matrix (as shown in Table 4.1) can be used to describe the correlation between two attributes a and b defined on a domain $1, 2, ..., m$ at the 2-dimensional-m-value case (and a tensor for the multi-dimensional-m-value case), in which $P_{a_i b_j}$ denotes the probability that attribute a equals to i and attribute b equals to j simultaneously. Due to the page limit, the details have to be put into an outside document. The link can be found here: `https://github.com/accountud/udldp/blob/main/appendix.pdf`

4.2.2 Perturbation and Calibration

User Perturbation Algorithm. On the user side, every user holds a binary attribute array B=$\{b_1, b_2, b_3, ..., b_n\}$ which is divided in λ clusters with c_i attributes($\sum c_i =$

Table 4.1: Correlation matrix.

Correlation matrix	1	2	...	m
1	Pa_1b_1	Pa_1b_2	$Pa_1b_{...}$	Pa_1b_m
2	Pa_2b_1	Pa_2b_2	$Pa_2b_{...}$	Pa_2b_m
...	...	...	...	...
m	Pa_mb_1	Pa_mb_2	$Pa_mb_{...}$	Pa_mb_m

n) in cluster C_t[1]. Each user obtains the perturbation probability $Q_t(t = \{0, 1, ..., \lambda\})$ of cluster C_t from the data collector.

For each user's binary attribute b_j in cluster C_t, a binary value $b_j^{t'}$ is generated with probability

$$b_j'^t = \begin{cases} b_j^t, & w.p.\ Q_t \\ 1 - b_j^t & w.p.\ 1 - Q_t \end{cases}$$

Each user sends the generated value vector $B' = \{B_1', B_2', ..., B_\lambda'\}$ to the data collector, where $B_i' = \{b_1^{i'}, ..., b_{c_i}^{i'}\}$.

Calibration and Estimation. Data collector estimates the frequency from the collected reports, i.e., the total number of "1"s of each attribute from N users. Let b_j^t denote the j-th attribute in cluster C_t, Q_t denote the corresponding perturbation probability, m_j denote the number of users reporting "1" collected, and π_j denote the true proportion of users reporting "1". The following theorem shows an unbiased estimation of such frequency:

Theorem 2. *In CBP, the frequency estimation of the j-th attribute in cluster C_t calculated from the following formula is unbiased:*

$$\widetilde{N_{b_j^t}} = \frac{m_j - (1 - Q_t)N}{(2Q_t - 1)}$$

[1] In this chapter, we mainly discuss binary case.

Proof.

$$
\begin{aligned}
E[\widetilde{N_{b_j^t}}] &= E[\frac{m_j - (1-Q_t)N}{(2Q_t - 1)}] \\
&= \frac{N\pi_j Q_t + N(1-\pi_j)(1-Q_t) - (1-Q_t)N}{(2Q_t - 1)} = N\pi_j
\end{aligned}
$$

□

4.3 CBPS: Correlation-Bounded Perturbation with Sampling

So far, this chapter has presented the CBP protocol to estimate statistics from multi-attribute data with correlations. This chapter generalizes CBP to support sampling, where users are partitioned into groups, and all users in the same group send their values of the same attribute. On the one hand, sampling is a common technique in sensor networks and the Internet of Things to reduce bandwidth costs. On the other hand, it is also an alternative method to achieve good data utility for LDP in multi-dimensional data. As such, it is natural to combine CBP and sampling for better accuracy under the same privacy budget. The main challenge is that, since existing correlations break the equality of attributes, the proportion of samples from different attributes should no longer be equal. This chapter will derive an optimal sampling strategy for the CBP protocol, namely CBPS, that optimizes the overall estimation accuracy. Note that CBPS only needs to concern one cluster at a time, as the attributes in each cluster are independent of those in the other clusters.

4.3.1 Optimal Sample Allocation

In the context of sampling among attributes, the "one-user-one-attribute" sampling strategy is commonly believed to achieve the minimum overall variance [114], i.e., the best estimation accuracy. However, when sampling with CBP, the symmetry between attributes no longer exists. As such, this sampling strategy is no longer optimal. In fact, this sample allocation problem can be formulated as below.

Optimal Sample Allocation Problem. Let n denote the number of attributes, N the total number of samples to be collected, and γ the number of attributes each user is asked to collect.Without loss of generality, the n attributes can be partitioned into g disjoint groups $(G_1, G_2, ..., G_g)$, each with γ attributes and with a perturbation probability $(\mathbb{Q}_1, \mathbb{Q}_2, ..., \mathbb{Q}_g)$. Each user is assigned to one group and sends the perturbed γ attribute values of this group to the collector. The optimal sample allocation problem is to decide N_i, the number of users in each group G_i, where $\Sigma N_i = N$ such that the total variance of estimation of all attributes is minimized.

Variance Estimation. Let us first decide the minimum variance of estimation in CBPS, where all perturbation possibilities $\mathbb{Q}_i$ of G_i are already given:

Lemma 2. *The minimum variance of estimation in CBPS $Var[\widetilde{\pi}(B)]$ is determined by the following formula:*

$$Var[\widetilde{\pi}(B)] \geq (\sum_{i=1}^{g} \sqrt{\gamma \frac{\mathbb{Q}_i(1-\mathbb{Q}_i)}{(2\mathbb{Q}_i - 1)^2}})^2/N$$

If and only if vector $\sqrt{\frac{\gamma \mathbb{Q}_i(1-\mathbb{Q}_i)}{N(2\mathbb{Q}_i-1)^2}}$ *and* $\sqrt{N}$ *are linearly dependent, the inequality holds as an equality.*

Proof. The proof can be found in the same outside document linked in Chapter IV, the "Multi-value case" part. □

From Lemma 2, a natural deduction of the optimal choice of γ can be made instantly:

Corollary. If all $\mathbb{Q}_i$s are pre-given, the minimal possible variances of all γ can be directly calculated. An optimal γ can be chosen by simple comparison.

Optimal Proportion of Users. With Lemma 2, the optimal proportion of users can be determined by the following theorem:

Theorem 3. *The optimal proportion of N_i is determined by the solution of the following $g+1$ formulas, where λ is an unknown parameter to be determined in the calculation:*

$$\lambda N_i = \sqrt{\frac{\gamma \mathbb{Q}_i(1-\mathbb{Q}_i)}{N_i(2\mathbb{Q}_i-1)^2}}, \sum N_i = N, i \in \{1, 2, ..., g\} \tag{4.4}$$

Proof. The proof can be found in the same document where Lemma 2 is proved. □

4.3.2 Calibration and Estimation

Now the collector estimates the number of "1"s from the collected reports of N users. Let b_j^i denote attribute b_j in group G_i, $\mathbb{Q}_i$ denote the corresponding perturbation probability, N_i denote the number of users in G_i, and m_j^i denote the number of users reporting "1". The following theorem shows an unbiased estimation of such frequency.

Theorem 4. *In CBPS, the frequency estimation of the j-th attribute in Group G_i calculated from the following formula is unbiased.*

$$\widetilde{N_{b_j^i}} = \frac{m_j^i - (1-\mathbb{Q}_i)N_i}{(2\mathbb{Q}_i - 1)}$$

Proof. The proof follows that of Theorem 2. □

4.4 Experimental Results

This chapter evaluates the accuracy of the proposed methods, namely CBP and CBPS, on both real-world and synthetic datasets. For comparison, this chapter also evaluates the basic RAPPOR [45], optimized unary encoding (OUE, state-of-the-art frequency estimation protocol for single attributes) [114], and Sampling (each user randomly selecting and perturbing some attributes). The accuracy is measured by the mean square error **MSE** [83], which is the sum of the square difference between the real frequency $F(b)$ and the estimated frequency $F(b)'$ of all attribute $b \in X$. Formally,

$$MSE = \frac{1}{n}\sum_{b \in X}(F(b) - F(b)')^2$$

This chapter conducts the experiments using MATLAB R2019b on a PC with AMD Ryzen 7 2700X eight-core processor, 64GB RAM, Windows 10. All measurements are repeated 100 times and averaged.In the following, the experimental results are shown in four settings: (1) The performance robustness of different correlation ranges, (2) a single cluster of multiple binary attributes, (3) multiple clusters of high-dimensional binary attributes, and (4) a single cluster of multivalued attributes.

There are four real datasets **PM2.5**[2], **TH**[3], **CMC**[4] and **CLAVE**[5], where parameters are summarized in Table 4.2. **PM2.5** is an hourly data set that consists of the PM2.5 concentration data from US embassy in Beijing between 2010 to 2014. **TH** consists of the energy use data in a low energy building, i.e., the temperature and humidity conditions. This chapter normalizes the domain of each attribute into binary values {0,1} (by setting those below average to 0, and vice versa). **CMC** is a subset of 1987 National Indonesia Contraceptive Prevalence Survey, which includes 8 binary attributes on married women, such as pregnancy. **Clave** consists of 16 binary attributes, which are attack-point vectors where “1” indicates the substantial presence (and “0” indicating absence) in a certain time window. These correlation ranges are calculated using 10% of user data as prior knowledge, while the remaining 90% is used for experimental validation.

[2]https://archive.ics.uci.edu/ml/datasets/Beijing
[3]http://mlr.cs.umass.edu/ml/datasets.html
[4]https://archive.ics.uci.edu/ml/datasets/Contraceptive+Method+Choice
[5]https://archive.ics.uci.edu/ml/datasets/Firm-Teacher-Clave-Direction-Classification

Table 4.2: Single cluster 2 value Datasets

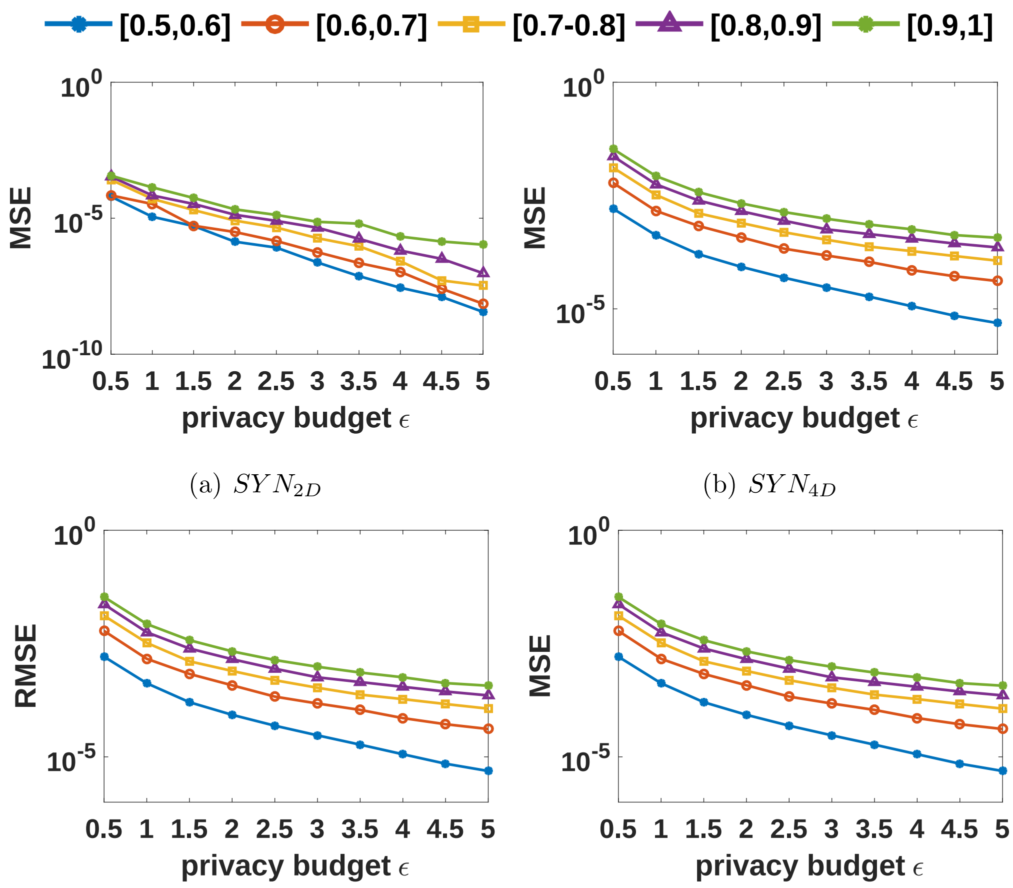
data	dimension	number of users	correlation range
TH	10	19.7K	0.50-0.80
PM2.5	8	43.6K	0.50-0.85
CMC	8	1.5K	0.50-0.80
CLAVE	16	10.8K	0.50-0.55

4.4.1 Performance of CBP

Robustness of Correlation Ranges. In the first set of experiments, the trend of MSE of CBP with respect to ϵ in different ranges of correlations is verified. As synthetic data are more appropriate for such controlled experiment, four datasets $SYN_{2D}, SYN_{4D}, SYN_{7D}$ and SYN_{10D} with 100K users' data and 2, 4, 7, and 10 dimensions, respectively, are generated. For the four datasets, five equal-interval ranges of correlation are employed to observe the impact on MSE. It can be seen from the figures that the performance between the five curves is stable in different intervals - they decrease with the increment of ϵ in similar trends, and the closer the correlation to 0.5, the smaller the MSE is.

Accuracy. Fig.4.4 shows the frequency estimation accuracy of the four candidate methods, that is, CBP, basic RAPPOR [45], SUE [114], OUE [114] and $Sampling_1$ (1 denotes sampling only one attribute). Notice CBP and $Sampling_1$ are two important cases of CBPS: The former is sampling all the attributes (aka. $CBPS_{full}$), that is, each user sends all n attributes. The other is sampling only one attribute (aka. $CBPS_{single}$), in which each user sends only one attribute.

The result shows, when the privacy budget is relatively small, $Sampling_1(CBPS_{single})$ behaves well, as the cost of utility is rather large when partitioning a small epsilon in pure LDP protocols [114]. With the increment of privacy budget ϵ, the impact of privacy partition gets weaker and the performance degrades soon. However, our



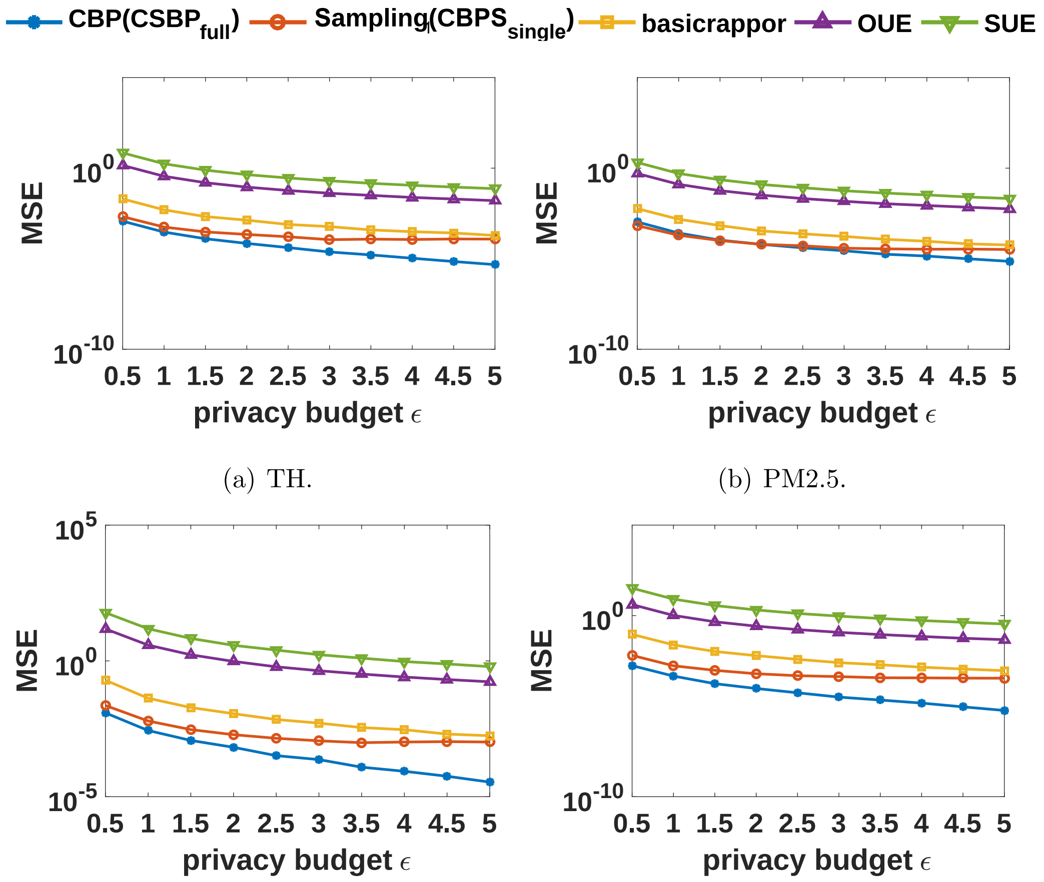
(a) SYN_{2D} (b) SYN_{4D} (c) SYN_{7D} (d) SYN_{10D}

Figure 4.3: CBP on four datasets in different intervals.

CBP ($CBPS_{full}$) benefits from the correlation between attributes to gain a better perturbation probability, which remarkably alleviates the impact of privacy partition when ϵ is small.

4.4.2 Performance of CBPS

Accuracy. Fig. 4.5 compares the performance between CBPS and Sampling. Both CBPS and Sampling sample $i(i \in \{1, 2, ..., n\})$ attributes, that is, each user uploads i attributes instead of only one. For fairness, the latter samples with the same grouping scheme (and thus the same number of attributes to sample). Clearly, CBPS degrades to Sampling at the 1-dimension point, while degrading to CBP at the maximal di-



(a) TH.

(b) PM2.5.

(c) CMC.

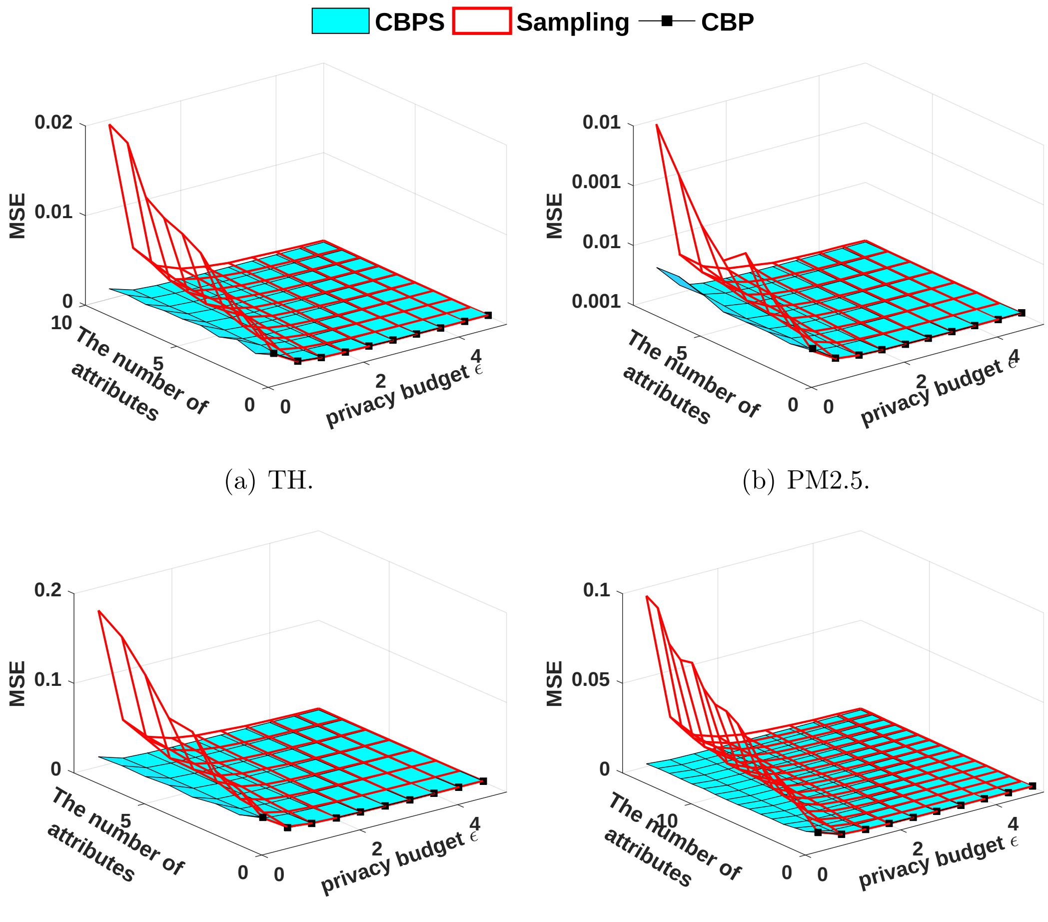
(d) CLAVE.

Figure 4.4: The MSE of different methods.

mension point.It is observed that the surface of CBPS's is always lower than that of Sampling since, while they collect the same number of dimensions, the latter has higher perturbation probabilities and therefore has lower MSE.

Variance. The empirically measured variances are now shown to match the theoretical results in Lemma 2, so sampling only one attribute is not always the optimal solution. To ensure the credibility of the measured variance, the measurement is repeated 1000 times in all experiments below.

Fig. 4.6 shows the comparison of empirical and theoretical results of CBPS sampling different attributes, in which EV_{S1} denotes the empirical variance of sampling only one attribute, and TV_{S1} denotes the theoretical variance of sampling only one attribute.



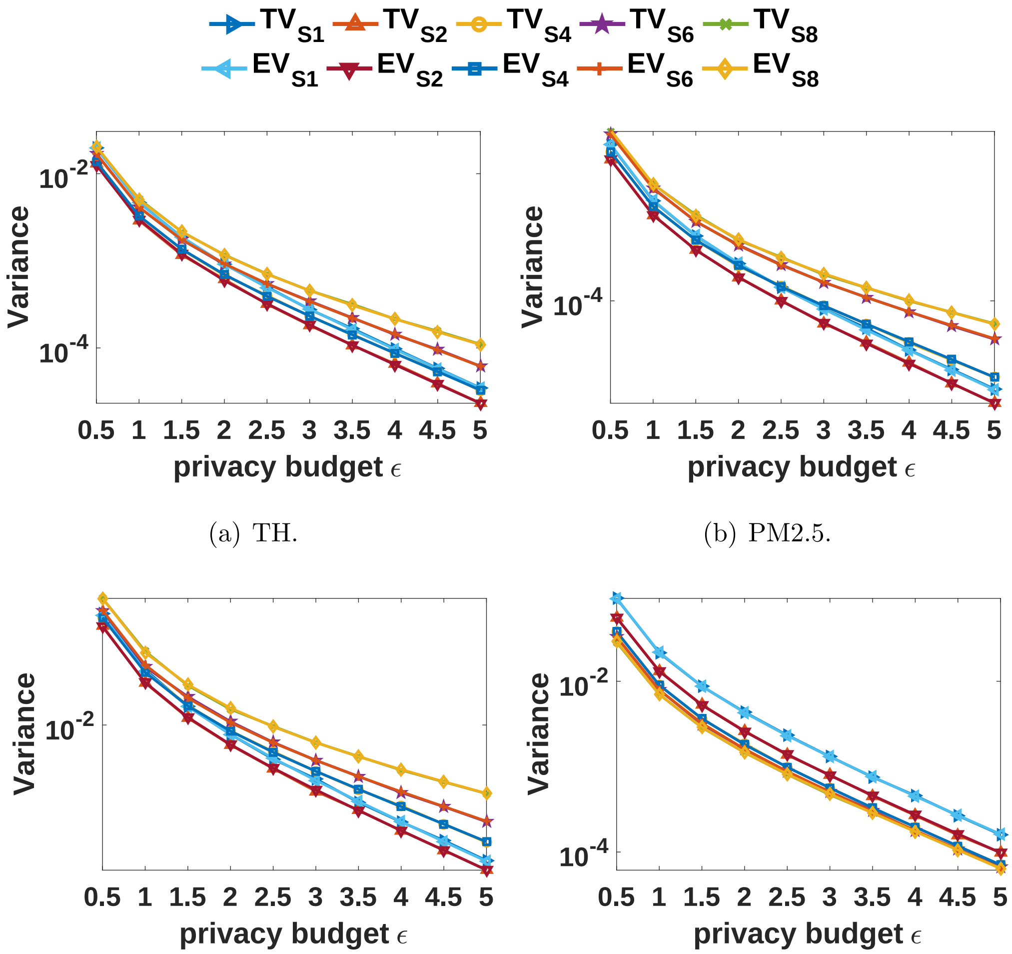
(a) TH. (b) PM2.5.

(c) CMC. (d) CLAVE.

Figure 4.5: The MSE of Sampling vs. CBPS.

It can be seen from the figure that the empirical results match very well with the theoretical results. Figs. 4.6(a), 4.6(b), and 4.6(c) show that sampling 2 attributes outperforms others while sampling 8 attributes behaves the worst. It implies that sampling only one attribute is not usually the best choice. Fig. 4.6(d) shows that the variance decreases with respect to the increment of attributes, in which sampling 8 attributes outperforms others. It can be concluded that when correlations between data are relatively weak, the more attributes sampled, the better the variance is.

User Allocation Scheme. According to Lemma 2, when n attributes are distributed to different groups, there should exist an optimal user allocation scheme such that the minimum variance of estimation in CBPS is reached. This part com-



(a) TH.

(b) PM2.5.

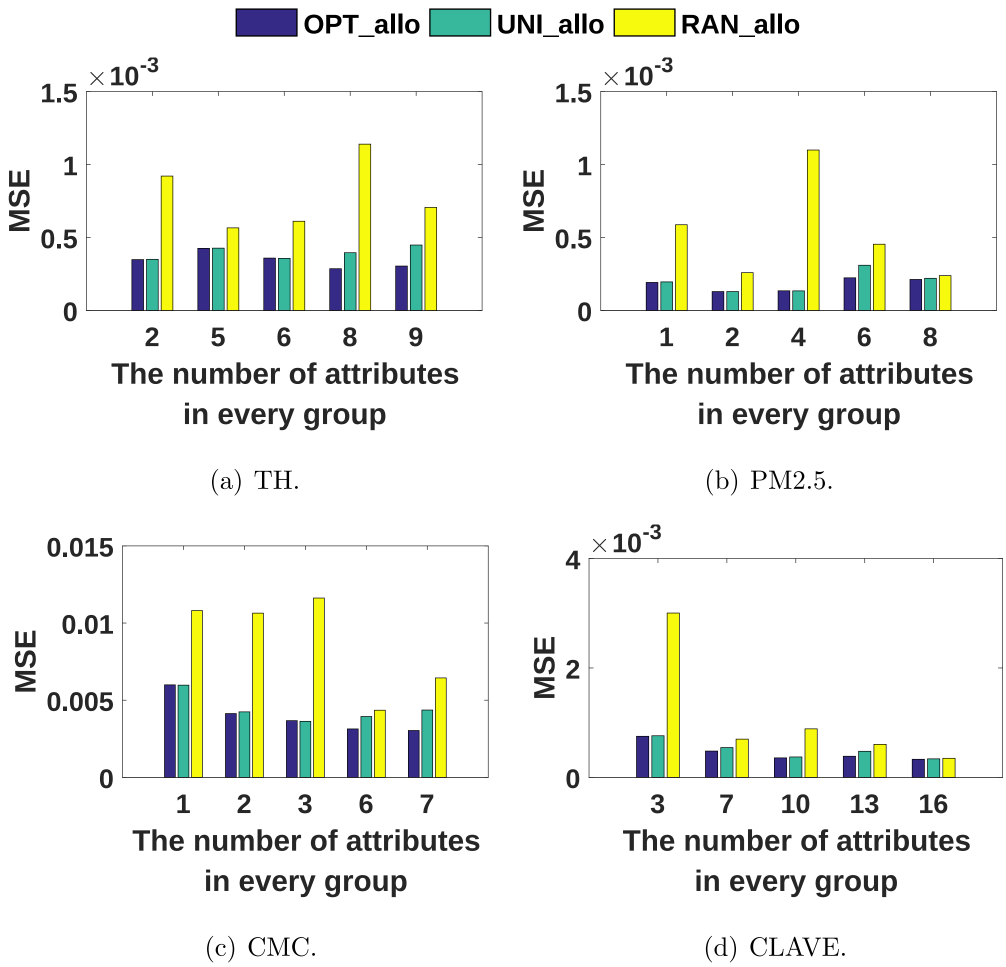
(c) CMC.

(d) CLAVE.

Figure 4.6: The variance of Sampling vs. CBPS.

pares the performance of three different schemes where $\epsilon = 1$ to show that our optimal allocation scheme outperforms others, including our scheme (OPT_{allo}), the uniform allocation (UNI_{allo}, each group has the same number of users) and the random allocation (RAN_{allo}, each group has a random number of users).

From Fig. 4.7, it can be seen that OPT_{allo} outperforms others, as there are different perturbation parameters Q_i in different groups and the scheme achieves the minimum variance by allocating n_i according to the corresponding Q_i.



(a) TH.

(b) PM2.5.

(c) CMC.

(d) CLAVE.

Figure 4.7: The MSE of different user allocation schemes.

4.4.3 Conclusion of Experimental Results

Based on the above experimental results, the following conclusions are drawn on the performance of CBP and CBPS. First, the weaker the correlations are, the better accuracy of the CBP is. Second, CBP and $Sampling_1$, two special cases of CBPS, outperform other schemes. At most cases, $Sampling_1$ performs better than CBP when the privacy budget is small. Third, given the grouping scheme, CBPS is always better than Sampling. Furthermore, the empirically measured variances of CBPS match the theoretical results, which means that the best sampling scheme can be chosen by comparing the theoretical variances of different sampling schemes. Finally, an optimal user allocation scheme of CBPS is verified to achieve a smaller MSE than uniform allocation and random allocation.

4.5 Summary

This chapter studies how to collect correlated high dimensional data with a relaxed privacy model of LDP, namely, UDLDP. Based on this model, correlation-bounded perturbation (CBP) protocol and CBP with sampling (CBPS) are presented. Both of them address the overallocated privacy budget issue from traditional LDP techniques. The experimental results on both real-world datasets and synthetic data show the efficiency of the proposed CBP and CBPS protocols.

Chapter 5

Top-k Discovery under Local Differential Privacy: An Adaptive Sampling Approach

This chapter studies top-k item discovery in a set-valued dataset under LDP. In this setting, each user has a set of private items, such as the music playlist, web search history, and location trajectories. To protect the privacy, each user perturbs his data using LDP before sending it to the data collector. The collector then analyzes the perturbed data to identify the most frequent k items. For instance, in iOS [7], users' emoji data is perturbed using LDP before being sent to Apple. This enables Apple to estimate the frequency of each emoji and identify the most popular ones, without compromising user privacy.

However, the main challenge in discovering top-k items from a set-valued dataset by conventional LDP mechanisms lies in the fact that users have varying numbers of items and the item domain (with size d) is usually extremely large, which leads to poor utility of the estimation. LDPMiner [88] and SVIM [115] address these challenges using a padding-and-sampling-based frequency oracle (PSFO) approach, in which a subset of users (e.g., 20%) report the perturbed size of their private sets to determine the padding length l, which is typically set to the 90^{th} percentile of the former. The remaining users augment their private set with dummy items to pad to the size of l. Then one item is randomly selected from the padded set and reported using a frequency oracle GRR or OLH [115]. Although PSFO increases the probability of sampling an existing item by each user, it still faces several challenges, including inaccuracies in determining the padding length l for small datasets, excessive padding length for most users, and biased frequency estimation due to some users with more items than the padding length l.

Uniform sampling is an unbiased scheme that treats all items equally over time and can effectively address PSFO issues. Each user generates a d-bit string with existing item locations marked as 1 and all other locations marked as 0. The user then samples and reports a bit (i.e., 0 or 1) from the string uniformly at random. However, uniform sampling can lead to most users uploading non-existent items, which contributes little information to the results. To overcome this limitation, the data collector can adjust the sampling scheme based on the information collected over time. For example, less frequent items can be sampled less often, enabling the estimation results to focus on frequent items from the real top-k set.

Identifying the top-k items via adaptive sampling can be regarded as a Multi-armed bandit (MAB) problem. In the MAB problem, the decision-maker encounters a d-armed bandit, where each arm corresponds to a unique probability distribution. The decision-maker adaptively samples from each arm to obtain its corresponding reward and achieve their intended targets, such as returning the k arms with the highest reward to minimize regret. The arm represents the item in top-k items estimation, with the decision-maker as the data collector. The discovery of the top-k items of set value data is equivalent to selecting k arms with the highest reward in a MAB problem.

This chapter explores the problem of identifying top-k set-valued data under LDP settings, with the goal of returning the sets of top-k items along with their frequencies, given a fixed number of samples. However, traditional MAB solutions need to be revised due to limited sample sizes, privacy budgets, and time constraints in LDP systems. To overcome these challenges, this chapter proposes an adaptive sampling algorithm, namely Adaptive-RR Bandit Sampling (ARBS), which divides users into multiple partitions and sequentially lets users in each partition report their local data in light of the previous estimation results. Consequently, it adaptively increases the sampling probabilities for those frequent items. On the other hand, this chapter further introduces ARBS with frequency (ARBSF) for estimating frequencies of the identified items and a Delay-contrained Batch Sampling algorithm (DBS) for optimizing the user allocation when given fixed rounds of interactions between the users and the collector, ensuring both accuracy and timeliness. To summarize, our contributions are three folded.

- To the best of our knowledge, this is the first work to formulae set-valued data collection under LDP as an MAB problem, which inspires us to enhance the estimation results significantly.
- This chapter proposes ARBS for identifying top-k frequent items and ARBSF for estimating the frequencies of them, in which this chapter utilizes data characteristics and adaptively adjust the sampling scheme to obtain better utility.
- This chapter proposes a minimal error scheme to ensure the constraint on limited rounds is fully satisfied, and its effectiveness is demonstrated through extensive empirical analysis.

The remainder of this chapter is organized as follows. Chapter 5.1 formally gives the problem definition. Chapter 5.2 presents the details of the proposed adaptive sampling scheme, followed by a delay-constrained solution in Chapter 5.3. Chapter 5.4 provides extensive experimental evaluations, and the chapter is concluded in Chapter 5.5.

5.1 Problem Definition and Naive Solutions

5.1.1 Problem Definition

There is a data collector and n users in the system. Each user maintains a private set of items, which of each is from the domain $V = \{v_1, v_2, ..., v_d\}$ with d distinct items. Let p_i denote the frequency of item v_i, and without loss of generality, it is assumed that

$$p_1 \geq p_2 \geq \cdots \geq p_d. \tag{5.1}$$

For the sake of privacy, each user perturbs and reports her set of items to the data collector, by following LDP protocols. Upon receiving the perturbed data from all users, the goals of the data collector are to identify the most frequent k items and to estimate their frequencies.

The first goal can be formulated as a top-k discovery problem, which involves finding a set of most frequent items while preserving a ranking that is close to the real one. Formally, it can be defined as follows.

Definition 4. *(Top-k discovery) Given the items set $V = \{v_1, \ldots, v_d\}$ with the true frequency in decreasing order, the top-k problem finds the item set T, where $T \subseteq V$, $|T| = k$ and for any item $v_i \in T$ with frequency p_i and for any item $v_j \in T^{\complement} = V - T$ with frequency p_j, therefore:*

$$p_i \geq p_j. \tag{5.2}$$

In particular, given the k-th most frequent item v_k's frequency f_k, it is known that $\forall v_i \in T$, $\exists \theta \geq 0$

$$p_i \geq p_k + \theta. \tag{5.3}$$

The second goal is to ensure the frequency estimation accuracy of items in T. Specifically, given the top-k frequent item set T, each p_i for the item $v_i \in T$ is estimated. This can be defined as a top-k item frequency estimation problem as follows.

Definition 5. *(Top-k frequency estimation) The top-k frequency estimation problem involves estimating the frequency p_i of each item $v_i \in T$, where T is a set of top-k frequent items. The frequency p_i is defined as the proportion of v_i's occurrences among all users.*

5.1.2 Uniform Sampling

Intuitively, each user can encode her set of items using a d bit string $\{0,1\}^d$, where 1 (resp. 0) indicates the existence (resp. non-existence) of an item in her set. Then a bit (0 or 1) can be sampled uniformly at random, perturbed by RR mechanism, and sent to the data collector. Then the collector estimates the frequency of each item based on the perturbed data from all user. In particular, item v_i observed by the data collector follows a Bernoulli distribution.

$$f_i = p_i \frac{e^\epsilon}{e^\epsilon + 1} + (1 - p_i)\frac{1}{e^\epsilon + 1} \tag{5.4}$$

where p_i is the real frequency of item v_i, and ϵ is the privacy budget. The above procedure adopts a **uniform sampling** strategy, where each item is sampled from the domain with equal probability. However, there are several issues that need to be further considered. Firstly, the approach neglects the differences between items. To be more specific, items with smaller frequencies can be sampled less frequently, which would allow for more users to be allocated towards the frequent items. To take this strategy further, in order to gather information on different items, the sampling process should be carried out in multiple rounds and the sampling strategy should be adjusted in a timely manner, thereby achieving optimal results.

5.2 Adaptive Sampling

This chapter proposes an adaptive sampling algorithm for identifying top-k items and estimating their frequencies. The system overview of the proposed scheme is first

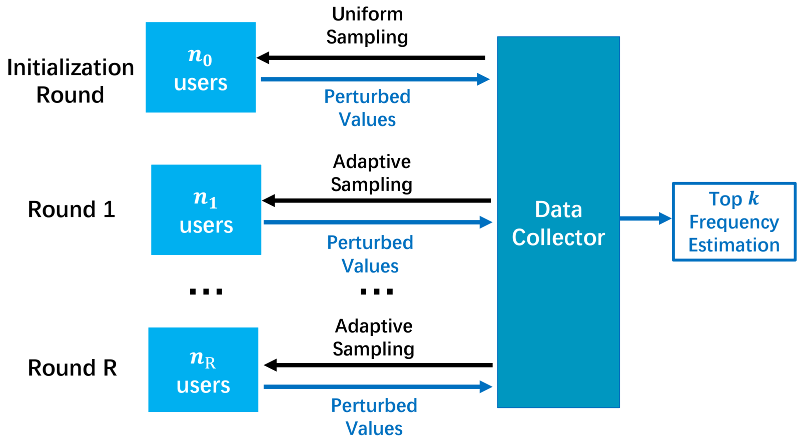


Figure 5.1: Illustration of the framework

presented, and then the implementation details are described.

5.2.1 System Overview

Our proposed framework employs an adaptive by-round strategy to overcome the limitations of naive solutions, as shown in Fig. 5.1. The data collection process begins with a uniform sampling of all items from n_0 users, establishing an initial knowledge base called Initialization. The remaining users are subsequently divided into R partitions, with $n_1, n_2, \ldots, n_R$ users respectively, for R rounds of data uploading. In each round, users in corresponding partition report their perturbed values only once to maintain ϵ-LDP privacy guarantee as defined in [69]. In particular, the sampling strategy in the upcoming round is in light of the information collected in the previous rounds. After completing all rounds, the data collector aggregates the top-k items and their frequencies. This chapter first considers a simplified case where there involves only one user in each of R round, i.e., $n_1 = n_2 = \cdots = n_R = 1$. This enables us to adjust our sampling strategy in real time.

5.2.2 Knowledge Initialization

To estimate the frequency of each item, let t_i denote the number of times item v_i is sampled and let $\{x_1^i, \dots, x_{t_i}^i\}$ denote the reported values. The observed frequency is given by $\hat{f}_i = 1/t_i \sum_{j=1}^{t_i} x_j^i$, which is an unbiased estimation of f_i in Equ. 5.4. It is unjustifiable to set different t_i without any knowledge on v_i. Therefore, to implement the adaptive sampling scheme, it is crucial to establish some prior knowledge about the items (e.g., item frequency distribution), which can be accomplished in a similar manner as uniform sampling.

Algorithm 2 shows our idea. Note that in uniform sampling, each user randomly selects an item to report, which generally results in each item being chosen approximately the same number of times. However, there still exists some approximation error due to randomness. Algorithm 2 corrects this by letting the data collector directs each user to sample a specific item (and then perturb it), which effectively controls the number of each item being sampled. Consequently, the data collector collects each item a fixed number of times $t = \frac{n_0}{d}$ (line 3), estimates the frequency for all items (line 4) and returns $\hat{f}_i$ as prior knowledge for further use (line 5).

Algorithm 2: Initialization(n_0)

Input: Privacy budget ϵ and initialization parameter n_0

Output: The aggregated frequency $\{\hat{f}_1, \dots, \hat{f}_d\}$

1: $t = \frac{n_0}{d}$ **for** $i = 1$ *to* d **do**
2:
Let t users report v_i by RR, denoted by $\{x_1^i, ..., x_t^i\}$
3: Calculate v_i's frequency as $\hat{f}_i = \sum_{j=1}^{t} x_j^i/(\frac{n_0}{d})$
4: **return** $\{\hat{f}_1, \dots, \hat{f}_d\}$

It is critical to determine an appropriate value for n_0 during initialization to ensure accurate results. A small n_0 may result in inaccurate information collected in the initialization round, leading to error accumulation in subsequent rounds. On the contrary, with a limited number of users, an excessively large n_0 may cause too many users to be concentrated in the initialization round, resulting in fewer users learning and utilizing prior knowledge in subsequent rounds. Previous studies have proposed various scales for n_0 [27, 63]; however, these values cannot be directly applied to our scheme. For example, Chen et al. [27] suggest a minimum n_0 of 5 to guarantee proper prior knowledge estimation. In the LDP setting, it is not possible to establish such a minimum n_0 because when ϵ is closer to 0, the prior knowledge becomes less accurate, and a larger n_0 is required. Therefore, under the constraints of LDP noise and a limited number of users, selecting a suitable n_0 is essential for performance enhancement.

To determine the optimal value of n_0, the prior knowledge derived from Algorithm 2 needs to be quantified first. This is done by defining the **initialization error and confidence**.

Definition 6. *(Initialization error and confidence) Let* e_0 *denote the initialization error, and* $1 - e_0$ *denote the initialization confidence in a top-*k *discovery problem. For an estimated top-*k *item set* $\hat{T}$*, let* $\hat{T}^{\complement} = V - \hat{T}$*.* $\exists e_0 \in (0, 1)$*,* $\forall \theta > 0$*,* $v_i \in \hat{T}$*, and* $v_j \in \hat{T^{\complement}}$*, we have* $\hat{f}_i \geq f_j - \theta$ *with the probability of* $1 - e_0$*.*

This definition does not take into account the order of the returned items and a larger initialization confidence (i.e., a smaller initialization error) indicates better performance in identifying the top-k items. The initialization error is obtained by applying the union bound and Hoeffding's inequality after sampling all items $\frac{n_0}{d}$ times, as demonstrated in the following theorem:

Theorem 5. $\forall \theta > 0$*, the probability of error* e_0 *of Initialization(*n_0*) in the top-*k *discovery satisfies*

$$e_0 \leq de^{-\frac{\theta^2 n_0}{2d}} \tag{5.5}$$

Proof. Recall that T is the top-k item set and items in $T^{\complement}$ are ranked below the top-k. For any θ, according to Eqs. 5.1 and 5.3, the following connection between T and $T^{\complement}$

can be established:

$$\forall i \in T,\ \forall j \in T^{\complement} : (f_i - f_j > \theta) \tag{5.6}$$

After initialization, according to Definition 6, there exists a $\hat{T} \subset V$ such that $|\hat{T}| = k$, and $\forall i \in \hat{T},\ \forall j \in (V - \hat{T}) :\ (\hat{f}_i \geq \hat{f}_j)$. An item v_j in $T^{\complement}$ can occur in $\hat{T}$, only if there is some items in T such that $\hat{f}_i < \hat{f}_j$. In turn, Equ. 5.6 implies that the latter event only occurs if $\hat{f}_i \leq f_i - \frac{\theta}{2}$ or $\hat{f}_j \geq f_j + \frac{\theta}{2}$. In terms of probabilities, by applying the union bound and Hoeffding's inequality, therefore:

$$\begin{aligned}
&P(\exists v_j \in T^{\complement} : (v_j \in \hat{T})) \\
&\leq P\left(\exists v_i \in T : \left(\hat{f}_i \leq f_i - \tfrac{\theta}{2}\right)\right) \\
&\quad + P\left(\exists v_j \in T^{\complement} : \left(\hat{f}_j \geq f_j + \tfrac{\theta}{2}\right)\right) \\
&\leq \textstyle\sum_{i \in T} P\left(\hat{f}_i \leq f_i - \frac{\theta}{2}\right) + \sum_{j \in T^{\complement}} P\left(\hat{f}_j \geq f_j + \frac{\theta}{2}\right) \\
&\leq |T|\, e^{-\frac{\theta^2 n_0}{2d}} + |T^{\complement}| e^{-\frac{\theta^2 n_0}{2d}} \\
&\leq \left(|T| + |T^{\complement}|\right) e^{-\frac{\theta^2 n_0}{2d}} \\
&\leq d e^{-\frac{\theta^2 n_0}{2d}}
\end{aligned} \tag{5.7}$$

□

The selection of n_0 is reduced to an optimization problem, with the goal of maximizing the total initialization confidence held by all users. This problem is simplified into two stages: the Initialization round and the subsequent rounds. During the Initialization round, the confidence is zero, and there are n_0 users involved. After initialization, the remaining users have an initialized confidence of $1 - de^{-\frac{\theta^2 n_0}{2d}}$. Thus, the selection of n_0 aims to maximize the overall initialized confidence I while considering the value of ϵ. The expression for I is given as follows:

$$\arg\max_{n_0} I = n_0 \cdot 0 + (n - n_0)(1 - de^{-\frac{\theta^2 n_0}{2d}}), \tag{5.8}$$

where θ is a parameter defined as

$$\theta = \frac{e^{\epsilon}}{e^{\epsilon} + 1} - \frac{1}{e^{\epsilon} + 1} = \frac{e^{\epsilon} - 1}{e^{\epsilon} + 1}.$$

Intuitively, as ϵ increases and θ decreases, a larger n_0 is necessary to achieve a satisfactory level of initialization confidence. While obtaining an analytical solution for the transcendental equation presented in Equ. 5.8 is challenging, numerical methods (e.g., Newton's method [50]) can be used to approximate n_0.

Remark. While an unbiased frequency estimate $\hat{p}_i$ can be further derived from $\hat{f}_i$ (see Eq. 5.13), $\hat{f}_i$ is chosen instead of $\hat{p}_i$ to determine n_0. Intuitively, the larger the item frequency distances are, the less accurate top-k discovery becomes. Therefore, an adaptive sampling scheme should be designed based on the observed item frequency distances. Applying $\hat{p}_i$ on an adaptive sampling scheme is good if data are collected without perturbation. However, when perturbation exists, the data collector obtains $\hat{f}_i$ rather than $\hat{p}_i$. If $\hat{p}_i$ is applied under LDP, items near the k-th item will be undersampled, which degrades the utility of top-k discovery. Therefore, designing adaptive sampling strategy based on $\hat{f}_i$ is able to achieve better performance.

5.2.3 Top-k Items Discovery

The uniform sampling scheme samples each item equally and ignores the distribution of items. However, when the items in T and $T^{\complement}$ are clearly separated, fewer users are needed to identify the top-k items. In such cases, it is more effective to assign more users to report items whose frequencies are near p_k, i.e., the frequency of the k-th most frequent item. This chapter proposes an adaptive sampling scheme, called Adaptive-RR Bandit Sampling (**ARBS**), which formalizes the sampling process as an MAB problem.

Design of ARBS. Before starting a new round, the frequency of all items has been estimated, which should ideally be $\hat{f}_1 \geq \dots \hat{f}_d$. However, due to limited statistical counts, an accurate ordering may not be obtained. Thus, the order obtained is set as $\hat{f}_{\xi_1} \geq \cdots \geq \hat{f}_{\xi_i} \geq \cdots \geq \hat{f}_{\xi_d}$, where $\hat{f}_{\xi_i}$ denotes the i-th most frequent item estimated.

The sampling probability for each item needs to be adjusted based on this information. To accomplish this, inspiration is drawn from the literature [63] and the concentration inequalities are inverted. Specifically, the error probability δ_i for each item is calculated, which reflects the distance between $\hat{f}_i$ and a boundary point μ of $\hat{T}$ and $\hat{T^{\complement}}$. Intuitively, μ should be drawn from the interval $[\hat{f_{\xi_k}}, \hat{f}\xi k+1]$, where $\hat{f}\xi_k$ can be regarded as the lower bound of item frequency in the set $\hat{T}$, while $\hat{f}\xi_{k+1}$ is the upper bound of the set $\hat{T^{\complement}}$. The closer $\hat{f}_i$ is to μ, the more users are required to determine whether its frequency is greater than μ.

Choosing an appropriate μ is critical, and a straightforward approach is to take the mean value of $\hat{f_{\xi_k}}$ and $\hat{f}\xi k+1$. However, as an item's frequency approaches μ, more users are required to estimate its frequency accurately. Therefore, the aim is to choose μ in $[\hat{f_{\xi_k}}, \hat{f}\xi k+1]$ that maximizes the distance between it and all f_i's. To optimize the sampling scheme, two different values of μ are set to maximize the distance between them. The choice of μ is described as follows:

$$\mu = \begin{cases} \hat{f}_{\xi_{k+1}}, & \text{if } \ \hat{f}_i \leq \hat{f}_{\xi_k} \\ \hat{f}_{\xi_k} & \text{otherwise} \end{cases}$$

The choice of μ in such two cases can maximize the distance between μ and $\hat{f}_i$'s and thus optimize the sampling while retaining the relative order among items.

After the initialization round, each item v_i has t_i reports from users, i.e., $\{x_1^i, \cdots, x_{t_i}^i\}$. Thus, an estimated frequency can be derived $\hat{f}_i = \frac{1}{t_i}\sum_{j=1}^{t} x_j^i$, and an empirical variance $\overline{\sigma}_i = \sqrt{\frac{1}{t_i}\sum_{j=1}^{t_i}(x_j^i - \hat{f}_i)^2}$ of all reports. Bernstein's inequality, as presented in Eq. 3.5, is then used to derive a tighter bound than that of Hoeffding's inequality. The resulting formula for δ_i is as follows.

Definition 7. *Given a frequency μ, for item v_i that is observed t_i times with f_i, the inverted error δ_i can be obtained using the inverse of Bernstein's inequality (in Equ. 3.5).*

$$\delta_i = 3e^{(-\frac{\overline{\sigma_i^2}t_i + \overline{\sigma_i}t_i\sqrt{\overline{\sigma_i^2} + 6|\hat{f}_i - \mu| + 3t_i|\hat{f}_i - \mu|}}{9})^2 t_i} \tag{5.9}$$

where $\overline{\sigma}_i^2 = \frac{1}{t_i}\sum_{j=1}^{t_i}(x_j^i - \hat{f}_i)^2$ *and* $\hat{f}_i = \frac{1}{t_i}\sum_{j=1}^{t} x_j^i$.

Proof. In this thesis, $R = 1$ and let $H = \sqrt{\frac{\log(3/\delta_i)}{t_i}}$. According to Equ. 3.5, we have $3H^2 + \sqrt{2}\overline{\sigma}_i H + |\hat{f}_i - \mu| = 0$. The solution for H^2 is

$$H^2 = \frac{|\hat{f}_i - \mu|}{3} + \frac{\overline{\sigma}_i\sqrt{\overline{\sigma}_i^2 + 6|\hat{f}_i - \mu|}}{9} + \frac{\overline{\sigma}_i^2}{9}.$$

Then, by eliminating H, δ can be expressed as:

$$\delta = 3e^{-H^2 t_i} = 3e^{(-\frac{|\hat{f}t-\mu|}{3} + \frac{\overline{\sigma}_i\sqrt{\overline{\sigma}_i^2+6|\hat{f}t-\mu|}}{9} + \frac{\overline{\sigma}_i^2}{9})t_i}$$

□

The parameter δ_i reflects the error in determining whether f_i is larger or smaller than μ, and items with larger δ are more likely to require additional users. To incorporate δ_i into the sampling scheme, users sample an item v_i by following the probability distribution

$$P(v_i) = \frac{\delta_i}{\sum_{j=1}^{d} \delta_j} \tag{5.10}$$

where $i \in \{1, 2, \cdots, d\}$. Equ. 5.10 normalizes each inverted error δ_i values across all items, ensuring that the sampling probabilities of all item sum up to 1. In each round of sampling, each user samples an item v_i with probability $P(v_i)$, and then update the sampling probability distribution by Equ. 5.9 and Equ. 5.10 after the current-round collection.

Algorithm of ARBS. The proposed ARBS scheme is given by Algorithm 3. At the beginning of the algorithm, each item is sampled $\frac{n_0}{d}$ times during Initialization (line 1). Then, the probabilities are updated based on the current knowledge using Equ. 5.10 (line 2). Next, the algorithm enters a by-round sampling phase, where only one item is sampled in each round (line 3). Specifically, an item v_s is sampled from the item set V with probability $P(v_s)$ (line 4). The estimates are updated using Equ. 5.9 and the sampling probabilities are updated using Equ. 5.10 (lines 5-6). After all the users upload the perturbed data, the estimated top-k items are determined by selecting the items with the highest k frequencies (line 7).

Algorithm 3: Adaptive-RR Bandit Sampling (ARBS)

Input: Privacy budget ϵ and initialization parameter n_0

Output: Top-k frequent item set $\hat{T}$

Initialization(n_0);

Update $\{P(v_1), , ..., P(v_d)\}$ according to Equ. 5.10; **for** $k = n_0 + 1$ *to* n **do**

Sample an item v_s from $V = \{v_1, ..., v_d\}$ following probability distribution $\{P(v_1), ...P(v_d)\}$;

Update δ_s according to Equ. 5.9;

Update $\{P(v_1), ..., P(v_d)\}$ according to Equ. 5.10;

Find $\hat{T} \subset V$ such that $|\hat{T}| = m$, and $\forall v_i \in \hat{T}, \forall v_j \in (V - \hat{T}) : \ (\hat{f}_i \geq \hat{f}_j)$. **endfor**

return $\hat{T}$

Error analysis. To address the limitation of the error bounds estimated by Equ. 3.4 and Equ. 3.5, which tend to be impractically large due to the limited number of users available for each item, this chapter utilizes the p-value of a one-sided two-sample T-test [98] to obtain more accurate error estimates for ARBS. The p-value is the probability of obtaining test results that are at least as extreme as a result observed, assuming the null hypothesis is true. For example, when testing the null hypothesis $f_i \leq f_j$, the p-value obtained from observing the samples v_i and v_j is 0.05, indicating a 0.05 probability of rejecting the null hypothesis. The p-value of v_i and v_j is denoted as $pv_{i,j}$, and the following theorem is presented based on these p-values:

Theorem 6. *Given the null hypothesis, i.e.,* $\forall v_i$ *in* $\hat{T}$ *and* $\forall v_j$ *in* $\hat{T}^{\complement}$ *satisfy*

$$f_i > f_j$$

and for any pair of items $\{v_i, v_j\}$ *with p-value* $pv_{i,j}$*, the error on the validity of this observation can be written as:*

$$\Delta = 1 - \prod_{i=1}^{i=k}(1 - pv_{i,k+1}) \prod_{i=k+1}^{i=d}(1 - pv_{i,k}).$$

Given the perturbed frequency of v_k is f_k, and for any positive value γ, we have $f_i - f_k > \gamma$ for all v_i in $\hat{T}$, and $f_k - f_j > \gamma$ for all v_j not in $\hat{T}$, the upper bound of the error is:

$$\Delta_{worst} = 1-$$
$$\prod_{i=1}^{i=k}(1 - \sum_{j=0}^{\frac{n}{d}} \binom{\frac{n}{d}}{j} \overline{F}^j (1 - (\overline{F}))^{\frac{n}{d}-j} \sum_{g=0}^{j} \binom{j}{g} \underline{F}^g (1 - \underline{F})^{j-g})$$
$$\prod_{i=k+1}^{i=d} (1 - \sum_{j=0}^{\frac{n}{d}} \binom{\frac{n}{d}}{j} f_k^j (1 - f_k)^{\frac{n}{d}-j} \sum_{g=0}^{j} \binom{j}{g} \underline{F}^g (1 - (\underline{F}))^{j-g}),$$

where $\overline{F} = f_k + \gamma$ and $\underline{F} = f_k - \gamma$.

Proof. Since the k values do not need to be returned in any particular order, it is only necessary to compare each item in $\hat{T}$ with $\hat{f}k + 1$ and compare those in $\hat{T^{\complement}}$ with $\hat{f}k$. Let δ_{pi} denote the doubt on v_i. The following holds:

$$\delta_{pi} = \begin{cases} pv_{i,k+1}, & \text{if} \quad \hat{p}_i \leq \hat{p}_k \\ pv_{i,k} & \text{if} \quad \hat{p}_i \geq \hat{p}_{k+1} \end{cases}$$

Therefore, the top-k error is:

$$\begin{aligned} \Delta &= 1 - \prod (1 - \delta_{pi}) \\ &= 1 - \prod\nolimits_{i=1}^{i=k} (1 - pv_{i,k+1}) \prod\nolimits_{i=k+1}^{i=d} (1 - pv_{i,k}). \end{aligned} \tag{5.11}$$

For the theoretical top-k error bound, the worst-case scenario is considered, where all user frequencies are close to f_k. In other words, for any $\gamma > 0$, we have $f_i - f_{k+1} > \gamma$ (for all v_i in $\hat{T}$), and $f_k - f_j > \gamma$ (for all v_j not in $\hat{T}$). Here, the p-value for values v_k and v_j is analyzed.

Since γ is very small, the sampling count allocated to each user can be approximated by n/d. To test the null hypothesis $H_0 : f_k > f_j$, two independent binomial experiments are conducted on these two adjacent values, with: $\hat{f}k \sim \mathrm{B}(\frac{n}{d}, f_k)$ and $\hat{f}j \sim \mathrm{B}(\frac{n}{d}, f_j)$.

For each possible value of g, the probability mass function $P(\frac{n}{d}\hat{f}k - \frac{n}{d}\hat{f}j = g)$ needs to be calculated, and these are summed to obtain the p-value.

$$
\begin{aligned}
pv_{k,j} &= P(\frac{n}{d}\hat{f}_k - \frac{n}{d}\hat{f}_j \geq 0 \mid H_0) \\
&= \sum_{j=0}^{\frac{n}{d}} \binom{\frac{n}{d}}{j} f_k^j (1-f_k)^{\frac{n}{d}-j} \sum_{g=0}^{j} \binom{j}{g} f_j^g (1-f_j)^{j-g}
\end{aligned}
$$

Similarly, we obtain the p-value for $pv_{i,k+1}$:

$$
\begin{aligned}
pv_{i,k+1} &= P(\frac{n}{d}\hat{f}_i - \frac{n}{d}\hat{f}_{k+1} \geq 0 \mid H_0) \\
&= \sum_{j=0}^{\frac{n}{d}} \binom{\frac{n}{d}}{j} f_i^j (1-f_i)^{\frac{n}{d}-j} \sum_{g=0}^{j} \binom{j}{g} f_{k+1}^g (1-f_{k+1})^{j-g}
\end{aligned}
$$

Substituting $pv_{k,j}$ and $pv_{i,k+1}$ into Equ. 5.11, we obtain

$$
\begin{aligned}
&\Delta_{worst} = 1- \\
&\prod_{i=1}^{i=k}(1-\sum_{j=0}^{\frac{n}{d}} \binom{\frac{n}{d}}{j} f_i^j (1-f_i)^{\frac{n}{d}-j} \sum_{g=0}^{j} \binom{j}{g} f_{k+1}^g (1-f_{k+1})^{j-g}) \\
&\prod_{i=k+1}^{i=d}(1-\sum_{j=0}^{\frac{n}{d}} \binom{\frac{n}{d}}{j} f_k^j (1-f_k)^{\frac{n}{d}-j} \sum_{g=0}^{j} \binom{j}{g} f_j^g (1-f_j)^{j-g})
\end{aligned}
$$

Let $\overline{F}$ denote $f_k + \gamma$ and $\underline{F}$ denote $f_k - \gamma$. We use f_k solely to represent the error, which can be rewritten as:

$$
\begin{aligned}
&\Delta_{worst} = 1- \\
&\prod_{i=1}^{i=k}(1-\sum_{j=0}^{\frac{n}{d}} \binom{\frac{n}{d}}{j} \overline{F}^j (1-(\overline{F}))^{\frac{n}{d}-j} \sum_{g=0}^{j} \binom{j}{g} \underline{F}^g (1-\underline{F})^{j-g}) \\
&\prod_{i=k+1}^{i=d}(1-\sum_{j=0}^{\frac{n}{d}} \binom{\frac{n}{d}}{j} f_k^j (1-f_k)^{\frac{n}{d}-j} \sum_{g=0}^{j} \binom{j}{g} \underline{F}^g (1-(\underline{F}))^{j-g}).
\end{aligned}
$$

□

5.2.4 Frequency Estimation of Top-k Item Set

Besides the top-k discovery, this chapter further presents how to estimate their frequencies. While ARBS performs well in top-k discovery, few users are assigned to items with high frequencies, which affects the accuracy of frequency estimation. To address this issue, this chapter presents Adaptive-RR Bandit Sampling with frequency (**ARBSF**) in this chapter that can achieve accurate frequency estimation of $\hat{T}$ while also maintaining effective top-k discovery performance.

Designing of ARBSF. This chapter continues to use the ARBS algorithm to identify the top-k items, but with a modification that involves allocating some users for estimating the frequencies of items. The items in the sets $\hat{T}$ and $\hat{T^{\complement}}$ are treated separately. For an item $v_j \in \hat{T^{\complement}}$, its δ_j is computed as in the ARBS algorithm. For items $v_i \in \hat{T}$, δ_i is computed to achieve the same variance $\overline{\sigma_0}$. Specifically, the variance is first calculated based on the distribution information for items in $\hat{T}$, and the minimum variance is selected as $\overline{\sigma_0}$. The δ_i values for all items in $\hat{T}$ are set to achieve $\overline{\sigma_0}$ based on Eq. 3.5 as

$$\begin{aligned}\overline{\sigma_0} &= \frac{1}{t_i}|f_i - \hat{f}_i|^2 \\ &\leq \frac{1}{t_i}(\overline{\sigma_i}\sqrt{\frac{2log(3/\delta_i)}{t_i}} + \frac{3log(3/\delta_i)}{t_i})^2\end{aligned} \tag{5.12}$$

Then, based on all $\delta_i (1 \leq i \leq d)$, the sampling probability for v_i can be obtained according to Eq. 5.10. This modification improves the accuracy of ARBS in estimating the frequency of top-k items by allocating more users for the estimation.

Calibration and privacy analysis. The data collector receives perturbed data from all users, and counts bit "1"s among the t_i reports for item v_i. Then the frequencies of items in $\hat{T}$, denoted by $\{\hat{p}_1, \dots, \hat{p}_k\}$ can be derived by Equ. 5.13 for noise calibration. Theorem 7 proves that $\hat{p}_i$ is an unbiased frequency estimation for these items.

$$\hat{p}_i = \frac{\hat{f}_i - (1 - \frac{e^{\epsilon}}{e^{\epsilon}+1})}{(2\frac{e^{\epsilon}}{e^{\epsilon}+1} - 1)} \tag{5.13}$$

Theorem 7. *$\hat{p_i}$ in Equ. 5.13 is an unbiased estimator of the true frequency of the v_i.*

Proof.

$$
\begin{aligned}
E[\hat{p_i}] &= E[\frac{\hat{f_i} - (1 - \frac{e^\epsilon}{e^\epsilon+1})}{(2\frac{e^\epsilon}{e^\epsilon+1} - 1)}] \\
&= \frac{(p_i\frac{e^\epsilon}{e^\epsilon+1} + (1-p_i)(1-\frac{e^\epsilon}{e^\epsilon+1})) - (1-\frac{e^\epsilon}{e^\epsilon+1})}{(2\frac{e^\epsilon}{e^\epsilon+1} - 1)} \\
&= \frac{p_i\frac{e^\epsilon}{e^\epsilon+1} + 1 - p_i + p_i\frac{e^\epsilon}{e^\epsilon+1} - \frac{e^\epsilon}{e^\epsilon+1} - 1 + \frac{e^\epsilon}{e^\epsilon+1}}{(2\frac{e^\epsilon}{e^\epsilon+1} - 1)} = p_i
\end{aligned}
$$

□

All sampling methods satisfy the ϵ-LDP, if each user samples only one value with a privacy budget of ϵ.

Theorem 8. *For sampling mechanism $\mathcal{M}$, if for each user is sampled if and only one value with single value with privacy budget ϵ, the sampling algorithm $\mathcal{M}$ satisfy ϵ-LDP.*

Proof. For user i, the j-th set value is collected with privacy budget ϵ, and thus the data collector cannot obtain any valid information from the uncollected values, which is equivalent to having a privacy budget of 0. Therefore:

$$
\prod_{i=1}^{d-1} e^0 e^{-\epsilon} \leq \frac{P[\mathcal{M}(x_1, \ldots, x_d) = Y]}{P[\mathcal{M}(x'_1, \ldots, x'd) = Y]} \leq e^\epsilon \prod_{i=1}^{d-1} e^0
$$

Therefore, the following can be concluded:

$$
e^{-\epsilon} \leq \frac{P[\mathcal{M}(x_1, \ldots, x_d) = Y]}{P[\mathcal{M}(x'_1, \ldots, x'_d) = Y]} \leq e^\epsilon.
$$

Thus, $\mathcal{M}$ satisfies ϵ-local differential privacy. □

Algorithm of ARBSF. Algorithm 4 presents the ARBSF algorithm, which shares similarities with ARBS, but with three notable distinctions. Initially, in lines 5-6,

Algorithm 4: Adaptive-RR Bandit Sampling with frequency(ARBSF)

Input: Privacy budget ϵ and initialization parameter n_0

Output: Top-k frequent item set $\hat{T}$, frequency estimation $\{\hat{p_1}, \ldots, \hat{p_k}\}$

1: Initialization(n_0);

2: Update $\{P(v_1), ..., P(v_d)\}$ according to Equ. 5.10; **for** $k = n_0 + 1$ *to* n **do**

3: Sample an item v_s from $V = \{v_1, ..., v_d\}$ following probability distribution $\{P(v_1), ...P(v_d)\}$;

4: Determine the minimum variance $\overline{\sigma_0}$ of items in $\hat{T}$; **for** $k = n_0 + 1$ *to* n **do**

5: **If** $\hat{f}_i < \hat{f}_j$

6: Update δ_s according to Equ. 5.9; **else**

7: Update δ_s according to Equ. 5.12; **endfor**

8: Update $\{P(v_1), ..., P(v_d)\}$ according to Equ. 5.10;

9: Find $\hat{T} \subset V$ such that $|\hat{T}| = m$, and $\forall v_i \in \hat{T}, \forall v_j \in (V - \hat{T}) : \ (\hat{f}_i \geq \hat{f}_j)$. **endfor**

10: Calculate $\hat{p_i}$ for $i \in \{1, \ldots, k\}$ according to Equ. 5.13;

11: **return** $\hat{T}, \{\hat{p_1}, \ldots, \hat{p_k}\}$

$\overline{\sigma_t}$ is determined and subsequently δ_i is adjusted to allocate more users to the top-k items. Secondly, in line 9, it is necessary to estimate $\hat{p_i}$ for items in $\hat{T}$. In the end, the algorithm at line 10 returns the items in $\hat{T}$, and their corresponding frequencies $\hat{p_i}$.

5.2.5 Large-Scale Solution

When dealing with a large domain d (e.g., tens of thousands of items), the data collector first needs to focus on a narrower candidate set value range. This is essential as sampling methods cannot accurately capture valid information from the entire set value range. To achieve this, users are evenly divided into two groups for efficient processing. The first group is tasked with pruning the large domain into a smaller range, while the second group performs adaptive sampling methods within this reduced range.

Specifically, each user in the first group selects a value randomly from their private value set and perturbs it using a perturbation mechanism such as OUE, OLH, etc. This process identifies the 10k most frequent items from d.

While this process introduces a bias in frequency estimation, at this initial phase, our primary goal is only to roughly identify the top $10k$ items without focusing too much on precise frequency estimations. Additionally, this procedure satisfies ϵ-LDP, which is proven in Theorem 8. It is important to note that this phase is unnecessary if the original domain size is close to or fewer than $10k$ items. In the experiments, OUE was utilized, and the results were then reported.

5.3 Delay-Constrained Solution

In traditional MAB problems, the interaction between the gambler and the arms is relatively straightforward and short-lived. However, in the LDP setting, the interaction between users and the data collector can be much more prolonged. This means that, unlike traditional MAB problems, the multiple rounds of communication between users and the data collector can be time-consuming, which becomes a new challenge for both Adaptive Recommender Bandit Selection (ARBS) and Adaptive Recommender Bandit Selection with Feedback (ARBSF). On the other hand, in each round, a single user may only contribute limited information.

To address these challenges, this chapter proposes a batch processing method, which can help reduce the system delay. By processing user contributions in batches, the issues caused by the prolonged interaction between users and the data collector in the LDP setting can be mitigated.

5.3.1 Delay-Constrained Solution

To reduce communication overhead between users and the data collector, an intuitive solution is to let a batch of users send reports in a round, thus reducing the total rounds needed. However, simply allocating the same number of users in each round is not an optimal solution, as the impact of the data collected in earlier rounds is greater than that of the later rounds. A more effective sampling scheme is to allocate fewer users in earlier rounds to allow for timely adjustments, while ensuring sufficiently accurate information collected from these rounds. Striking a balance between these trade-offs is critical in developing an optimal sampling scheme. This chapter proposes the Delay-contrained Batch Sampling algorithm (**DBS**), which models and derives an optimal user allocation scheme when the number of rounds is fixed to R. RBS guarantees both high accuracy of top-k discovery and low communication overhead by dynamically allocating users based on each user's potential to contribute to the discovery of top-k items.

User allocation in each round. Let n_i denote the number of users in round i and N_r denote the number of remaining users after $r-1$ rounds, which can be written as:

$$N_r = n - n_0 - \sum\nolimits_{i=1}^{r-1} n_i.$$

As shown in Equ. 5.9, δ_i represents the inverted error for v_i. Let e^I_{r-1} denote the cumulative error obtained from δ_i after the $r-1$ rounds:

$$e^I_{r-1} = \sum\nolimits_{i=1}^{d} \delta_i. \tag{5.14}$$

The ideal user allocation scheme would be to minimize the overall error over all rounds by dynamically assigning a batch of users to each round. However, this information can only be obtained after each round of data collection, which is not feasible. Therefore, a greedy search is applied to determine the optimal number of next-round users based on the current-round information. Specifically, the r-th round and the $(r+1)$-th round are combined and split into two virtual phases to simulate the upcoming sampling process. In phase I, n_r^I users are collected to simulate the occurrence of round r, and in phase II, n_r^{II} users are collected to simulate the occurrence of round $r+1$. The total number of users for these two virtual phases is denoted by $n_{(r,r+1)}$, and the following holds:

$$n_{(r,r+1)} = n_r^I + n_r^{II} \tag{5.15}$$

Let t_i represent the number of reports from users for v_i before the virtual phase I, and it changes to t_i' after the virtual phase I and can be calculated as

$$t_i' = t_i + n_r^I P(v_i),$$

where $P(v_i)$ can be calculated by Equ. 5.10.

Since the virtual process is not actually implemented, the variance of each item in virtual phase I cannot be obtained. Therefore, the error after virtual phase I is processed by using the inverse of Hoeffding's inequality instead of Bernstein's inequality, as:

$$\delta_i' = \frac{2}{exp(2t_i'(\hat{f}_i - \mu)^2)}, \tag{5.16}$$

where $\hat{f}_i$ and μ are set based on the latest real knowledge obtained in round $r-1$.

Similar to Eq. 5.14, the cumulative error e_{r-1}^{II} after virtual phase I can be simulated based on δ_i'. Specifically,

$$e_{r-1}^{II} = \sum_{i=1}^{d} \delta_i'.$$

Theorem 9. *e_{r-1}^{II} is a theoretical lower bound, such that all items simultaneously satisfy $|\hat{f}_i - \mu| \leq B_i$, where $B_i = \sqrt{\frac{2\log(2/\delta_i')}{t_i'}}$.*

Proof. The error bound reflects our overall requirement for all items to simultaneously satisfy the condition $|\hat{f}_i - \mu| \leq B_i$, therefore:

$$P(|\hat{f}_1 - \mu| \leq B_1, |\hat{f}_2 - \mu| \leq B_2, ..., |\hat{f}_d - \mu| \leq B_d) \geq 1 - \delta,$$

where δ is the overall error tolerance, which represents our required confidence level for the joint probability. And $B_i = \sqrt{\frac{2\log(2/\delta_i')}{t_i'}}$.

According to the Boole inequality, the joint probability is bounded from below by one minus the sum of the individual probabilities. Specifically, therefore:

$$P(|\hat{f}_1 - \mu| \leq B_1, |\hat{f}_2 - \mu| \leq B_2, ..., |\hat{f}d - \mu| \leq B_d) \geq 1 - \sum_{i=1}^{d} \delta_i'.$$

□

To find the optimal number of users allocated to the current round, the problem is formalized as minimizing the cumulative user error, i.e., the sum of cumulative errors for certain users, in the two virtual phases. Specifically, in phase I, n_r^I users' data are collected with cumulative error e_{r-1}^I, while n_r^{II} users' data are collected with cumulative error e_{r-1}^{II} in phase II. The cumulative user error for n_r^I and n_r^{II} users can be expressed as:

$$\begin{aligned} &\arg\min_{n_r^I} n_r^I e_1^{r-1} + n_r^{II} e_2^{r-1}, \\ \text{s.t.} \quad &sup(n_{(r,r+1)}) = \frac{2N_r}{R-(r-1)} \end{aligned} \tag{5.17}$$

Here, $sup(n_{(r,r+1)})$ denotes the upper bound on the combined number of users across the two virtual phases. This condition can be inferred from the following equation according to $n_r < n_{r+1}, r \in \{1, \ldots, R-1\}$ established in Theorem 10:

$$\begin{aligned} n_{(r,r+1)} &\leq n_r + n_{r+1} \\ &\leq \frac{N_r}{R-(r-1)} + \frac{N_r - \frac{N_r}{R-(r-1)}}{R-(r-1)} \\ &= \frac{2N_r}{R-(r-1)}. \end{aligned}$$

According to Eq. 5.17, a numerical solution for n_r^1 is obtained, which represents the number of users allocated to round r. The number of users allocated to round $r+1$

can be estimated using either n^1_{r+1} from phase I results or n^2_r from phase II results in round r. n^1_{r+1} is selected as it has aggregated more information. Formally, we have:

$$n_r = n_r^1 \tag{5.18}$$

Algorithm of DBS. The batchwise procedure, as presented in Algorithm 5, can be summarized as follows. Specifically, when a new round r commences, the number of users n_r allocated to the current round is determined (line 4). Subsequently, n_r users are collected based on the current $P(v_i)$ (lines 5-6). The frequency estimation $\hat{f}_i$, the inverted error, and sampling probability $P(v_i)$ are independently updated at the end of each round in both ARBS and ARBSF (lines 7-9).

Algorithm 5: Delay-contrained Batch Sampling (DBS)

Input: Privacy budget ϵ, initialization parameter n_0 and number of round R

Output: Top-k frequent item set $\hat{T}$, frequency estimation $\{\hat{p_1}, \ldots, \hat{p_k}\}$

1: Initialization(n_0);

2: Update $\{P(v_1), ..., P(v_d)\}$ according to Equ. 5.10; **for** $r = 1$ *to* R **do**

3:

    Calculate n_r according to Equ. 5.17 and 5.18; **for** $i = 1$ *to* n_r **do**

4:

    Sample an item v_s from $V = \{v_1, ..., v_d\}$ following probability distribution $\{P(v_1), ...P(v_d)\}$; **endfor**

5: **ARBS**: Update δ_s according to Equ. 5.9;

6: **ARBSF**: Determine the minimum variance $\overline{\sigma_0}$ of the items in $\hat{T}$; Update δ_s according to Equ. 5.9 or Equ. 5.12;

7: Update $\{P(v_1), ...P(v_d)\}$ according to Equ. 5.10;

8: Find $\hat{T} \subset V$ such that $|\hat{T}| = m$, and $\forall v_i \in \hat{T}\ \forall v_j \in (V - \hat{T}) :\ (\hat{f}_i \geq \hat{f}_j)$. **endfor**

9: Calculate $\hat{p_i}$ for $i \in \{1, \ldots, d\}$ according to Equ. 5.13.

10: **return** **ARBS**: $\hat{T}$;

11: **ARBSF**: $\hat{T}, \{\hat{p_1}, \ldots, \hat{p_k}\}$

Monotonicity analysis. In DBS, n_r progressively increases with each round. This growth pattern results in a smaller cumulative user error for all users compared to other sampling orders, as demonstrated in the following theorem:

Theorem 10. *Given a set of numbers of sampled users in each round $n_1, ..., n_R$, the ordering that satisfies the inequality*

$$n_1 < \cdots < n_r < \cdots < n_R.$$

guarantees an optimal cumulative user error for all users.

Proof. With each round, the number of collected users increases, which leads to a continuous decrease in the error. Recall e_1^{r-1} denotes the cumulative error for all items after $r-1$ rounds (same as Equ. 5.14). This establishes the following inequality:

$$e_1^{r-1} > e_1^r, r \in \{1, \ldots, R\}.$$

In a similar manner to Equ. 5.17, which defines the cumulative user error for all users, we can propose the following equation:

$$M = n_1 e_1^0 + n_2 e_1^1 + ... + n_{R-1} e_1^{R-1}. \tag{5.19}$$

For every choice of real numbers $n_1 < n_2 < \cdots < n_R$ and $n_{\xi_1}, n_{\xi_2}, \ldots, n_{\xi_R}$ is a permutation of $n_1, ..., n_R$, the rearrangement inequality [57] leads us to the following inequality:

$$\begin{aligned} n_R e_1^0 + n_{R-1} e_1^1 + ... + n_1 e_1^{R-1} > \\ n_{\xi_1} e_1^0 + n_{\xi_2} e_1^1 + \cdots + n_{\xi_h} e_1^R > \\ n_1 e_1^0 + n_2 e_1^1 + \cdots + n_R e_1^{R-1}. \end{aligned}$$

Therefore, it is deduced that in order to achieve the minimal error for Eq. 5.19, the sampling sequence should satisfy

$$n_1 < \cdots < n_r < \cdots < n_R.$$

□

5.4 Experimental Results

5.4.1 Experimental Setup

This chapter conducts evaluations of the proposed schemes, ARBS and ARBSF, on both synthetic and real-world datasets to validate their accuracy. For comparison purpose, the results of uniform sampling (i.e., each user randomly selects and perturbs one item), SVIM (based on either GRR or OLH) [115] and Mwheel (designed for multiset frequency estimation) [138] are also shown. The experiments are conducted on a PC equipped with an AMD Ryzen 7 2700X eight-core processor, 64GB RAM, and Windows 10, using MATLAB R2019b and Python 3.10. All datasets and code are available online[1].

5.4.1.1 Experiment Design

The experiments are repeated 50 times and then averaged. The following are some parameters related to the performance of the adaptive schemes, and this chapter will conduct experiments to study their impacts on ARBS and ARBSF.

1. The number of rounds R. Due to the delay limitation, the aim is to establish whether satisfactory results can be achieved with a relatively small number of rounds.

2. Item frequency distribution. The performance of top-k estimation relies on the item frequency, and the lower the frequency is, the smaller the estimated error becomes.

3. Privacy budget ϵ. A larger ϵ leads to less perturbation noise and thus more accurate estimation result.

[1] https://github.com/RONGDUGithub/ARBSF

4. The size of frequent item set k. The proposed adaptive schemes should reveal the robustness for identifying frequent item set with varying k.

5. Data sparsity s. It affects the final experimental results, and the scheme is expected to perform better when the data are not very sparse.

5.4.1.2 Utility Metrics

This chapter applies normalized cumulative rank (NCR) and mean square error (MSE) to measure the accuracy of top-k item discovery and frequency estimation, respectively.

- NCR [115] is proposed to reveal the true rank information of the top-k values. For each item v_i, the ranked value V_i as

  $$V_i = \frac{q(v_i)|v_i \in T}{q(v_i)|v_i \in \hat{T}} = \frac{q(v_i)|v_i \in T}{k(k+1)/2}$$

  where $q(\cdot)$ is a quality function defined as

  $$q(v_i) = \begin{cases} k - i + 1, & \text{if} \quad v_i \in T \\ 0, & \text{if} \quad v_i \in \hat{T} \end{cases}$$

  Then NCR is calculated based on the ranked values across all items. Formally,

  $$NCR = \frac{\sum_{v_i \in \hat{T}} V_i}{\sum_{v_j \in T} V_j}.$$

- MSE [82] measures the frequency estimation accuracy in terms of squared errors as

  $$MSE = \frac{1}{k} \sum\nolimits_{v_i \in T} (p_i - \hat{p}_i)^2,$$

  where the estimated frequencies $\hat{p}_i$ are set to 0 for items that are not successfully identified by the protocol.

Table 5.1: Statistics of datasets

Dataset	Number of users	Domain	Minimum Count	Maximum Count
Linear	10000	100	5	28
Beta	10000	100	2	27
Gamma	100000	10000	436	641
Laplace	100000	9883	29	92
$World_{google}$	10000	26	1	13
$World_{mit}$	10000	26	1	7
Retail	540455	2602	1	7
Kosarak	990002	41270	1	2498

5.4.1.3 Datasets

The experiments are conducted over four synthetic datasets and four real datasets. The dataset statistics are detailed in TABLE 5.1. The last three columns are the minimum, maximum and average numbers of items possessed by each user.

Synthetic Datasets. This chapter generates two synthetic datasets with 10,000 users and 100 items. Users' item ownership follows different distributions: **Linear** and **Beta**. Additionally, this chapter generates two large-scale datasets with 100,000 users and 10,000 items, where users' item ownership follows **Gamma** and **Laplace** distributions, respectively.

Real Datasets. This chapter uses four publicly available set-valued datasets as follows. **Word**$_{google}$[2] contains the 10,000 most frequent English words sorted by frequency, and this chapter estimates the letter frequency among these words. **Word**$_{mit}$ is similar to **Word**$_{google}$ and contains 10,000 words from 26 letters. **Retail**[3] contains all the transactions occurring between 2010 and 2011 for a UK-based and registered non-store online retail, including merchant transactions for half a million users in

[2] https://github.com/first20hours/google-10000-english

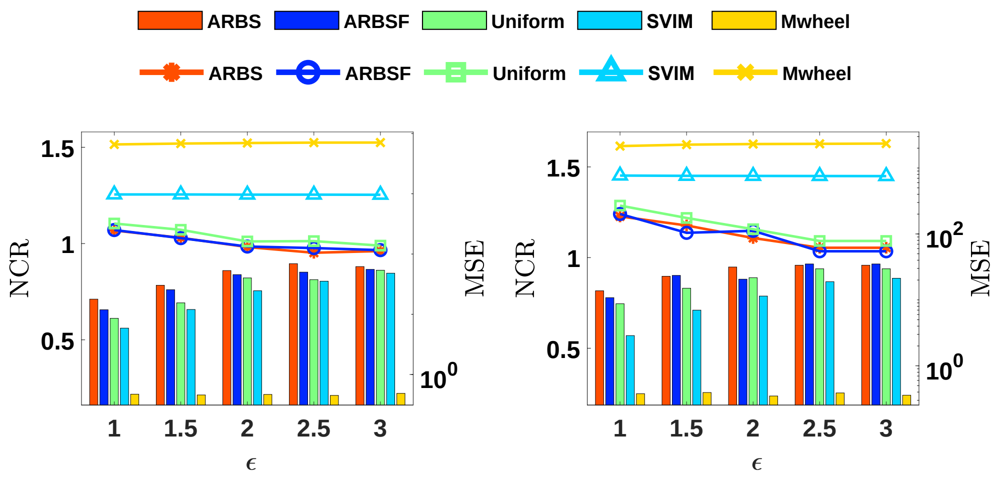
[3] https://archive.ics.uci.edu/dataset/352/online+retail

2,603 categories. **Kosarak**[4] contains click streams on a Hungarian website that contains around one million users and 42 thousand categories.

The impact of ϵ. Fig. 5.2 illustrates the performance of various algorithms in terms of NCR and MSE by varying ϵ, with k fixed at 30 for all datasets. The histograms show the NCR for top-k discovery, while the curves indicate the MSE for frequency estimation. The results indicate that a larger ϵ leads to a higher NCR and a lower MSE across all five algorithms. In most cases, ARBS and ARBSF outperform SVIM, Mwheel and uniform sampling, achieving much higher NCR and lower MSE. This is because SVIM introduces a large error when the dataset is not sparse, leading to poor performance. Furthermore, the adaptive sampling schemes of ARBS and ARBSF perform better than uniform sampling by constantly learning and applying new knowledge to update their sampling schemes.We can observe that Mwheel performs poorly across all datasets. This is because Mwheel utilizes principles similar to PEM [116] to find the corresponding top-k data, and PEM tends to perform poorly for small ranges of d. Moreover, this scheme is designed to handle multi-value sets, where the same set may contain duplicate data, and it may perform better in such scenarios. Although our set-value is a special case of multi-value sets, it cannot leverage the advantages of Mwheel. In terms of NCR, ARBS generally performs better than ARBSF, while ARBSF performs slightly better in terms of MSE. This suggests that the two adaptive schemes have different objectives. Overall, these results suggest that adaptive sampling algorithms such as ARBS and ARBSF can improve the utility of differentially private data analysis tasks, even for large-scale datasets.

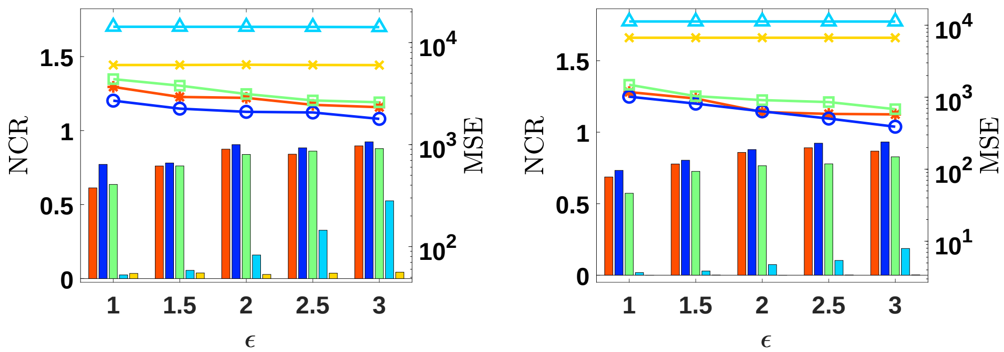
In addition to NCR and MSE, this chapter also uses another metric for measuring accuracy: the hit rate [101], as shown in TABLE 5.2. The hit rate is used to describe the proportion of correctly identified or classified instances. Unlike NCR, which focuses on the ranking of accurate values, the hit rate is concerned with the quantity of accurate values. From the data presented in the table, it is evident that our adap-

[4] https://github.com/cpearce/HARM/blob/master/datasets/kosarak.csv



(a) **Linear**, $k = 30$.

(b) **Beta**, $k = 30$.



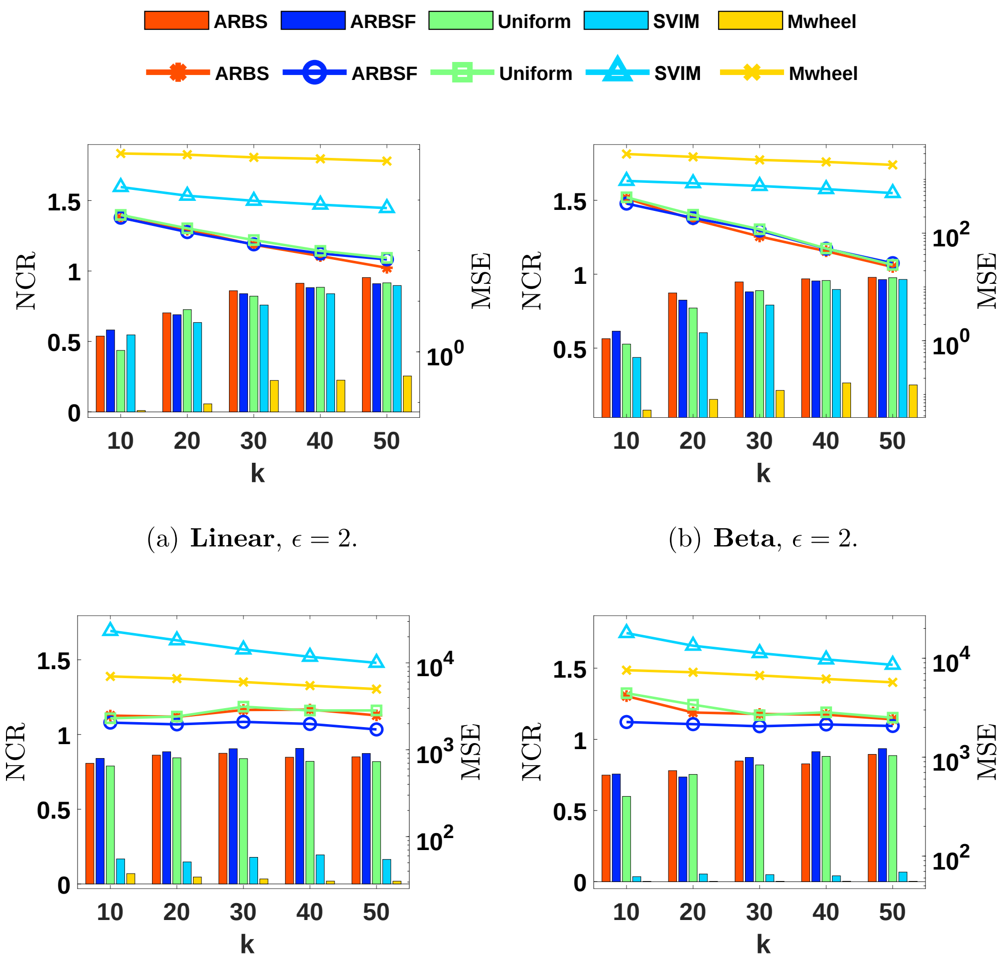
(c) **Gamma**, $k = 30$.

(d) **Laplace**, $k = 30$.

Figure 5.2: The results of NCR & MSE w.r.t. ϵ on synthetic datasets

tive sampling methods achieve a higher hit rate. In most cases, ARBSF outperforms ARBS, and its performance is generally consistent with NCR. Moreover, the larger the value of ϵ, the higher the hit rate.

The impact of k**.** To investigate the impact of k on the performance of different algorithms, this chapter conducts an experiment where the chapter varies k while keeping ϵ fixed at 2. Fig. 5.3 illustrates that all algorithms achieve better MSE as k increases, since more low-frequency items are estimated and the distance between the estimated frequency and the actual frequency becomes smaller. The performance of the NCR algorithm is heavily influenced by different datasets. For the **Beta**



(a) **Linear**, $\epsilon = 2$.

(b) **Beta**, $\epsilon = 2$.

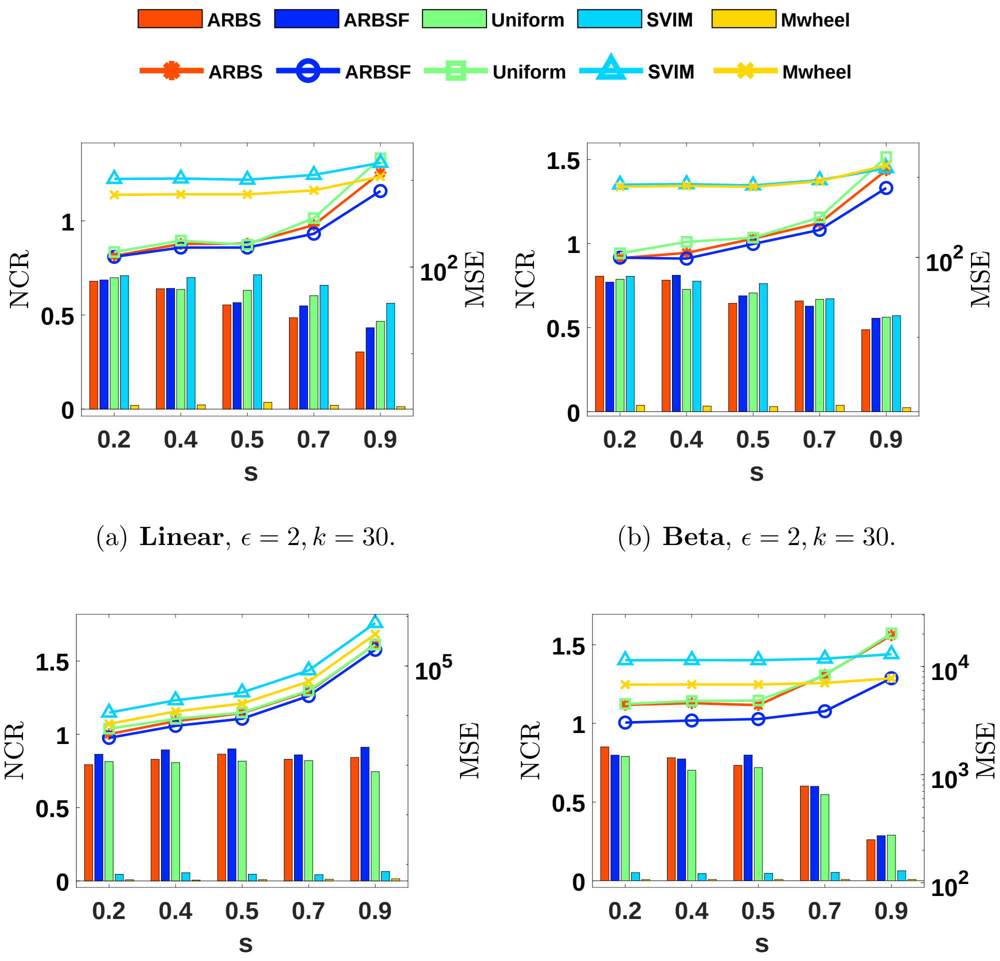
(c) **Gamma**, $\epsilon = 2$.

(d) **Laplace**, $\epsilon = 2$.

Figure 5.3: The results of NCR & MSE w.r.t. k on synthetic datasets

dataset, where the frequency difference between items is constant, the difficulty of distinguishing the items remains the same, and the NCR for ARBS and ARBFS does not change significantly with respect to k.

The impact of s**.** Fig.5.4 demonstrates the impact of dataset sparsity s from 0.2 to 0.9. Here, as s increases, the data becomes more sparse. When $s = 0.2$, 20% of the data in the original dataset is removed, resulting in sparsity. As expected, as s increases, the utility (measured by both NCR and MSE) decreases, indicating that all algorithms perform worse when the dataset is sparse. However, since both the original data and estimated amounts decrease, leading to a smaller interpolation, the



(a) **Linear**, $\epsilon = 2, k = 30$.

(b) **Beta**, $\epsilon = 2, k = 30$.

(c) **Gamma**, $\epsilon = 2, k = 30$.

(d) **Laplace**, $\epsilon = 2, k = 30$.

Figure 5.4: The results of NCR & MSE w.r.t. s on synthetic datasets

Table 5.2: Hit rate of real datasets

Dataset	Scheme	$\epsilon = 1$	$\epsilon = 1.5$	$\epsilon = 2$	$\epsilon = 2.5$	$\epsilon = 3$
linear	ARBS	0.68	0.74	0.8	0.83	0.82
	ARBSF	0.65	0.72	0.79	0.79	0.81
	Uniform	0.58	0.63	0.76	0.74	0.78
	SVIM	0.51	0.60	0.7	0.73	0.78
	Mwheel	0.23	0.23	0.22	0.21	0.22
Beta	ARBS	0.77	0.82	0.87	0.91	0.91
	ARBSF	0.73	0.85	0.84	0.91	0.91
	Uniform	0.71	0.77	0.85	0.89	0.89
	SVIM	0.55	0.67	0.76	0.83	0.85
	Mwheel	0.25	0.24	0.24	0.25	0.25
Gamma	ARBS	0.54	0.67	0.75	0.83	0.77
	ARBSF	0.66	0.80	0.83	0.88	0.81
	Uniform	0.58	0.62	0.68	0.78	0.75
	SVIM	0.02	0.05	0.12	0.25	0.41
	Mwheel	0.07	0.07	0.05	0.06	0.07
Laplace	ARBS	0.62	0.67	0.78	0.79	0.80
	ARBSF	0.67	0.75	0.77	0.82	0.86
	Uniform	0.53	0.67	0.69	0.73	0.76
	SVIM	0.02	0.03	0.05	0.08	0.14
	Mwheel	0.001	0.001	0.002	0.002	0.003

MSE might decrease. To enable a fair comparison of MSE across different values of s, this chapter proposes a normalization of MSE.

Specifically, assume that the original frequency of a dataset is f, and the estimated frequency is $\hat{f}$. Given s, the expected estimate should be $(1-s)f$, and if the actual estimate is denoted by $(1-s)\hat{f}$, the difference in the MSE expressions before and after normalization is:

$$\begin{aligned} MSE &= \frac{1}{n}\sum_{i=1}^{n}((\hat{f}_i) - f_i)^2 = \frac{1}{(1-s)^2}\frac{(1-s)^2}{n}\sum_{i=1}^{n}((\hat{f}_i) - f_i)^2 \\ &= \frac{1}{(1-s)^2}\frac{1}{n}\sum_{i=1}^{n}(((1-s)\hat{f}_i) - (1-s)f_i)^2 \\ &= \frac{1}{(1-s)^2}MSE_s \end{aligned} \tag{5.20}$$

where MSE_s is the experimental result, and our normalization is to multiply the calculated experimental result MSE_s by $1/(1-s^2)$.

The experimental results show that as the data becomes sparser, NCR decreases, and MSE increases. This is because a sparse dataset has fewer available data points for each user, making it more difficult to estimate the user's preference accurately. In most cases, our proposed schemes outperform the others under different s values. Although SVIM shows better NCR performance on the **linear**, its MSE is still worse than that of our methods.

5.4.2 Overall Results on Real Datasets

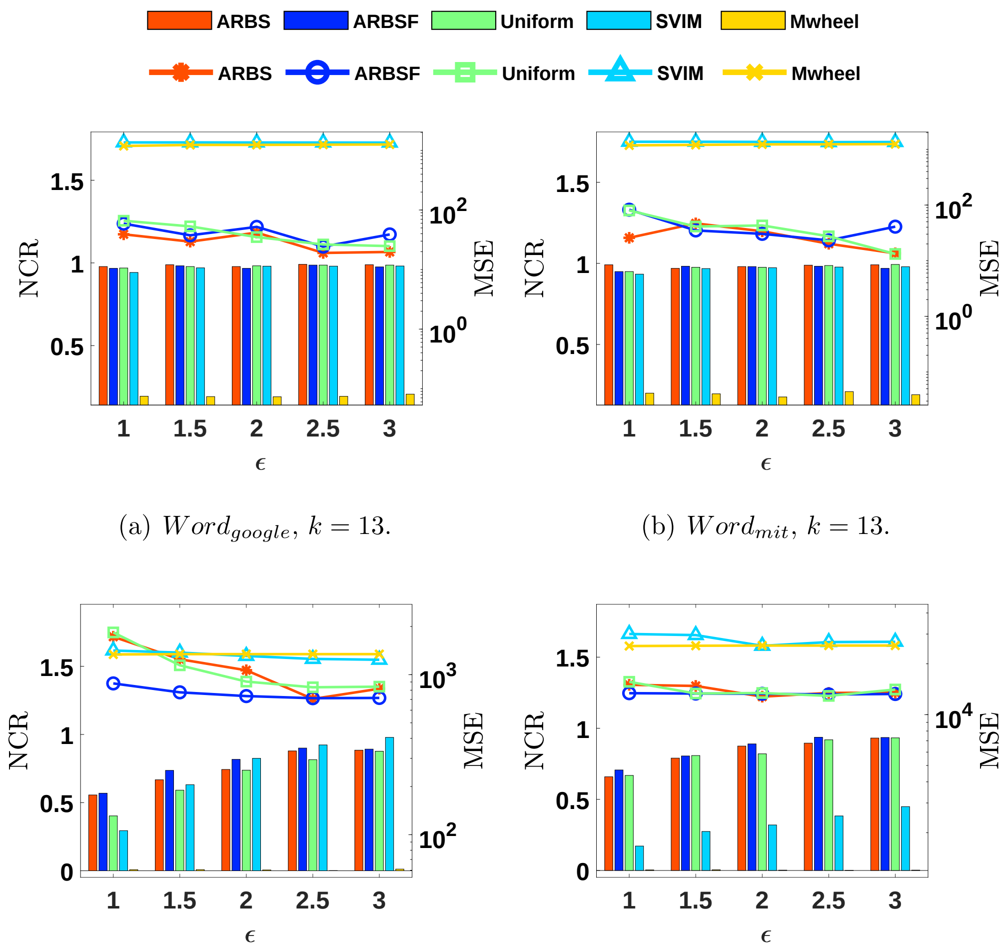
This chapter conducts experiments on the estimation of top-k items using four real datasets with varying ϵ and k. The bar charts represent the NCR results, while the line plots represent the MSE results. To evaluate the performance on **Retail** and **Kosarak**, this chapter first prunes the domain, and the specific process can be referred to in Chapter 5.2.5.

The impact of ϵ. Fig. 5.5 depicts the accuracy of frequency estimation with respect to ϵ. Consistent with the synthetic data, an improvement in utility is observed as ϵ increases. It can be observed that for the dataset, Mwheel performs the worst. Mwheel's set domain length is an exponential of 2, which is different from the domain length of the data. After converting the data into binary, values with the same lower bits may be recovered simultaneously, leading to inaccurate results. Furthermore, the dataset does not have repeated values within a single set, so Mwheel cannot take advantage of its strengths. Even in Figures 5.5 (c)-(d), its NCR values are close to 0. Additionally, for small-scale datasets, the query NCR results of other methods are similar. However, for the large-scale **Retail** and **Kosarak** dataset, the adaptive schemes generally perform better than other approaches. This proves the effectiveness of the handling of large-scale data in Chapter 5.2.5.

Regarding MSE, it can be observed that ARBSF performs optimally in most cases, as this method allocates a portion of the sampling budget to the discovered top-k values, thus outperforming ARBS. Furthermore, the sampling-based schemes achieve better MSE than SVIM because the methods guarantee unbiasedness, resulting in more accurate frequency estimates.

The impact of k. The Fig. 5.6 presents the NCR and MSE values by varying k, and our adaptive algorithms demonstrate superior performance in most cases. The MSE consistently decreases as k increases, while the NCR performance is influenced by the characteristics of the dataset. In particular, for $\textbf{Word}_{google}$ and $\textbf{Word}_{mit}$, NCR improves as k increases. However, for **Retail** and **Kosarak**, NCR worsens as k increases. This discrepancy arises because the first two datasets have relatively uniform data frequency distributions. In contrast, for the latter two datasets, as k increases, the density of data points around k also increases, making it more challenging to accurately distinguish the true top-k values, thereby leading to a decline in NCR performance.

In Fig. 5.6 (c), when $k = 10, 40, 50$, SVIM achieves the best NCR performance. However, as k decreases, the proposed schemes outperform SVIM. This indicates that while SVIM is not as proficient as the proposed schemes in estimating the counts for extremely high-frequency values. Additionally, since SVIM is not an unbiased estimator, its corresponding MSE performance is also inferior to that of the proposed schemes.



(a) $Word_{google}$, $k = 13$.

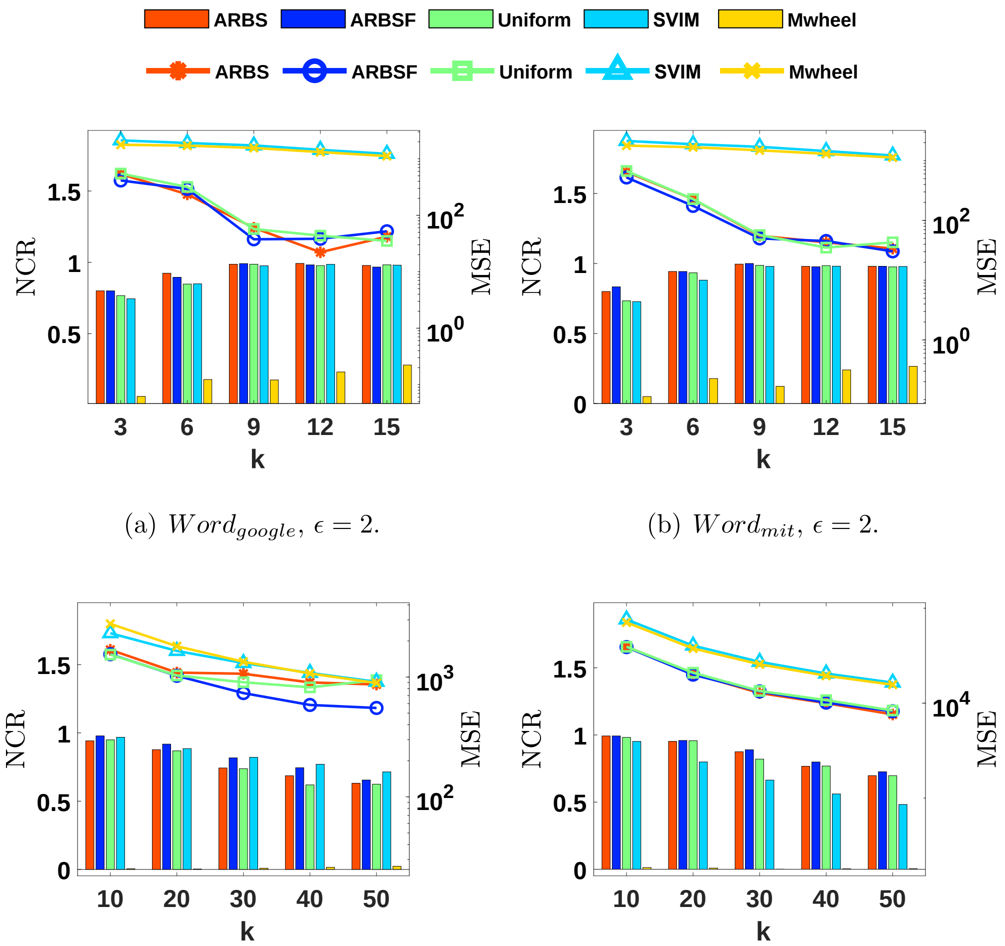
(b) $Word_{mit}$, $k = 13$.

(c) **Retail**, $k = 30$.

(d) **Kosarak**, $k = 30$.

Figure 5.5: The results of NCR & MSE w.r.t. ϵ on real-world datasets.

The hit rate on real datasets with varying k is also tested, and the results are shown in TABLE 5.3. Consistent with the NCR results, in most cases, our adaptive sampling method yielded superior results. However, the trend of the hit rate with varying k is related to the distribution of the dataset itself. When there are more items with frequencies similar to the k-th item, the accuracy tends to be lower, and vice versa. For the **Retail** and **Kosarak** datasets, as k increases, the frequency of items near the k-th item is lower but there are more such items, resulting in a lower hit rate accuracy.



(a) $Word_{google}$, $\epsilon = 2$.

(b) $Word_{mit}$, $\epsilon = 2$.

(c) **Retail**, $\epsilon = 2$.

(d) **Kosarak**, $\epsilon = 2$.

Figure 5.6: The results of NCR & MSE w.r.t. k on real-world datasets.

Table 5.3: Hit rate of synthetic datasets

Dataset	Scheme	k 3(10)	k 6(20)	k 9(30)	k 12(40)	k 15(50)
$Word_{google}$	ARBS	0.73	0.87	0.937	0.95	0.91
	ARBSF	0.80	0.83	0.96	0.92	0.89
	Uniform	0.73	0.83	0.96	0.92	0.91
	PFSO	0.66	0.79	0.95	0.93	0.94
	Mwheel	0.08	0.198	0.26	0.27	0.29
$Word_{mit}$	ARBS	0.67	0.87	0.98	0.90	0.92
	ARBSF	0.73	0.90	1.00	0.90	0.92
	Uniform	0.67	0.87	0.93	0.93	0.89
	PFSO	0.64	0.82	0.96	0.90	0.93
	Mwheel	0.05	0.21	0.23	0.30	0.28
Retail	ARBS	0.86	0.74	0.58	0.545	0.48
	ARBSF	0.94	0.81	0.65	0.6	0.51
	Uniform	0.86	0.75	0.58	0.47	0.47
	PFSO	0.90	0.78	0.67	0.61	0.55
	Mwheel	0.006	0.006	0.008	0.013	0.020
Kosarak	ARBS	0.96	0.84	0.71	0.60	0.52
	ARBSF	0.98	0.87	0.79	0.63	0.56
	Uniform	0.92	0.86	0.65	0.60	0.52
	PFSO	0.89	0.63	0.46	0.37	0.30
	Mwheel	0.015	0.013	0.003	0.009	0.010

5.4.3 Performance of Adaptive Schemes

Finally, the performance of the adaptive schemes is studied in terms of sampling times allocation, error and the number of rounds. For a more intuitive understanding, the schemes are mainly executed on **Linear** 45°, where there are 10,000 users with 100 items and the frequency monotonically increases at equal intervals from 0 to 1. Moreover, the experiments are also executed on data following the Beta(2,5) distribution.

5.4.3.1 Time Analysis

In Table 5.4, the time complexity across different datasets is analyzed. Since the interaction process of the data collector cannot be accurately simulated, the numbers of interaction between a user and the data collector are used as a basic unit, and the performance for different schemes is analyzed. If the sampling probability distribution is the same, multiple users can interact with the data collector simultaneously, which can be considered a single interaction.

Without batch operations, the data collector needs to adjust the sampling probability after collecting data from each individual user, so the number of interactions equals the number of users. This means users cannot upload data synchronously because each user must wait for the previous user to finish uploading before the data collector updates the sampling probability, making the process excessively time-consuming. However, with the batch scheme, the data collector can collect data from a group of users simultaneously and then update the sampling probability before interacting with the next group of users. This approach significantly reduces the time complexity by enabling synchronous data uploads.

This chapter then discusses the number of interactions for different datasets in two categories: when the dataset range is relatively small, such as **Linear** and **Beta**, the number of interactions is equal to R; when the dataset range is relatively large, such as **Retail** and **Kosarak**, the number of interactions is $R+1$. According to Chapter 5.2.5, half of the users will first interact with the data collector to prune the domain. These users can interact with the data collector simultaneously, and then the remaining interactions are added, resulting in a total of $R+1$ interactions.

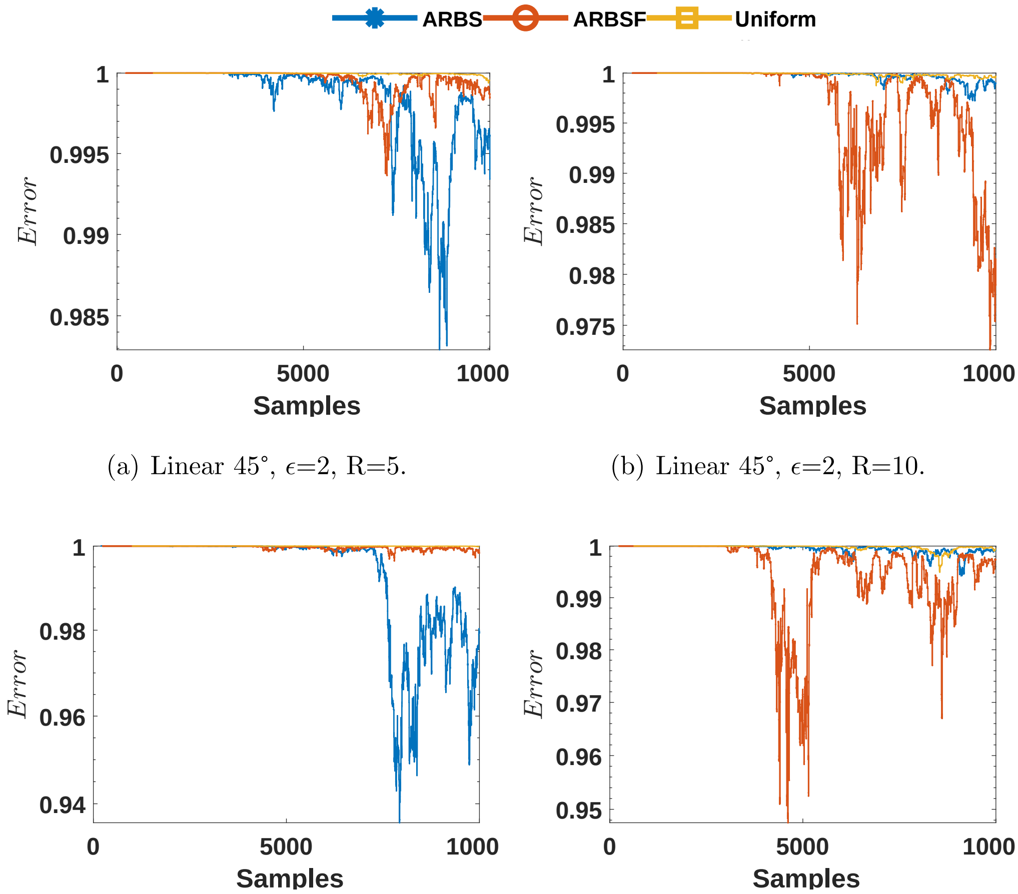
Table 5.4: Number of interactions for different datasets

Dataset	No Batch	Round 5	Round 10	Round R
Linear	10000	5	10	R
Beta	10000	5	10	R
gamma	100000	6	11	$R+1$
laplace	100000	6	11	$R+1$
$Word_{google}$	10000	5	10	R
$Word_{mit}$	10000	5	10	R
Retail	270229	6	11	$R+1$
Kosarak	990002	6	11	$R+1$

5.4.3.2 Top-k Discovery Error Δ

In Fig. 5.7, a one-sided two-sample T-test is utilized to assess the error Δ in the top-k discovery, as mentioned in Theorem 6. Since only ARBS, ARBSF, and Uniform collect data from users one by one or across multiple rounds, their frequency distributions can be dynamically captured. For the other methods, data is collected in a one-time process, and aggregation can only be performed after all data has been collected. A higher Δ is indicative of a decrease in the estimation's accuracy. The yellow line represents the performance of the uniform sampling technique, which exhibits the highest error among the sampling algorithms considered. Furthermore, the error decreases slowly as the number of users increases. On the other hand, the ARBS and ARBSF schemes, shown in blue and red, respectively, exhibit lower error and decrease dramatically as the number of collected users increases. It is important to note that the error does not always decrease with an increasing number of collected users, as $\hat{p}_k$ and $\hat{p}_{k+1}$ may change during the data collection process. However, the error gradually reduces when p_k and p_{k+1} remain constant. Our scheme adapts to changing probabilities and increasing information by dynamically updating p_k and

p_{k+1}, providing both flexibility and accuracy.



(a) Linear 45°, ϵ=2, R=5.
(b) Linear 45°, ϵ=2, R=10.
(c) Beta(2,5), ϵ=2, R=15.
(d) Beta(2,5), ϵ=2, R=20.

Figure 5.7: The results of error estimated by T-test.

5.4.3.3 Impact of Number of Rounds

Fig. 5.8 shows the impact of the number of rounds (i.e., 5, 10, 15, and 20) on the overall results. A continuous increase in accuracy with increasing ϵ is observed, as evidenced by the gradual increase in NCR and decrease in MSE. However, the effect of increasing the number of rounds on NCR is not particularly significant, especially for larger privacy budgets (e.g., $\epsilon > 2$). In fact, as the number of rounds increases, the final result may not necessarily exhibit a monotonic increase in accuracy. This is due to the data-dependent nature of the experiments and the potential variation in the number of users assigned to each round. Such differences can impact the learning outcomes in each round and ultimately affect the final accuracy. However, it is observed that MSE with more rounds generally performs better. If there are no constraints on communication overhead, increasing the number of rounds would enhance accuracy.

5.4.3.4 Impact of Allocated User Number Per Item

The histograms in Fig. 5.9 depict the number of users allocated to different items by the ARBS and the ARBSF. The items are sorted from the smallest to the largest, and k is set to 30. The yellow line representing uniform sampling is flat, indicating that naive sampling treats all items equally. The blue curve represents ARBS and includes a small peak around the 70-th item, which indicates that more sampling probability is given to the items nearby the k-th item. The red curve represents the ARBSF scheme, and rises between the items of 70 and 100, illustrating how the aim is to achieve higher statistical frequency accuracy while ensuring that the top-k discovery is generally correct. The experimental results are consistent with the intention described in Chapter 5.2.

5.4.3.5 Impact of Allocated User Number per Round

In Fig. 5.10, it can be observed that the number of collected users increases with each round, thereby illustrating the correctness of Theorem 10. It is worth noting that the last round typically has the highest number of users in each figure. This is because the scheme is locally optimal, and after the first $(r-1)$ rounds, it would be desirable to allocate all the remaining data to the final round. This ensures that the maximum amount of information is obtained from the dataset. It is noteworthy that when $R = 20$, the number of users in the final three rounds remains almost constant. This is because the decrease in error during these rounds is insignificant for **Linear**, leading to an inconsequential effect on the number of users. Nevertheless, this scenario still adheres to a progressively increasing n_r, as demonstrated in Theorem 10.

5.4.3.6 Robust analysis

Fig. 6.15 shows the changes in the estimated results as the number of rounds increases. The overall trend of the frequency estimation after the first round (represented by the blue line) is already close to the ground truth (indicated by the dashed line). This is because in the initialization process, each user collected data $\frac{n_0}{d}$ times, providing an initial frequency estimation that is relatively accurate. However, there are many outliers in the blue line, representing cases where the frequency estimation is highly inaccurate after the initialization round.

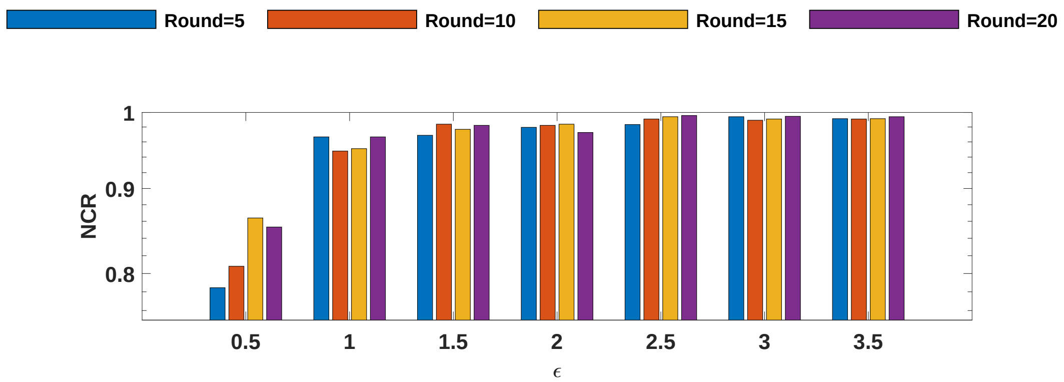
As the number of rounds increases, the frequency estimation is gradually corrected and approaches the true results. This demonstrates that the mechanism can effectively rectify the frequency errors introduced during the initialization process. This is because each sample has a probability of being sampled, and the difference in these probabilities is not too large.

5.5 Summary

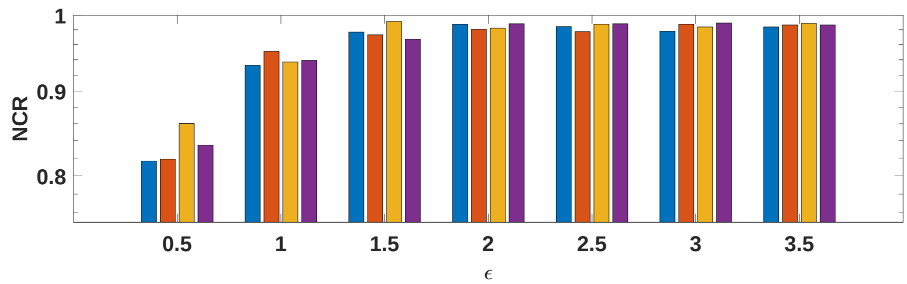
In this chapter, the focus is on top-k estimation for set-valued data under LDP. First, a comprehensive overview of existing LDP techniques and an evaluation of their suitability for set-valued data are provided. Then, a new perspective is offered and an unbiased and adaptive study of top-k estimation under LDP is presented.

Specifically, two adaptive sampling methods are deeply investigated: ARBS for identifying top-k items and ARBSF for both top-k item discovery and frequency estimation on these items. Additionally, an optimization to reduce computational complexity while maintaining low communication overhead is proposed. The theoretical and experimental results demonstrate the effectiveness of the proposed methods.

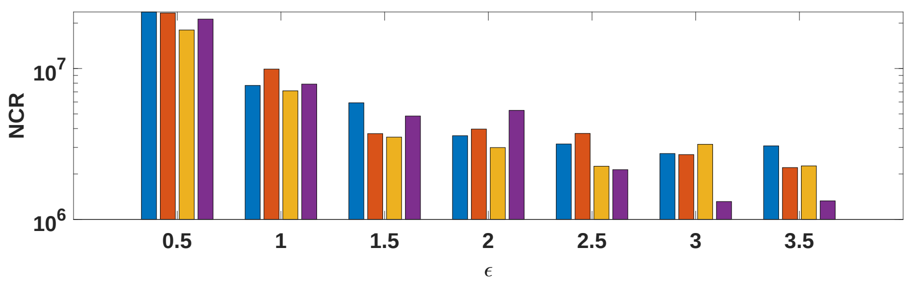
For future work, the plan is to explore multi-dimensional data for identifying and estimating top-k items and design suitable sampling strategies. Additionally, the development of a dynamic recommendation algorithm for streaming user data, which extends from this work by adaptively updating top-k items and continuously revising the sampling strategy in real time, is also planned.



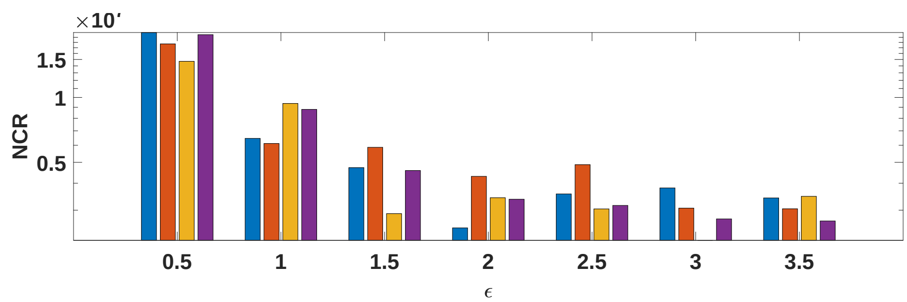
(a) **Linear** 45°, **ARBS**.



(b) **Linear** 45°, **ARBSF**.

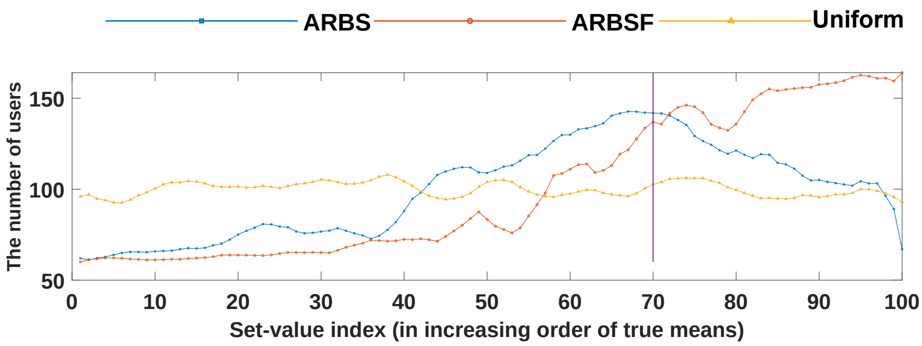


(c) **Beta(2,5)**, **ARBS**.

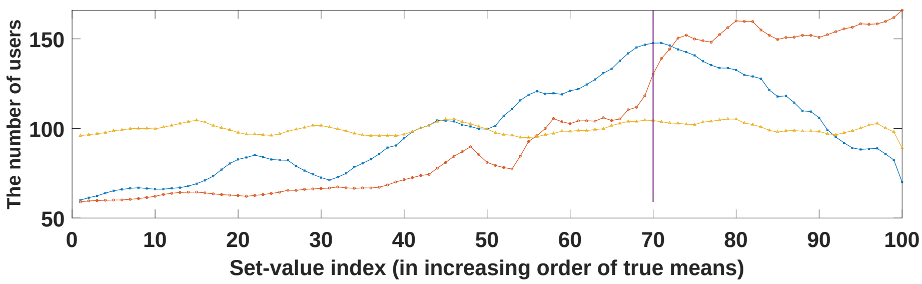


(d) **Beta(2,5)**, **ARBSF**.

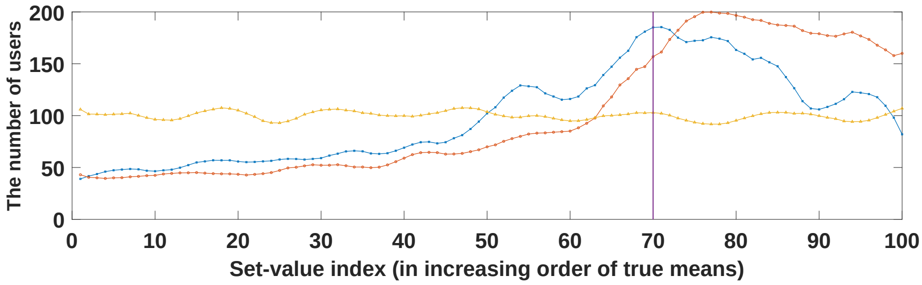
Figure 5.8: The results of NCR & MSE w.r.t. ϵ with different round.



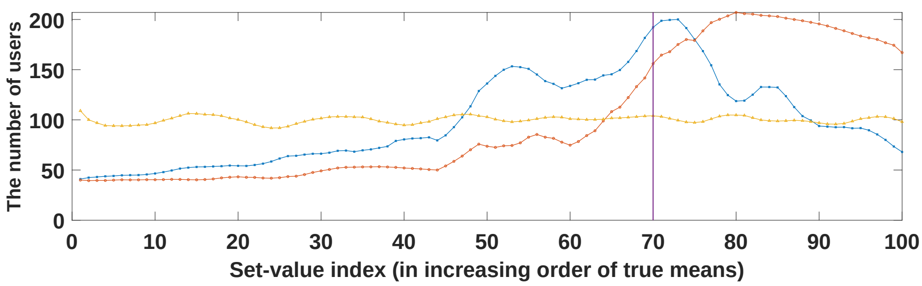
(a) **Linear** 45°, $\epsilon = 1$, **R=10**.



(b) **Linear** 45°, $\epsilon = 1$, **R=20**.

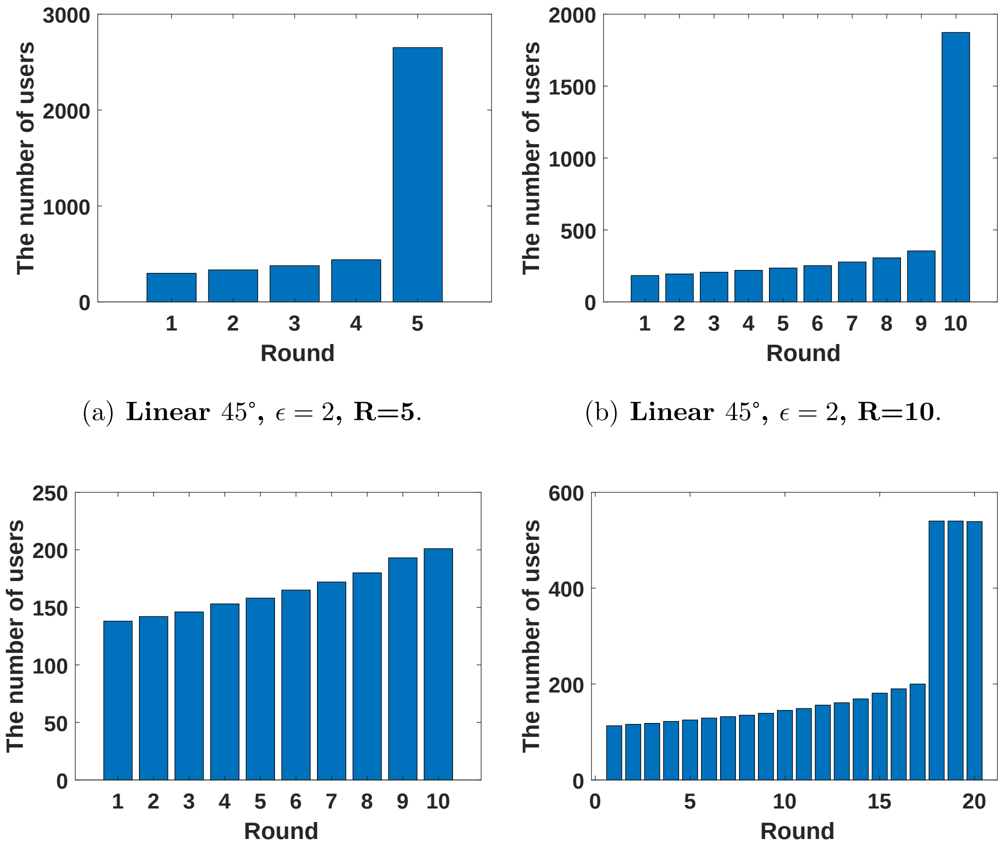


(c) **Linear** 45°, $\epsilon = 2$, **R=10**.



(d) **Linear** 45°, $\epsilon = 2$, **R=20**.

Figure 5.9: The number of users allocated on each item.



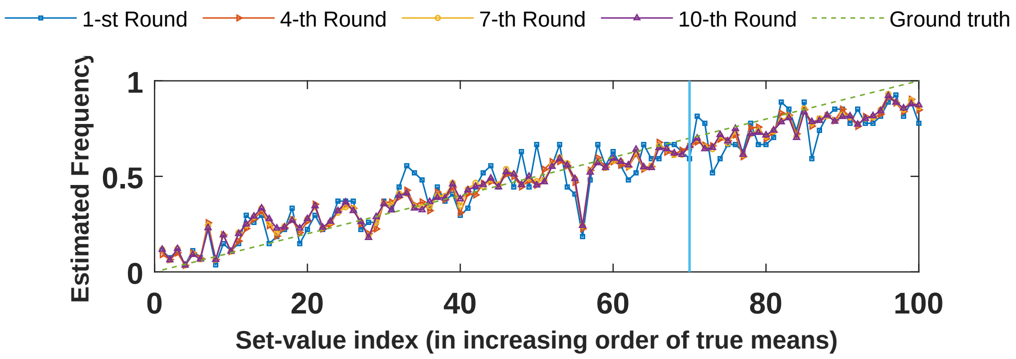
(a) **Linear** 45°, $\epsilon = 2$, **R=5**.

(b) **Linear** 45°, $\epsilon = 2$, **R=10**.

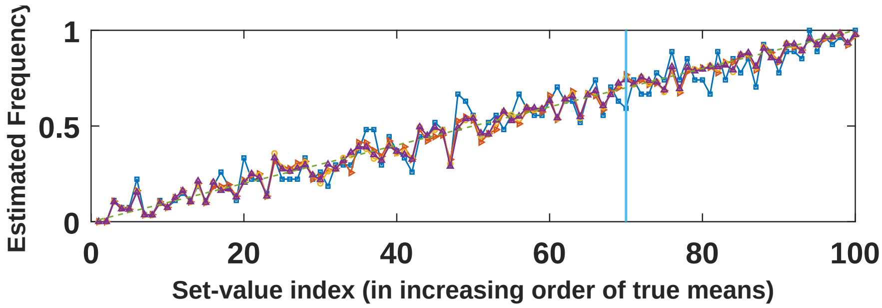
(c) **Linear** 45°, $\epsilon = 2$, **R=15**.

(d) **Linear** 45°, $\epsilon = 2$, **R=20**.

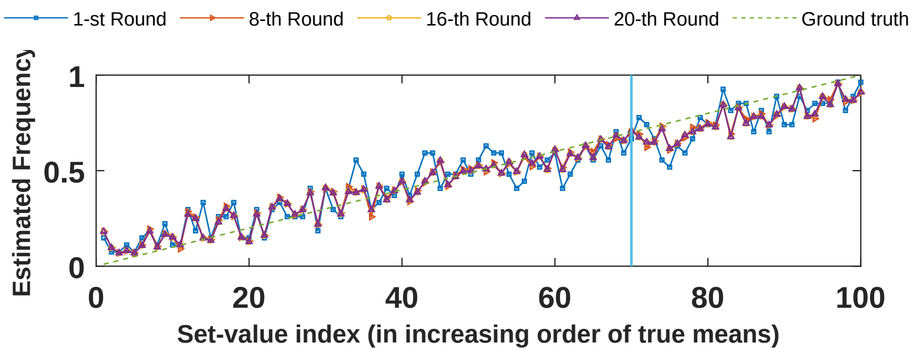
Figure 5.10: The number of users allocated in each round.



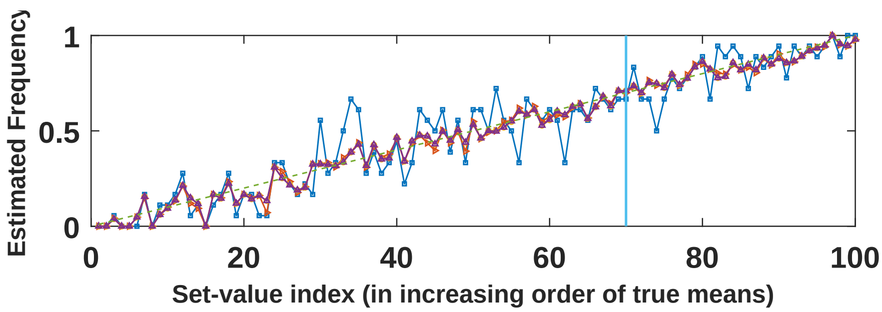
(a) **Linear** 45°, $\epsilon = 2, R = 10$.



(b) **Linear** 45°, $\epsilon = 3, R = 10$.



(c) **Linear** 45°, $\epsilon = 2, R = 20$.



(d) **Linear** 45°, $\epsilon = 3, R = 20$.

Figure 5.11: The results of NCR & MSE w.r.t. ϵ with different round.

Chapter 6

Distribution Estimation under LDP against Arbitrarily Distributed Attacks

In today's voting or product rating scenarios, LDP is commonly employed to protect user privacy [99, 118, 123]. However, the inherent data perturbation feature of LDP presents any user with the opportunity to deny or disprove poison values. This introduces a novel challenge in detecting Byzantine users through traditional methods [77, 122]. A recent work shows how poor the performance of an LDP protocol can be under a malicious model assumption [30]. Although a few works have proposed some countermeasures to address these Byzantine attacks in LDP [24, 120], they all require prior knowledge of either the attacking pattern or the poison value distribution, which is impractical as they can be easily evaded by the attackers.

This chapter studies distribution estimation under LDP model against a **general malicious threat model where attackers are opportunistic and colluding**. "Opportunistic" means that such attackers, whose objective is to increase the frequency or proportion of certain values in the distribution, can manipulate their poison values in their best interests. “Colluding”, also known as Sybil attacks, means the attackers can share their strategy and orchestrate their poison values. This is practical as these attackers can arise from a single Botnet launched by a single attacker. This threat model is more generic than any existing threat model in that it does not limit the attacking strategy, nor does it assume the collector know about the attacking strategy or probabilistic distribution of poison values.

Our prior study [38] introduces an innovative strategy in which an adversary is capable of executing arbitrary attacks within a predefined threshold range. This method is distinct from conventional methods such as trimming, as it does not attempt to differentiate a poison value from regular ones. Rather, it estimates and mitigates the aggregate impact of poison values. This unique strategy eliminates the need to establish a fixed trimming threshold. The fundamental technique adopted is the Expectation Maximization Filter (EMF) algorithm, based on which sophisticated techniques, collectively called EMF for mean estimation (ME-EMF for short), were developed to estimates Byzantine features from collected data. These features help probe and filter the overall impact of poison values on collected data. However, ME-EMF is limited to mean estimation only for the following reasons:

- Due to the nature of mean estimation, Byzantine users incline to add poison values on one side of the mean. However, this is no longer applicable to distribution estimation, where Byzantine users can add noise across the entire domain.
- ME-EMF pre-determines an overestimated mean to identify the range of the poison values’ location. As a result, data from normal users may be misclassified as Byzantine ones, leading to a poor distribution estimation.
- ME-EMF can only identify poison values that are biased to one side. When some poison values are injected into the other side of the mean, false positives occur and result in inaccurate distribution estimation.

To investigate the problem of LDP-based distribution estimation in the presence of Byzantine users, this chapter first introduces the General Byzantine Attack (GBA) as our threat model. In order to mitigate the impact of poison values under this attack, and to accurately estimate the data distribution for normal users, this chapter proposes a building block, namely, the Segmented Expectation-Maximization Filter (SEMF). It leverages the EMF's ability to extract information about poison values and roughly determine their locations, by partitioning the domain into several segments and detecting poison values in these individual segments. To determine the optimal number of segments, this chapter also devises an adaptive method that progressively increases the number of segments to achieve the most precise localization of poison values.

Based on SEMF, this chapter then presents the Distribution Estimation-EMF (DE-EMF), specifically engineered to estimate the distribution of poison values by simultaneously determining the distribution and proportion of Byzantine users. To optimize the performance of DE-EMF, this chapter introduces two post-processing methods: DE-EMF* and DE-REMF*. The former aims to further mitigate the effects of poison values, while the latter takes advantage of the estimated number of Byzantine users. Interestingly, all these techniques need a small privacy budget to accurately estimate the Byzantine users' proportion, and meanwhile need a large privacy budget to accurately estimate the distribution. To get the better of both worlds, this chapter proposes the Differential Aggregation Protocol (DAP) , which divides users into groups, each with a different privacy budget ϵ. The collector estimates the distribution for each group and then combines these estimates by minimizing the Mean Squared Error (MSE).

To summarize, our main contributions in this chapter are as follows:

- This chapter introduces a general threat model for Byzantine attacks in LDP, the first of its kind that can adapt to arbitrary attacking patterns. Under this threat model, this chapter proposes SEMF, a dynamic method that can roughly identify the location of poison values.

- This chapter designs DE-EMF, along with two post-processing methods, DE-EMF* and DE-REMF*, to accurately estimate the distribution in the presence of colluding malicious attackers.
- This chapter devises a multi-group differential aggregation protocol. This protocol implements a group-wise distribution aggregation scheme, assigns varying privacy budgets and weights to each group, and strives to optimize distribution estimation while minimizing the MSE.

The rest of this chapter is organized as follows. Chapter 6.1 formally defines our threat model and describes the framework. Chapter 6.2 introduces SEMF to estimate the approximate range of poison values. Chapter 6.3 proposes three methods to estimate the distribution for normal users. This chapter then presents the Differential Aggregation Protocol, a more secure and effective protocol for distribution estimation, in Chapter 6.4. The experimental results are shown in Chapter 6.5, and Chapter 6.6 concludes this chapter.

6.1 Problem Definition and Framework Overview

An essential assumption of most existing LDP works is that users will report their values honestly, which is impractical in real-world applications. Some recent studies show that LDP protocols are vulnerable to Byzantine attacks [24,30] and the situation becomes even worse when the perturbation is more substantial, i.e., with a smaller privacy budget ϵ.[1] This chapter first presents our threat model, based on which this chapter then introduces a framework for data distribution estimation in the context of LDP.

[1] ϵ is usually no more than 5.0 in existing LDP schemes, and no more than 3.0 in these attacks.

6.1.1 Threat Model

This chapter assumes an **unknown** number[2] of **colluding** Byzantine users know the LDP perturbation mechanism and the privacy budget ϵ, so they can send arbitrary values in the perturbation output domain $[D_L, D_R]$ to the data collector to undermine the data distribution. This chapter formalizes this attack as the threat model below.

Definition 8. ***General Byzantine Attack (GBA).*** *Given a normalized perturbation value domain $[D_L, D_R]$ and m colluding Byzantine users U_B with original values $V_B = \{v_1, ..., v_m\}$, a general Byzantine attack from U_B, denoted by $GBA(U_B)$, reports arbitrary poison values $V_B' = GBA(V_B, D_L, D_R)$ to the collector, where $V_B' \in [D_L, D_R]^m$.*

GBA is a general model which also covers input manipulation attacks[3], where Byzantine users further perturb poison values using the same LDP protocol as normal users, since the perturbed poison values still fall within the range $[D_L, D_R]$.

6.1.2 System Model and Framework

Fig. 6.1 shows the system model and our aggregation framework. There are N users, among which n are normal users (in green) and m are Byzantine users (in red). Normal users perturb and normalize their values v_i into $v_i' \in [D_L, D_R]$ according to an LDP perturbation mechanism, and report them to the data collector for aggregation. Byzantine users conduct GBA attacks by choosing and reporting poison values to the data collector (step ①). The goal of the data collector is to estimate the distribution of **normal users**. In contrast to existing detection-based methods [67, 73, 96], in our framework the data collector first probes collected values to approximate the

[2] To achieve Byzantine fault tolerance (BFT), the proportion of Byzantine users is bounded by 1/2.

[3] In our previous work [38], we design a strategy to combat this kind of attack, and obtained promising experimental results.

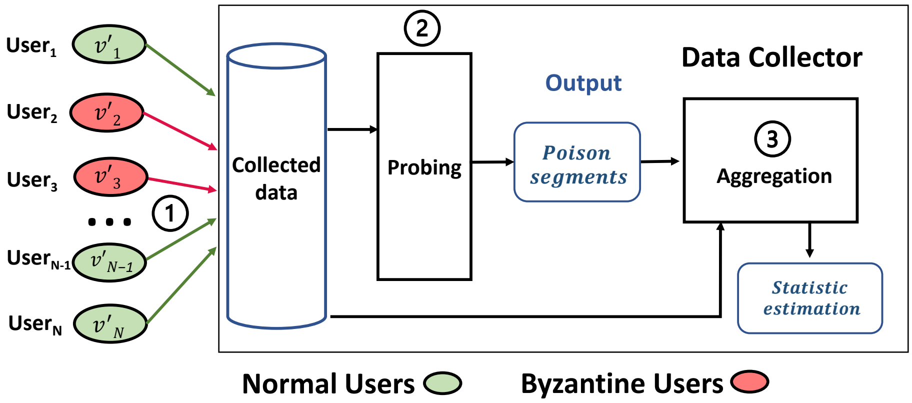


Figure 6.1: System model and aggregation framework

poison segments where the poison values V'_B are located (step ②), and then accurately estimates the aggregated distribution by mitigating the influence of V'_B. The main notations used in this chapter are listed in Table 6.1.

The rest of this chapter will illustrate how this framework can be used for distribution aggregation under LDP privacy model where Byzantine users exist. In Chapter 6.2, this chapter presents a method for probing poison segments and, based on this, estimating the distribution in Chapter 6.3. Then a security-enhanced protocol will be further proposed in Chapter 6.4.

6.2 Probing Poison Segments

This chapter proposes a method to approximate the location of poison values, which are referred to as "poison segments" throughout the chapter. In particular, this chapter first introduces Expectation-Maximization Filter (EMF) assuming the segment information is provided, and then propose Segmented Expectation-Maximization Filter (SEMF) to precisely identify the required poison segments. In what follows this chapter illustrates our algorithms using SW mechanism as the perturbation mechanism, and thus perturbation output domain $[D_L, D_R]$ becomes $[-b, 1+b]$.

Table 6.1: Notations

Symbol	**Description**
U	the set of users
U_B	the set of Byzantine users
N	the number of users in U, $\|\|U\|\| = N$
u_i	the i-th user in U
v_i	the original value of u_i
$v_i^{'}$	the perturbed value of u_i
V	the original values $V = \{v_1, ..., v_i, ..., v_N\}$
V_B	the original values of Byzantine users
V_B'	the collected values from Byzantine users
n	the number of normal users
m	the number of Byzantine users
γ	the proportion of Byzantine users $\gamma = \frac{m}{N}$
$\hat{m}$	the estimated number of Byzantine users
$\hat{\gamma}$	the estimated proportion of Byzantine users $\hat{\gamma} = \frac{\hat{m}}{N}$
u	the number of buckets for normal users
w	the number of buckets of perturbed values
G_t	the t-th group G_t
ϵ_t	the privacy budget in G_t
D_t	the estimated distribution for normal users in G_t
$\tilde{D}$	the aggregated distribution for normal users

6.2.1 Expectation-Maximization Filter

Recall that normal users perturb their values by SW Mechanism (i.e., Equ. 3.6), whereas Byzantine users report poison values directly. Such a difference inspires us to probe some features of poison values and then reconstruct the original distribution. Hereafter, this chapter presents EMF to estimate such information by Maximum Likelihood Estimation (MLE) [71]. The chapter uses a $w \times (u + w)$ transform matrix M to characterize the randomization process.

Design of M. The chapter discretizes the original value domain $[0, 1]$ into u buckets, and discretizes the perturbed value domain $[-b, 1 + b]$ into w buckets, respectively. For normal users, the frequency histogram of their original values in buckets $B_x = B_{x_1}, ..., B_{x_u}$ is denoted as $\mathbf{x} = x_1, ..., x_u$, and the frequency histogram of their perturbed values in buckets $B_b = B_{b_1}, ..., B_{b_w}$ is denoted as $b_1, ..., b_w$. Since Byzantine users can inject arbitrary values from $[-b, 1+b]$, the frequency histogram of the poison values in buckets $B_y = B_{y_1}, ..., B_{y_w}$ is denoted as $\mathbf{y} = y_1, ..., y_w$.

Let $V = \{v_1, v_2, ..., v_N\}$ denote the users' original values, and $V' = \{v'_1, v'_2, ..., v'_N\}$ denote the collected ones. The left-hand side of $\mathbf{M}$ is a $w \times u$ matrix for normal users, where each element is of the form $M_{i,j} = \Pr[v' \in B_{b_i} | v \in B_{x_j}] (i \in \{1, ..., w\}, j \in \{1, ..., u\})$, indicating the transition probability from an input B_{x_j} to an output B_{b_i}. More specifically, given the original value v falling in B_{x_j}, let p (as defined in Equ. 3.6) denote the probability that the perturbed value v' falls in a bucket B_{b_i} within the range $[v - b, v + b]$, with q denoting the probability otherwise. For example, as illustrated in Fig. 6.2, given an original value in bucket B_{x_3}, $M_{1,3} = q$ signifies that the conditional probability of the output falling into bucket B'_{b_1} is q.

The right-hand side of the transform matrix M is a $w \times w$ matrix for poison values, where $M_{i,j+u} = \Pr[v' \in B_{b_i} | v \in B_{y_j}]$ for $i, j \in 1, ..., w$. The buckets that contain true poison values are referred to as "poison buckets", and the corresponding set is labeled as B^p. Additionally, a broader set B^{p+} is tested as the potential buckets where poison

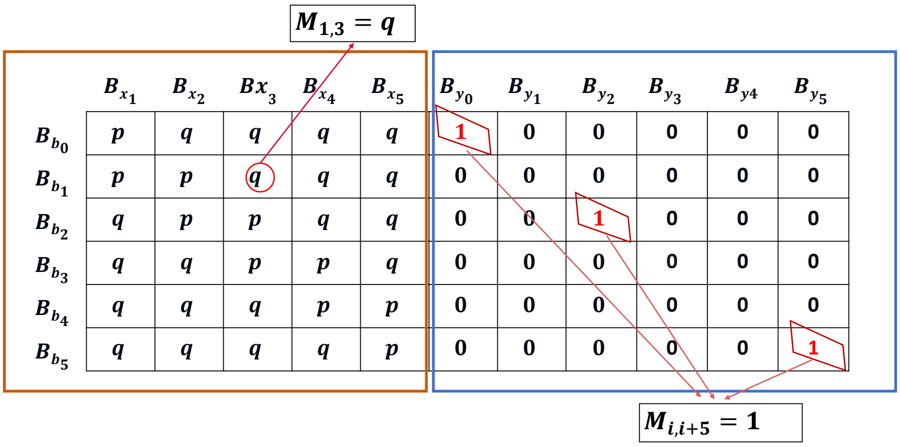


Figure 6.2: Transform matrix M

values might locate, satisfying $B^p \subset B^{p+} \subset B_y$. Since Byzantine users send poison values directly to the data collector, for each poison bucket $i \in B^p$, $M_{b_i y_i} = 1$, and the remaining elements in the right-hand matrix are set to 0.

Procedure of EMF. With the transformation matrix M, the next step for the EMF (Expectation-Maximization with Frequency estimation) is to reconstruct the frequency histogram F, involving the frequency counts of both normal users x and poison values y in the original value domain. That is, $F = \mathbf{x}, \mathbf{y} = x_1, ..., x_u, y_1, ..., y_w$. To adopt maximum likelihood estimation, the log-likelihood function of F is obtained as follows:

$$
\begin{gathered}
l(F) = ln\Pr[V^{'}|F] = ln\prod_{i=1}^{N}\Pr[v_i'|F] = \\
\sum_{i=1}^{N} ln(\sum_{k=1}^{u} x_k \Pr[v_i'|v_i \in B_{x_k}] + \sum_{B_{y_j}\in B^{p+}} y_j \Pr[v_i'|v_i \in B_{y_j}])
\end{gathered}
\tag{6.1}
$$

where $\Pr[v_i'|v_i \in B_{x_k}]$ and $\Pr[v_i'|v_i \in B_{y_j}]$ are constants, thus $l(F)$ is a concave function and EM algorithm can converge to the maximum likelihood estimator [20].

The EMF is designed to reconstruct the frequency histogram, where normal values are denoted as $\hat{x}$ and poison values as $\hat{y}$, and the specific steps are outlined in Algorithm 6. These steps are derived based on the methodology detailed in the literature [71]. Note that b_i in E-step denotes the count of perturbed values from V' in bucket B_{b_i}. First,

Algorithm 6: Expectation-Maximization Filter

Input: Collected values histogram $\mathbf{b} = \{b_1, b_2, ..., b_w\}$

Poison buckets B^{p+}

Output: Frequency histograms $\hat{x}$ and $\hat{y}$

1: Initialization: $\hat{x_k} = \hat{y_j} = \frac{1}{u+|B^{p+}|}$ **while** *not converge* **do**

2: **E-step:** $\forall B_{x_k}$ when $B_{y_k} \notin B^{p+}$

3: $P_{x_k} = \hat{x_k} \sum_{i=1}^{w} b_i \frac{M_{i,k}}{\sum_{t=1}^{u} M_{i,t}\hat{x_t}}$

4: $\forall B_{x_k}$ when $B_{y_k} \in B^{p+}$

5: $P_{x_k} = \hat{x_k} \sum_{i=1}^{w} b_i \frac{M_{i,k}}{\sum_{t=1}^{u} M_{i,t}\hat{x_t} + M_{i,i+u}\hat{y_i}}$

6: $\forall\ B_{y_j} \in B^{p+}$

7: $P_{y_j} = \hat{y_j} \sum_{i=1}^{w} b_i \frac{M_{i,j+u}}{\sum_{t=1}^{u} M_{i,t}\hat{x_t} + M_{i,i+u}\hat{y_i}}$

8: **M-step:** $\forall\ B_{x_k} \in B_x$

9: $\hat{x_k} = \frac{P_{x_k}}{\sum_{i=1}^{u} P_{x_i} + \sum_{j=w/2+1}^{w} P_{y_j}}$

10: $\forall\ B_{y_j} \in B^{p+}$

11: $\hat{y_j} = \frac{P_{y_j}}{\sum_{k=1}^{u} P_{x_k} + \sum_{i=w/2+1}^{w} P_{y_i}}$

12: **return** $\hat{x}$, $\hat{y}$

the algorithm assigns some non-zero initial values to $\hat{x}$ and $\hat{y}$, subject to $\hat{x} + \hat{y} = 1$ (line 1). Next, it executes the EM algorithm, alternating the E-step and the M-step. The E-step evaluates the log-likelihood expectation by the observed counts b_i and the current $\hat{x}$ and $\hat{y}$ (lines 3-8), and the M-step updates $\hat{x}$ and $\hat{y}$ that maximize the expected likelihood (lines 9-12) as inputs for the next round E-step. Finally, the algorithm returns the estimated frequency histogram $\hat{x}$ and $\hat{y}$ when the convergence condition is met (line 13).

Regarding the selection of B^{p+}, which includes all poison buckets, the following theorem shows B^{p+} cannot be equal to B_y. Furthermore, the closer B^{p+} is to B^p, the more accurate the result will be.

Theorem 11. *Given $B^{p+} = B_y$, $\exists\varepsilon : \mathbb{R}^+ \to \mathbb{R}^+$ such that when $|B^{p+}| - |B^p| \to \delta$, the deviation of normal users' distribution is bounded by $\varepsilon(\delta)$.*

Proof. Recall that B^p exclusively includes all poison buckets, and the set we are probing is denoted as B^{p+}. The chapter starts from the state where $B^{p+} = B_y$, and reconstruct the frequency histogram for poison values in $[-b, 1+b]$. The likelihood estimator in Equ. 6.1 becomes:

$$\begin{aligned} l(F) &= \sum_{i=1}^{N} ln(\sum_{k=1}^{u} \hat{x_k} Pr[v_i' | v_i \in B_{x_k}] + \sum_{j=1}^{w} \hat{y_j} Pr[v_i' | v_i \in B_{y_j}]) \\ &= \sum_{t=1}^{w} b_t ln(\sum_{k=1}^{u} \hat{x_k} M_{t,k} + \sum_{j=1}^{w} \hat{y_j} M_{t,j}). \end{aligned}$$

Note that $\sum_{k=1}^{u} \hat{x_k} + \sum_{j=1}^{w} \hat{y_j} = 1$, the Lagrangian multiplier method is employed to derive the extreme. The Lagrangian function can be written as:

$$L(F) = l(F) + \lambda(\sum_{k=1}^{u} \hat{x_k} + \sum_{j=1}^{w} \hat{y_j} - 1).$$

Let all first-order partial derivatives of L w.r.t. $\hat{x_k}$ and $\hat{y_j}$ equal zero

$$\begin{aligned} \frac{\partial L(F)}{\partial \hat{x_k}} &= \sum_{t=1}^{w} \frac{b_t M_{t,k}}{\sum_{k=1}^{u} \hat{x_k} M_{t,k} + \hat{y_t}} + \lambda = 0,\ k \in \{1, ..., u\} \\ \frac{\partial L(F)}{\partial \hat{y_j}} &= \sum_{t=1}^{w} \frac{b_t M_{t,j+u}}{\sum_{k=1}^{u} \hat{x_k} M_{t,k} + \hat{y_t}} + \lambda = 0,\ j \in \{1, ..., w\} \end{aligned}$$

Threrefore:

$$\hat{x_k} = 0,\ k \in \{1, ..., u\},\quad \hat{y_j} = \frac{b_j}{N},\ j \in \{1, ..., w\},\ \lambda = -N.$$

This result shows all collected values converge to poison values when $B^{p+} = B_y$, and therefore:

$$(\sum_{k=1}^{u} \hat{x_k} + \sum_{B_{y_j} \in B^p} \hat{y_j})\Big|_{y_i \neq 0, i \in \{1, ..., w\}} = \sum_{B_{y_j} \in B^p} \frac{b_j}{N}.$$

Let $B^c = B_y - B^p$ and $B^c = \{B_{y_{\theta,1}}, ..., B_{y_{\theta,|B^c|}}\}$. When we remove the bucket $B_{y_{\theta,1}}$ (by setting $y_{\theta,1} = 0$) in B^c and carry out EMF, the collected values in B'_{b_w} can only converge to $B_{x_k} (k \in \{1, ..., u\})$, but not B_y. Hence, every $\hat{x_k}$ will increase. Therefore, removing $B_{y_{\theta,1}}$ leads to the increase of all $\hat{x_k}$, which in turn results in the decrease of all $\hat{y}_j$. However, since the decrease of B^p is a part of increment of $\hat{x}$, we can figure out $(\sum_{k=1}^{u}\hat{x}_k + \sum_{B_{y_j}\in B^p}\hat{y}_j)\Big|_{y_i\neq 0, i\in\{1,\ldots,w\}} \leq (\sum_{k=1}^{u}\hat{x}_k + \sum_{B_{y_j}\in B^p}\hat{y}_j)\Big|_{y_{\theta,1}=0}$. Removing all buckets one by one in B^C similarly, we have: $(\sum_{k=1}^{u}\hat{x}_k + \sum_{B_{y_j}\in B^p}\hat{y}_j)\Big|_{y_i\neq 0, i\in\{1,\ldots,w\}}$ $\leq (\sum_{k=1}^{u}\hat{x}_k + \sum_{j=1}^{t}\hat{y}_j)\Big|_{y_{\theta,1}=0} \leq \ldots \leq (\sum_{k=1}^{u}\hat{x}_k + \sum_{B_{y_j}\in B^p}\hat{y}_j)\Big|_{y_{\theta,1}=0,\ldots,y_{\theta,|B^c|}=0}$. When the number of buckets removing in B^c increases, the corresponding interference of $B_y - B^c$ decreases. Therefore, the collected values more accurately converge to the buckets that they should belong to, and thus achieve a better convergence result.

After removing all buckets in B^c, all collected values will convergence to normal values and poison values in B^p and we can infer that $(\sum_{k=1}^{u}\hat{x}_k + \sum_{B_{y_j}\in B^p}\hat{y}_j)\Big|_{y_{\theta,1}=0,\ldots,y_{\theta,|B^c|}=0}$, which is the optimal case where none of the collected values will converge to buckets in B^c. $\square$

6.2.2 Segmented Expectation-Maximization Filter

So far, this chapter has proposed EMF on the assumption that B^{p+} is known, but in practice the buckets selected by Byzantine users are usually unknown. To address this challenge, this chapter introduces SEMF, which combines domain partitioning with EMF.

SEMF begins by partitioning the output domain into several segments. Each segment is then individually probed for poison values, assuming only the currently probed segment might contain poison values, while all other segments are considered poison-free. If a segment does not contain poison values, all poison values should converge to $\hat{x}$. To align the perturbation results of $\hat{x}$ closely with the collected data V', the values corresponding to this segment are also mapped to $\hat{x}$. As a result, if a probed segment does not contain poison values, the EMF will detect a minimal number of poison values.

Let's use an example to show how SEMF identifies the approximate range of poison values. If the domain is partitioned into four segments and the probing result is [0,0,0,1], this suggests that only the fourth segment is a poison segment containing poison values. Accordingly, B^{p+} will include all the buckets that are located within the poison segments.

A challenge arises from determining an appropriate number of segments, denoted by n_s. A large n_s is computationally hard, while a small n_s make it difficult to find poison value-free segments. To address this challenge, this chapter proposes an adaptive method that starts at $n_s = 2$ and increases it until a B^{p+} (rather than B_y) is found. Let $S(n_s, i)$ denote the set of all buckets in the i-th segment when the domain is partitioned into n_s segments. The details of this process are presented in Algorithm 7.

Algorithm 7: SEMF Algorithm

Input: Buckets set for collected data B_y

Number of segments n_s, Collected histogram **b**

Output: Poison set B^{p+}

1: Initialize: $n_s = 2$, flag=0, $B^{p+} = B_y$ **while** *flag==0 & $n_s < \tau$* **do**

2:

For$i = 1$ **to** n_s

3: $\hat{x}, \hat{y} = \text{EMF}(\mathbf{b}, S(n_s, i))$ **if** *sum*$(\hat{y}) < T$ **then**

4:

$B^{p+} = B^{p+} - S(n_s, i)$; flag=1 **EndIf**

5: $n_s = 2 * n_s$ **EndFor EndWhile if** *flag==1* **then**

6:

Find the segment k with the smallest sum$(\hat{y})$

7: $B^{p+} = B^{p+} - S(n_s, k)$

8: **EndIf**

9: **Return** B^{p+}

The algorithm first initializes $n_s = 2$ and $B^{p+} = B_y$ (line 1). It then partitions the w buckets into n_s segments, applying EMF to each segment (line 2). Specifically, when the algorithm checks whether the i-th segment is a poison segment, it applies the EMF and adjusts $\mathbf{M}$ based on that segment (line 4). If the number of poison values in a segment falls below a threshold T, all buckets within this segment are removed from B^{p+}. Otherwise, n_s is doubled for the next iteration, and the process is repeated (lines 5-7). If an available B^{p+} cannot be found within the maximum allowable value τ, the segment with the smallest sum of estimated distribution is removed from the poison set (lines 8-10). The final poison set B^{p+} is then returned (line 11). The procedure of SEMF probing the range of poison values is similar to a binary search with probing, with a complexity of O(log n).

6.3 Distribution Estimation under GBA

The previous discussion introduces EMF and a more practical version SEMF, for identifying the poison segments. This chapter leverages the information derived from poison segments and employ the EMF-based approach to estimate the distribution for normal users. This primarily includes DE-EMF and two post-processing methods, namely DE-EMF* and DE-REMF*, to enhance the performance of the distribution estimation.

6.3.1 DE-EMF

Distribution Estimation. DE-EMF is an intuitive solution for distribution estimation, by first adopting SEMF for identifying poison segments and then applying EMF for estimating the data distribution. The process is outlined in Algorithm 8. The data collector uses SEMF to find a B^{p+} (line 1) first, then utilizes the information from B^{p+} as the probing buckets set and estimate the distribution of normal users with EMF (line 2). The result $\hat{x}$ represents the distribution for normal users and $\hat{y}$ represents that for Byzantine users.

Algorithm 8: DE-EMF Algorithm

Input: Collected histogram $\mathbf{b} = \{b_1, b_2, ..., b_w\}$

Output: Frequency histograms $\hat{x}$ and $\hat{y}$

1: Run SEMF and obtain B^{p+}

2: $\hat{x}, \hat{y} = \text{EMF}(\mathbf{b}, B^{p+})$

3: Return $\hat{x}, \hat{y}$

The proportion of Byzantine users $\hat{\gamma}$ can also be derived from $\hat{y}$, the estimated frequency histogram of poison values by DE-EMF:

$$\hat{\gamma} = \sum_{B_{b_j} \in B^{p+}} \hat{y_j} = \frac{\hat{m}}{N} \approx \frac{m}{N} = \gamma, \tag{6.2}$$

where $\hat{m}$ denotes the estimated number of Byzantine users, and γ denotes the true proportion of Byzantine users.

When $\epsilon \to 0$, the estimated frequency histogram $\hat{x}$ of normal users converges to a uniform distribution, whereas that of Byzantine users $\hat{y}$ converges to the true distribution of poison values. Thus, Equ. 6.2 can be proved by the following theorem:

Theorem 12. *Let $a = \{a_1, ..., a_{|B^{p+}|}\}$ denote the count of poison values in corresponding buckets from B^{p+}. When $\epsilon \to 0$, the convergence results are $\hat{x_k} = \frac{n}{Nd}$(for $B_{x_k} \in B_x$) and $\hat{y_j} = \frac{a_j}{N}$(for $B_{y_j} \in B^{p+}$).*

Proof. When $\epsilon \to 0$, all inputs from normal users are equally perturbed into w buckets with probability $\frac{1}{w}$, which leads to a uniform distribution. Let $a = \{a_1, ..., a_{|B^{p+}|}\}$ denote the count of poison values in corresponding buckets from B^{p+}, and we have $b_t \to \frac{n}{w}, B_{x_t} \notin B^{p+}$ and $b_t \to \frac{n}{w} + a_t, B_{x_t} \in B^{p+}$.

Note that $\sum_{k=1}^{u} \hat{x_k} + \sum_{B_{x_j} \in B^{p+}} \hat{y_j} = 1$, the Lagrangian function of Equ. 6.1 can be written as:

$$L(F) = l(F) + \omega(\sum_{k=1}^{u} \hat{x_k} + \sum_{B_{x_j} \in B^{p+}} \hat{y_j} - 1).$$

Let all first-order partial derivatives of L w.r.t. $\hat{x_k}$ and $\hat{y_j}$ equal zero

$$\frac{\partial L(F)}{\partial \hat{x_k}} = \sum_{t=1}^{w} b_t \frac{\frac{1}{w}}{\sum_{k=1}^{u} \hat{x_k}\frac{1}{w}} + \omega,\ B_{x_k}, By_k \notin B^{p+}$$
$$\frac{\partial L(F)}{\partial \hat{x_k}} = \sum_{t=1}^{w} b_t \frac{\frac{1}{w}}{\sum_{k=1}^{u} \hat{x_k}\frac{1}{w} + \hat{y_t}} + \omega,\ B_{x_k}, By_k \in B^{p+}$$
$$\frac{\partial L(F)}{\partial \hat{y_j}} = b_j \frac{1}{\sum_{k=1}^{u} \hat{x_k}\frac{1}{w} + \hat{y_j}} + \omega,\ B_{y_j} \in B^{p+}$$

we have:

$$\hat{x_k} \to \frac{n}{Nd},\quad \hat{y_j} \to \frac{a_j}{N}, B_{y_j} \in B^{p+},\ \omega \to -N. \tag{6.3}$$

According to the deduction results, the frequency histogram $\hat{x}$ of normal users converges to a uniform distribution, while that of Byzantine users $\hat{y}$ converges to the true distribution of poisoned values. □

6.3.2 Post-processing Methods

DE-EMF seeks to extract $\{\hat{x}, \hat{y}\}$, a unified distribution for all users, from collected values. Clearly, the convergence result of $\hat{y}$ influences that of $\hat{x}$. Furthermore, when epsilon is large, the number of Byzantine users can be overestimated or underestimated, leading to inaccurate distribution estimation. To overcome these challenges and improve the accuracy of the distribution estimation for normal users, the chapter develops two optimized methods DE-EMF* and DE-REMF*, by combining DE-EMF with a post-processing step.

During the DE-EMF process, estimating the poison value distribution negatively impacts that of the normal users. Therefore, this chapter proposes DE-EMF*, which incorporates a post-processing step that involves re-running EMF after removing poison values.

The specific steps, as outlined in Algorithm 9, involve removing the poison values (obtained from DE-EMF) from the results of DE-EMF (line 2), then setting B^{p+} to an empty set (line 3), effectively assuming the system no longer contains Byzantine users. The convergence result $\hat{y}$ will be a zero vector, and $\hat{x}$ represents the refined distribution of data for normal users (line 4).

Algorithm 9: DE-EMF* Algorithm

Input: Poison histogram $\hat{y} = \{y_1, y_2, ..., y_w\}$

Collected histogram $\mathbf{b} = \{b_1, b_2, ..., b_w\}$

Output: Estimated histogram for normal users $\hat{x}$

1: Obtain $\hat{x}, \hat{y}$ by running DE-EMF

2: Collected histogram for normal users $\mathbf{b} = \mathbf{b} - \hat{y}$

3: $B^{p+} = \varnothing$

4: $\hat{x}, \hat{y} = \text{EMF}(\mathbf{b}, B^{p+})$

5: **Return** $\hat{x}$

DE-REMF*. Once the proportion of Byzantine users $\hat{\gamma}$ probed from Equ. 6.2 is known[4], it can be utilized to improve the convergence process, by imposing $\sum \hat{y} = \hat{\gamma}$, $\sum \hat{x} = 1 - \hat{\gamma}$ as two additional restrictions in EMF. This idea leads to the optimized method DE-REMF*. Accordingly, the maximization problem in **M** steps of EMF becomes:

$$\begin{aligned} &\arg\max_{\hat{x},\hat{y}} \sum_{k=1}^{u} P_{x_k} \ln \hat{x_k} + \sum_{B_{y_j} \in B^{p+}} P_{y_j} \ln \hat{y_j}, \\ &\text{subject to } \sum_{k=1}^{u} \hat{x_k} = 1 - \hat{\gamma}, \sum_{B_{y_j} \in B^{p+}} \hat{y_j} = \hat{\gamma} \end{aligned} \tag{6.4}$$

Algorithm 10 shows the pseudo-code of DE-REMF*, where the M-step from EMF has been modified with the results of Theorem 13. In addition, this chapter runs Algorithm 9 once again at line 12 to further eliminate the influence of poison values on the distribution estimation.

DE-REMF* can improve DE-EMF/DE-REMF* in terms of the accuracy of converged

[4] A more precise estimate of $\hat{\gamma}$ can either be obtained from pre-existing knowledge or acquired by executing the DE-EMF algorithm with a small epsilon.

Algorithm 10: DE-REMF* Algorithm

Input: Transform matrix $\mathbf{M}$

Collected values V'

Output: Frequency histograms $\hat{x}$ and $\hat{y}$

Initialization: Obtain $\hat{x}, \hat{y}$ by running DE-EMF

$\hat{x}, \hat{y}$ = DE-EMF

$\hat{\gamma} = \sum \hat{y}$ **while** *not converge* **do**

E-step:

Same as the E-step in EMF in Algorithm 6

M-step:

$\forall\ B_{x_k} \in B_x$

$\hat{x_k} = (1-\hat{\gamma})\frac{P_{x_k}}{\sum_{i=1}^{u} P_{x_i}}$

$\forall B_{y_j} \in B^{p+}$

$\hat{y_j} = \hat{\gamma}\frac{P_{y_j}}{\sum_{B_{y_j} \in B^{p+}} P_{y_i}}$

EndWhile

Run Algorithm 9

return $\hat{x}, \hat{y}$

poison value histogram because the additional restrictions eliminate those infeasible poison values. To resolve Equ. 6.4, the following theorem is shown:

Theorem 13. *The maximum in Equ. 6.4 is reached when the output frequency histograms are*

$$\hat{x_k} = (1-\hat{\gamma})\frac{Px_k}{\sum_{i=1}^{u} Px_i},\ \hat{y_j} = \hat{\gamma}\frac{Py_j}{\sum_{B_{b_i} \in B^{p+}} Py_i}.$$

Proof. We apply Lagrangian multiplier method to derive maximal value of Euq. 6.4. Let

$$\mathcal{L} = \sum_{k=1}^{u} P_{x_k} ln\hat{x_k} + \sum_{B_{y_j \in B^{p+}}} P_{y_j} ln\hat{y_j}$$
$$+ \lambda_1(\sum_{k=1}^{u} \hat{x_k} - 1 + \hat{\gamma}) + \lambda_2(\sum_{B_{y_j} \in B^{p+}} \hat{y_j} - \hat{\gamma}),$$

where λ_1 and λ_2 are two constants and the first-order partial derivatives of $\mathcal{L}$ w.r.t. $\hat{x_k}$ and $\hat{y_j}$ are

$$\frac{\partial \mathcal{L}}{\partial \hat{x_k}} = \frac{P_{x_k}}{\hat{x_k}} + \lambda_1,\ \frac{\partial \mathcal{L}}{\partial \hat{y_j}} = \frac{P_{y_j}}{\hat{y_j}} + \lambda_2.$$

Let $\frac{\partial l}{\partial \hat{x_k}}$ and $\frac{\partial l}{\partial \hat{y_j}}$ be zero, and we have

$$P_{x_k} + \lambda_1 \hat{x_k} = 0,\ P_{y_j} + \lambda_2 \hat{y_j} = 0. \tag{6.5}$$

From the restrictions in Equ. 6.4, we can deduce that

$$\lambda_1 = \frac{\sum_{i=1}^{u} P_{x_i}}{\hat{\gamma} - 1},\ \lambda_2 = \frac{\sum_{B_{y_i} \in B^{p+}} P_{y_i}}{-\hat{\gamma}}.$$

Replacing them in Equ. 6.5, we can reach that

$$\hat{x_k} = (1 - \hat{\gamma}) \frac{Px_k}{\sum_{i=1}^{u} P_{x_i}},\ \hat{y_j} = \hat{\gamma} \frac{Py_j}{\sum_{B_{y_i} \in B^{p+}} P_{y_i}}.$$

□

6.4 Differential Aggregation Protocol

Utilizing DE-EMF, DE-EMF* and DE-REMF* with a smaller ϵ allows for a more precise estimation of the proportion of poison values, while a larger ϵ leads to a more accurate distribution. To leverage the benefits provided by different ϵ values, this chapter introduces a multi-group collection protocol, named Differential Aggregation Protocol (DAP), in this chapter. Our idea is to randomly assign users into h groups, each with its own privacy budget setting. The collector performs SEMF (i.e., Algorithm 7) in each group to probe poison segments. And then the collector estimates a distribution from each group, based on which an inter-group distribution is aggregated. There advantages are three-fold. First, Byzantine users cannot differentiate if their values are used for probing or estimation and thus take the above strategy. Second, there is no need to split privacy budgets for users,

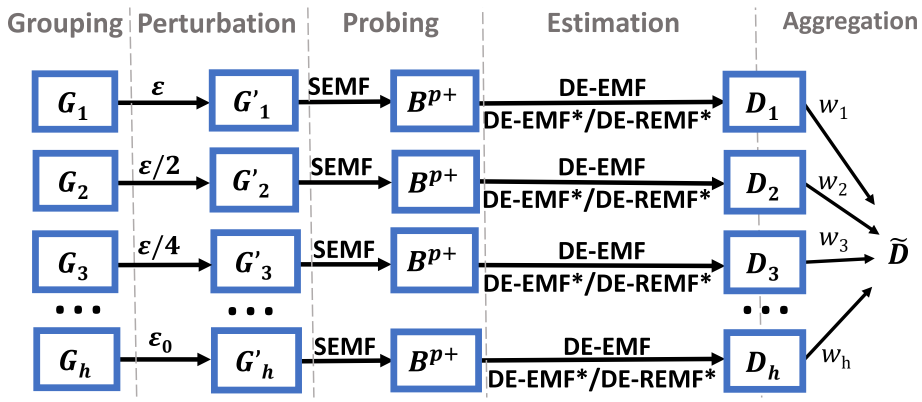


Figure 6.3: Differential Aggregation Protocol

which can improve the estimation accuracy. Third, this protocol can naturally handle users with different privacy budgets.

As illustrated in Fig. 6.3, the work flow of DAP has five stages:

1. **Grouping.** The data collector allocates users into groups and assigns each group with a dedicated privacy budget.
2. **Perturbation.** Users in each group perturb their values according to their assigned privacy budgets and send them to the data collector.
3. **Probing.** The data collector executes REMF* for each group to probe B^{p+}.
4. **Intra-group Estimation.** The data collector attains a distribution estimation from each group with DE-EMF/DE-EMF*/DE-REMF*.
5. **Inter-group Aggregation.** The data collector aggregates estimated distributions from all groups into one.

6.4.1 Grouping

First of all, the data collector determines a minimal acceptable privacy budget ϵ_0 to bound the perturbation noise of normal values according to the privacy budget ϵ for users. Then data collector creates $h = \lceil log_2(\epsilon/\epsilon_0)\rceil + 1$ equal-sized groups, denoted by $\{G_1, ..., G_t, ..., G_h\}$, with decreasing budgets $\{\epsilon, \frac{1}{2}\epsilon, \frac{1}{4}\epsilon, ..., \epsilon_t, ..., \epsilon_0\}$.[5] Users are randomly assigned to the groups

[5]Without loss of generality, we assume ϵ/ϵ_0 is a power of 2.

by the collector and perturb their values according to the ϵ_t of the groups they belong to. To guarantee all users have the same privacy budget, those assigned with smaller ϵ_t perturb and report multiple times until the overall privacy budget is depleted. Let V_t' denote the collected values from group G_t and $N_t = \frac{\epsilon N}{\epsilon_t h}$ denote the number of collected values from group G_t.

6.4.2 Aggregating Inter-group Estimations

Upon completing the intra-group distribution estimations, as detailed in Chapter 6.3, the individual estimations are consolidated into one unified distribution. However, the naive method of averaging all of them in equal weights does not provide the optimal data utility — values perturbed with larger ϵ_t have higher accuracy. Hence, the group they belong to deserves higher weight.[6] At the end of this chapter, inspired by [132], this chapter proposes an aggregation strategy that can linearly combine all estimated intra-group distributions $\{D_1, ..., D_t, ..., D_h\}$ into $\tilde{D}$, while achieving minimal overall MSE. Since the MSE is related to the true value which is unknown, this chapter thus only considers the minimal MSE under the worst-case, i.e., when all the original values are either 0 or 1.

Algorithm 11: Distribution Aggregation

Input: Distribution $\{D_1, ..., D_t, ..., D_h\}$

Privacy budgets $\{\epsilon_1, ..., \epsilon_t, ..., \epsilon_h\}$

Output: The aggregated distribution $\tilde{D}$

1: Initialization: $w_t = 0,\ t = \{1, 2, ..., h\}$ **for** $t = 1$ *to* h **do**

2: $w_t = [B_t \sum_{i=1}^{h} \frac{1}{B_i}]^{-1}$, see Theorem 6.1 for B_t. **EndFor**

3: $\tilde{D} = \sum_{t=1}^{h} w_t M_t$

4: **return** $\tilde{D}$

Algorithm 11 shows the detailed optimization procedure of such aggregation. Specifically, the estimated numbers of normal users in G_t can be obtained from $\hat{n}_t = N_t - \hat{m}_t$. All weights

[6] We ignore the influence of poison values, which have been addressed in the previous steps.

w_t of group G_t are initially set to 0 (line 1). Then they are assigned by the formula in line 3, based on which all distributions are combined in line 4. This aggregation satisfies ϵ-LDP according to the parallel composition theorem of LDP [69]. The following Theorem 14 guarantees the weight assignment in line 3 is optimal. Note that Lemma 3 below proves the worse-case variance of SW mechanism for Theorem 14.

Theorem 14. *The MSE of $\tilde{M}$ reaches the minimum $MSE(\tilde{M})_{min} = [\sum_{t=1}^{h} \frac{\hat{n_t}^2}{B_t}]^{-1}$, if the following formula holds:*

$$w_t = \frac{1}{B_t \sum_{i=1}^{h} \frac{1}{B_i}},$$

where $B_t = dVar_{worst}(v'_{tj})$.

Proof. Let $\widetilde{D}_t[j]$ denote the j-th value in group G_t, v'_{tj} the median value in $\widetilde{D}_t[j]$, and $\widetilde{D}_t$ the distribution in G_t. As deducted in literature [42], therefore:

$$\begin{aligned} MSE\left(\widetilde{\mathbf{D}}\right) &= \sum_{j=1}^{d} \mathrm{Var}\left[\widetilde{\mathbf{D}}[j]\right] \\ &= \sum_{j=1}^{d}\sum_{t=1}^{h} \omega_t^2 \left[\widetilde{\mathbf{D}}_t[j]\right] = \sum_{j=1}^{d}\sum_{t=1}^{h} \omega_t^2 Var(v'_{tj}) \end{aligned} \tag{6.6}$$

where $\sum w_t = 1$.

Since $Var(v'tj)$ relies on the input of each user, the worst-case at the maximum variance is considered, i.e., all inputs vtj are either 1 or 0. To simplify, the worst-case variance $Var_{worst}(v'tj)$ is considered, which can be found in Theorem 3. Let $B_t = dVarworst(v'_{tj})$ and Equ. 6.7 can be obtained as follows:

$$MSE(\widetilde{\mathbf{D}}) = \sum_{j=1}^{d}\sum_{t=1}^{h} \omega_t^2 k \,\mathrm{Var}(v'_{tj}) = \sum_{t=1}^{h} \omega_t^2 B_t \tag{6.7}$$

We regard the variance as a function of w_t, and the minimal variance is the extreme point of Equ. 6.7. By the Lagrangian method, we have:

$$\mathcal{L} = \sum_{t=1}^{h} w_t^2 B_t + C_0(1 - \sum_{t=1}^{h} w_t).$$

The first partial derivatives of $\mathcal{L}$ w.r.t. w_t is $\frac{\partial \mathcal{L}}{\partial w_t} = 2w_t B_t - C_0$. Let $\frac{\partial \mathcal{L}}{\partial w_t} = 0$, then we have $wt = \frac{C_0 \hat{n_t}^2}{2B_t}$. Through the restriction $\sum_{t=1}^{h} w_t = 1$, we figure out

$$C_0 = \frac{2}{\hat{n_t}^2 \sum_{t=1}^{h} \frac{1}{B_t}}, \; w_t = \frac{1}{B_t \sum_{i=1}^{h} \frac{1}{B_i}},$$

and the minimal variance of $\tilde{M}$:

$$MSE(\tilde{D})_{min} = [\sum_{t=1}^{h} \frac{\hat{n_t}^2}{B_t}]^{-1}.$$

$\square$

Lemma 3. *When an input value v is perturbed by SW, it yields a perturbed value v' such that*

$$E[v'] = 2bpv + \frac{1}{2}q(2v^2 - 4bv + 2b + 1),$$

and

$$Var[v'] = \frac{q\left(b^3 - (b-v)^3\right)}{3} + \frac{q\left((b+1)^3 - (b+v)^3\right)}{3} + \frac{p\left((b+v)^3 + (b-v)^3\right)}{3} - \left(\frac{q\left(2v^2 - 4bv + 2b + 1\right)}{2} + 2bpv\right)^2$$

When $v = 1$, it achieves the largest variance.

$$\begin{aligned} Var_{worst} &= Var(v'|v=1) \\ &= \frac{p\left((b-1)^3 + (b+1)^3\right)}{3} \\ &- \frac{q\left((b-1)^3 - b^3\right)}{3} - \left(2bp - \frac{q(2b-3)}{2}\right)^2. \end{aligned} \tag{6.8}$$

Proof. Based on the SW perturbation mechanism in Algorithm 3.6, we can infer the following:

$$\begin{aligned} E[v'] &= q\int_{-b}^{v-b} v'dv' + p\int_{v-b}^{v+b} v'dv' + q\int_{v+b}^{1+b} v'dv' \\ &= \frac{v-2b}{2}qv + 2bpv + \frac{v+1+2b}{2}q(1-v) \\ &= 2bpv + \frac{1}{2}q(2v^2 - 4bv + 2b + 1) \\ E[v'^2] &= q\int_{-b}^{v-b} v'^2dv' + p\int_{v-b}^{v+b} v'^2dv' + q\int_{v+b}^{1+b} v'^2dv' \\ &= \frac{1}{3}(q((v-b)^3 - (-b)^3) + p((v+b)^3 - (v-b)^3) \\ &+ q((1+b)^3 - (v+b)^3)). \end{aligned}$$

Further more,

$$\begin{aligned} Var[v'] &= E[v'^2] - (E[v'])^2 \\ &= \frac{1}{3}(q((v-b)^3 - (-b)^3) + p((v+b)^3 - (v-b)^3) + \\ & q((1+b)^3 - (v+b)^3)) - (2bpv + \frac{1}{2}q(2v^2 - 4bv + 2b + 1))^2 \end{aligned}$$

When $v = 1$, it achieves the largest variance.

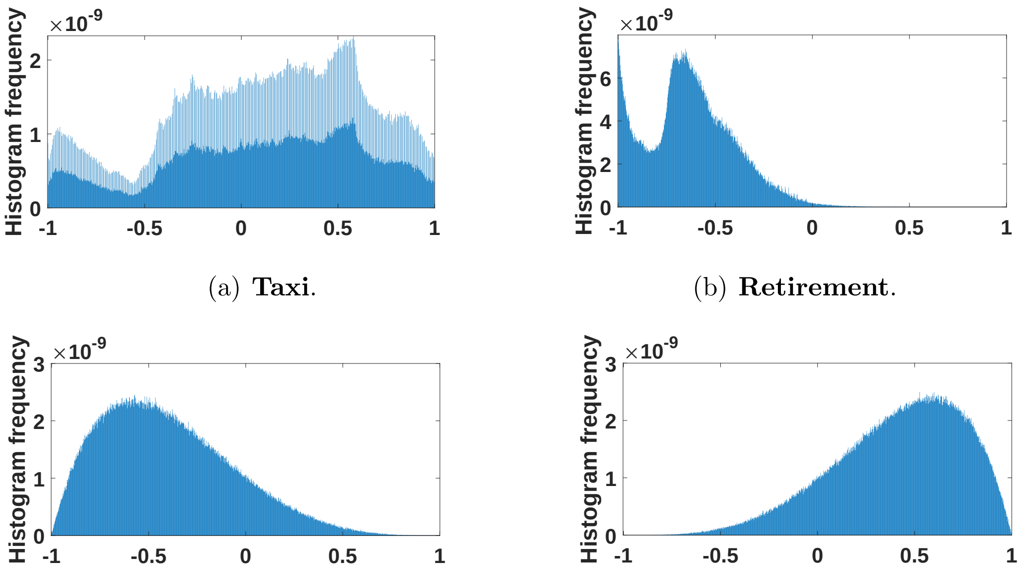
$$\begin{aligned} Var_{worst} = Var(v'|v = 1) = \frac{p\left((b-1)^3 + (b+1)^3\right)}{3} \\ - \frac{q\left((b-1)^3 - b^3\right)}{3} - \left(2bp - \frac{q(2b-3)}{2}\right)^2 \end{aligned} \tag{6.9}$$

In order to keep the expression concise, only b, p, and q are listed here.

$$p = \frac{e^{\epsilon}}{2be^{\epsilon} + 1}, q = \frac{1}{2be^{\epsilon} + 1}, b = \frac{\epsilon e^{\epsilon} - e^{\epsilon} + 1}{2e^{\epsilon}(e^{\epsilon} - 1 - \epsilon)}$$

The parameters b, p, and q are all uniquely determined as functions of the variable ϵ. □

6.5 Experimental Results



(a) **Taxi**.

(b) **Retirement**.

(c) **Beta(2,5)**.

(d) **Beta(5,2)**.

Figure 6.4: Normalized frequencies of datasets

This chapter evaluates the performance of DAP, on both real-world and synthetic datasets. Experiments were conducted using MATLAB R2021a on a PC with Intel i7-10700K RTX 3090 eight-core processor, 128GB RAM, and Windows 10 OS. The source code and datasets are available in [36].

6.5.1 Experiment Setup

Datasets. This chapter adopts two synthetic and two real-world numerical datasets. **Beta(2,5)** and **Beta(5,2)** are two synthetic datasets drawn from Beta distribution [60], each with 1,000,000 samples in the interval $[0, 1]$. **Taxi** [86] is the pick-up time in a day extracted from 2018 January New York Taxi data, which contains 1,048,575 integers from 0 to 86,340 (the number of seconds in 24 hours). **Retirement** [44] is extracted from the San Francisco employee retirement plans, which contains the salary and benefits paid to city employees since fiscal year 2013. The total compensation, which comprises a subset of 606,507 items in the interval $[10000, 60000]$, is employed. All these datasets are then normalized into interval $[0, 1]$. The normalized frequency histograms of all datasets are plotted in Fig. 6.4.

Parameter Setting. This chapter uses SW as our default perturbation mechanism and therefore, the poison values are injected into $[-b, 1 + b]$. This chapter partitions this range into n_d segments, and choose n_p of them to inject poison values, denoted by $Poi_{(n_d,n_p)}$. To ensure the robustness of the attack, this chapter tests different combinations of n_p segments as poison segments and calculate average of the experimental results. This chapter varies the percentage of Byzantine users and poison value distributions to evaluate the scalability of our protocol and adjust n_d and n_p according to specific poison patterns.

This chapter compares the proposed methods, including DE-EMF, DE-EMF* and DE-REMF*, with two existing solutions Ostrich and ME-EMF. Ostrich is the baseline scheme where the existence of Byzantine users is ignored. For the ME-EMF [38], the data collector first identifies the poison side (i.e., left or right), estimates the poison value's distribution there, removes their influence, and then uses EMF-based algorithms (DE-EMF, DE-EMF*, and DE-REMF*) to determine the poison value's distribution. EMF-based algorithms are utilized in DAP, where the minimum privacy budget ϵ_0 is uniformly set to 1/16 across all groups, and the values of ϵ are chosen from $\{1/4, 1/2, 3/2, 4/2, 5/2\}$.

The proportion of Byzantine users is set to 25% and the poison values are uniformly distributed in selected segments by default. The termination condition for EMF-based algorithms is $|l(F)^t - l(F)^{t+1}| < \xi$, where ξ is set to $0.01e^{\epsilon}$. This chapter chooses $u = \lfloor\sqrt{N}\rfloor$

and $w = \lceil(1+2S)u\rceil$. In Algorithm 7, the parameter τ is set to 64. The threshold T for determining whether there are poison values in the segment is set as $0.02 * IQR$ [34].

Performance Metric. The Mean Squared Error (MSE) is widely used in the field of machine learning and statistics to measure the average of the squares of the errors, i.e., the average squared difference between the estimated values and the actual one. Or formally, MSE is defined as

$$MSE = \frac{1}{r}\sum_{i=1}^{r}(Y_i - \hat{Y}_i)^2,$$

where r represents the number of data points. As the MSE in our experiments estimates the mean values, here, the value of r corresponds to the number of experimental rounds. For the experiments, r is chosen as 50. Y_i denotes the actual values, and $\hat{Y}_i$ represents the predicted values.

The Jensen-Shannon Divergence (JSD) [78] is a measurement of the similarity between two probability distributions P and Q. It is defined based on the Kullback-Leibler Divergence (KLD) [32] and additionally requires the divergence symmetry, i.e., the divergence from distribution P to Q is the same as the divergence from Q to P. Specifically, the JSD between P and Q is defined as follows:

$$JSD(P||Q) = \frac{1}{2}D_{KL}(P||M) + \frac{1}{2}D_{KL}(Q||M),$$

where $D_{KL}(P||M)$ and $D_{KL}(Q||M)$ are the Kullback-Leibler Divergences of P and Q from M, respectively, and M is the average of P and Q:

$$M = \frac{1}{2}(P+Q)$$

In this context, the Kullback-Leibler Divergence is defined as:

$$D_{KL}(P||Q) = \sum_{i} P(i) log\frac{P(i)}{Q(i)}$$

6.5.2 Overall Results

Percentage of Byzantine Users. In the first set of experiments, the accuracy of the proportion of Byzantine users obtained by DE-EMF with respect to ϵ is verified in Fig. 6.5.

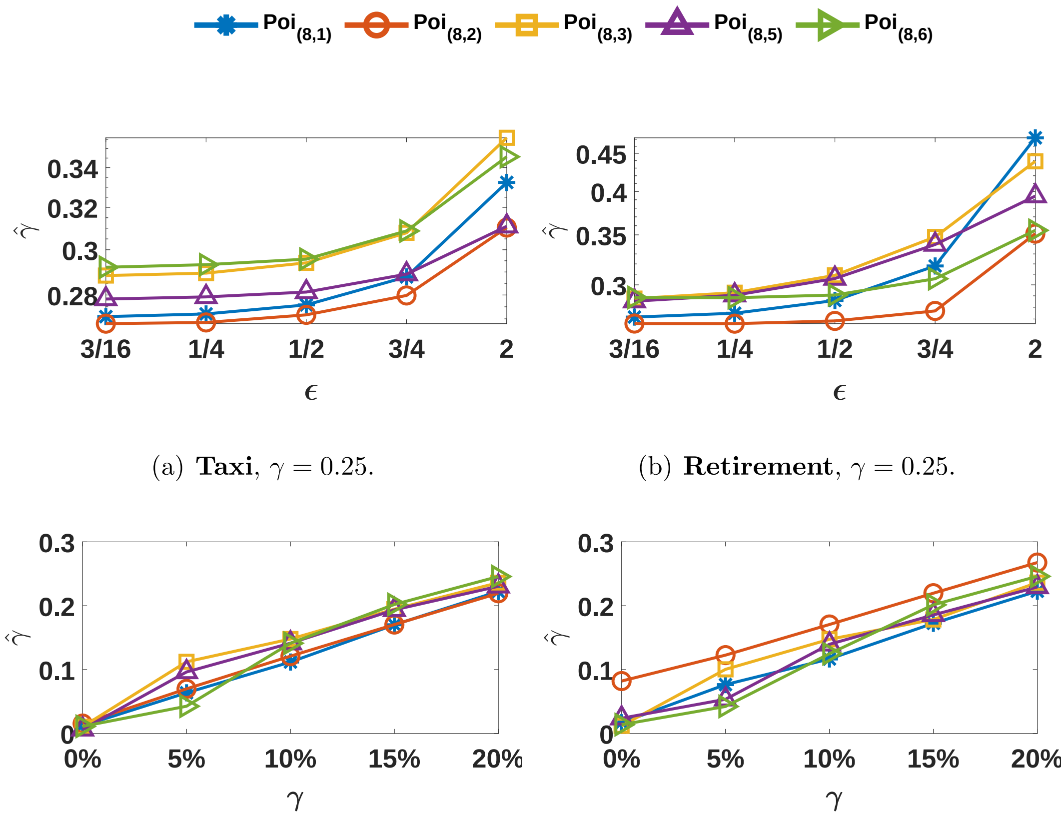
$\gamma = 0.25$ and compare the five poison pattern $Poi_{(8,1)}, Poi_{(8,2)}, Poi_{(8,3)}, Poi_{(8,5)}$ and $Poi_{(8,6)}$ with varying ϵ in Fig. 6.5 (a) (b). When ϵ is large, the estimated proportion $\hat{\gamma}$ deviates from γ. As ϵ decreases, the result becomes more accurate. This is because, according to Theorem 12, when ϵ becomes smaller, EMF distinguishes normal and poison values better. In these figures, regardless of poison values' ranges, datasets and γ, $|\hat{\gamma} - \gamma|$ converges to 0 as $\epsilon \to 0$.

In Fig. 6.5 (c) and (d), $\hat{\gamma}$ is evaluated with varying γ. It can be observed that as γ increases, the estimated value $\hat{\gamma}$ also increases. In most cases, the difference between $\hat{\gamma}$ and γ falls within the range of $[0.02, 0.04]$. However, in the case $Poi_{(8,6)}$, the poison values are distributed across a wide range of the domain, making probing challenging and leading to inadequate estimation performance. Especially when $\gamma = 0$, which means that there is no poison value, the γ estimated by EMF is equivalent to the false positive rate (fpr). In our observation of both of **Beta(2,5)** and **Beta(5,2)**, with $\epsilon_0 = 1/16$, the range of false positives is quite small (from 0 to 0.03), from which we assure that there are no Byzantine users. This illustrates the robustness of our schemes in scenarios where there are no Byzantine users in the system. Even if a few Byzantine users that are actually normal users are misidentified, the impact appears to be negligible.

It is important to note that when $\hat{\gamma}$ is close to 0, there is either no Byzantine users or they successfully evade. Two evasion cases, including input manipulation and random sampling, are considered. Since the latter uniformly increases all data points, this attack has minimal impact on the distribution. As such, only the former is discussed in Chapter 6.16.

Performance of Distribution Estimation. In Fig. 6.10, the JSD is compared with varying ϵ. In most cases, all EMF-based schemes outperform Ostrich as they are somehow able to identify poison values. Among the three proposed schemes, DE-EMF* generally performs better than DE-EMF as it recovers a more precise distribution. Moreover, DE-REMF* performs better than DE-EMF*, which may remove an inappropriate number of values.

In Figs. 6.10 (b) and (c), it is observed that ME-EMF outperforms others. This is because the poison values are skewed to one side, and the size of B^{p+} is smaller compared to other



(a) **Taxi**, $\gamma = 0.25$.

(b) **Retirement**, $\gamma = 0.25$.

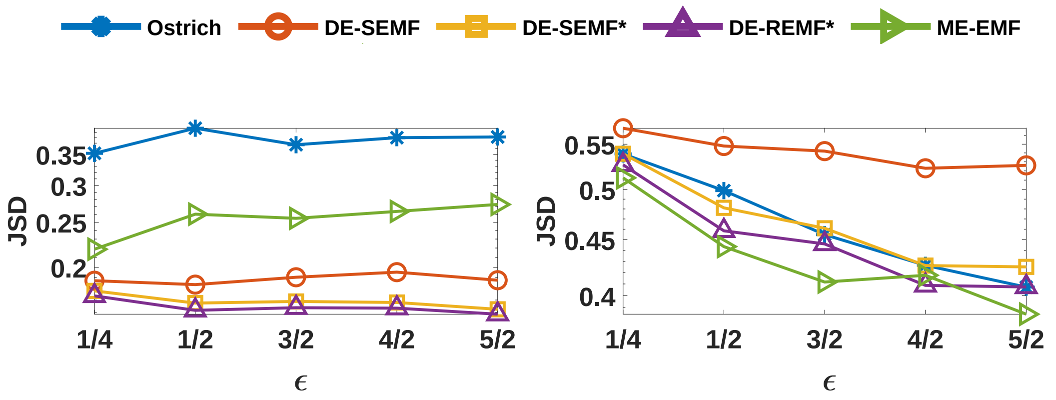
(c) **Beta(2,5)**, $\epsilon = 2$.

(d) **Beta(5,2)**, $\epsilon = 2$.

Figure 6.5: The proportion of Byzantine users estimated by EMF

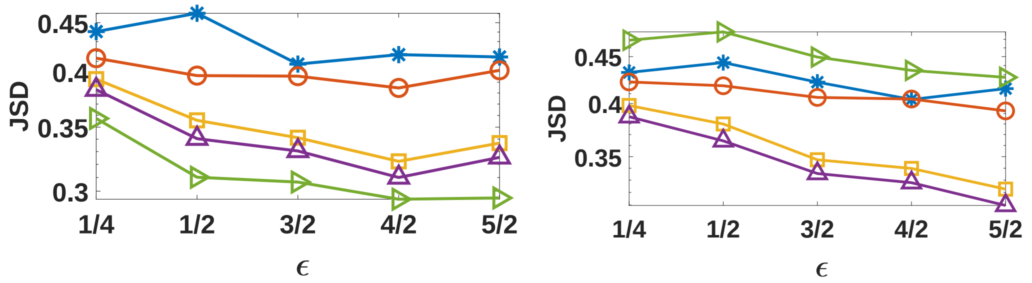
schemes. However, this advantage disappears once n_d exceeds 1, whereby ME-EMF neglects some poison values and worsens the result.

Performance of Mean Estimation. The protocol can restore both the distribution and the mean of normal users. After obtaining the distribution of raw data, the mean of the distribution can be calculated based on the obtained distribution. In Fig. 6.11, in most cases, both DE-REMF* and DE-EMF* outperform the others. They probe and remove poison values compared to Ostrich, and further mitigate the impact of poison values during convergence compared to DE-EMF. In most cases, DE-REMF* outperforms DE-EMF* because it leverages additional information from $\hat{\gamma}$, leading to a reduction in the error of the convergence result. ME-EMF occasionally outperforms (as shown in Fig. 6.11 (e) and (g)) due to its correspondence with fewer EMF buckets. Particularly, it shows enhanced



(a) **Taxi**, $Poi_{(2,1)}$.

(b) **Retirement**, $Poi_{(2,1)}$.



(c) **Beta(2,5)**, $Poi_{(2,1)}$.

(d) **Beta(5,2)**, $Poi_{(2,1)}$.

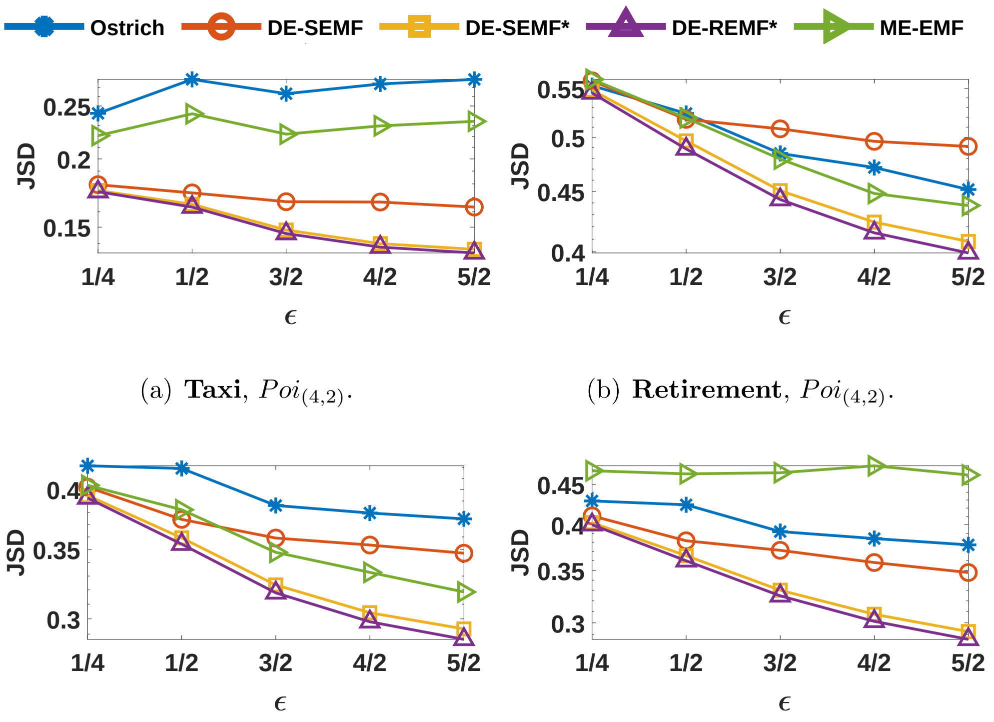
Figure 6.6: The results for JSD w.r.t. ϵ

performance when Byzantine users focus their attack on a single segment.

6.5.3 Robustness Study

Fig. 6.15 studies the robustness of different methods, by varying different datasets, proportions of Byzantine users and distributions of poison values at $\epsilon = 2$.

Robustness on Datasets. Fig. 6.15 (a)-(d) demonstrate that our proposed schemes yield lower JSD than both Ostrich and ME-EMF across all datasets. However, the final JSD performance is highly correlated with the distribution of each dataset. Specifically, the **Taxi** presents the lowest JSD, while other datasets show relatively higher values. This is due to the smaller kurtosis of the **Taxi** dataset (indicating a more uniform distribution), which makes poison values easier to probe. Conversely, the latter three distributions display significantly larger kurtosis. When poison values are near the peak of these distributions, our



(a) **Taxi**, $Poi_{(4,2)}$.

(b) **Retirement**, $Poi_{(4,2)}$.

(c) **Beta(2,5)**, $Poi_{(4,2)}$.

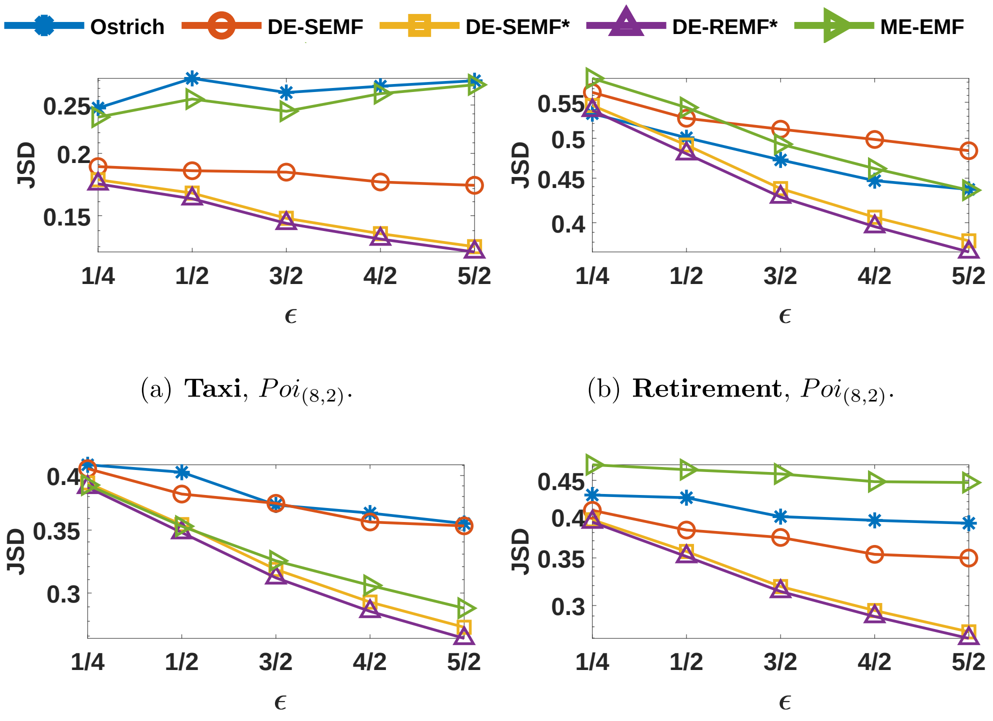
(d) **Beta(5,2)**, $Poi_{(4,2)}$.

Figure 6.7: The results for JSD w.r.t. ϵ

schemes perform less effectively, resulting in worse performance than on the **Taxi** dataset. Furthermore, the symmetric nature of the **beta(2,5)** and **beta(5,2)** distributions leads to similar JSD results for both.

Robustness on Poison Pattern. Fig. 6.15 (e)-(h) demonstrate the robustness on different poison patterns in **Beta(2,5)** with privacy budget $\epsilon = 2$. In Fig. 6.15 (e), when $n_p = 1$ and the poison pattern is $Poi_{(2,1)}$, ME-EMF performs the best. This is because its selected B^{p+} is closer to B^p than those selected in DE-EMF*/DE-REMF*. However, the performance of ME-EMF declines as n_d increases and B^{p+} deviates increasingly from B^p.

In Fig. 6.15 (f)-(h), different poison patterns at the same n_p/n_d are compared. When n_d is relatively small, poison values are more concentrated, and the performance of Ostrich is rather poor. However, as n_d increases, the attack capability of poison values decreases, and the performance of Ostrich improves. The changes in n_d have little impact on DE-EMF and DE-EMF*. Even in situations such as $Poi_{(32,16)}$, where the poison values are



(a) **Taxi**, $Poi_{(8,2)}$.

(b) **Retirement**, $Poi_{(8,2)}$.

(c) **Beta(2,5)**, $Poi_{(8,2)}$.

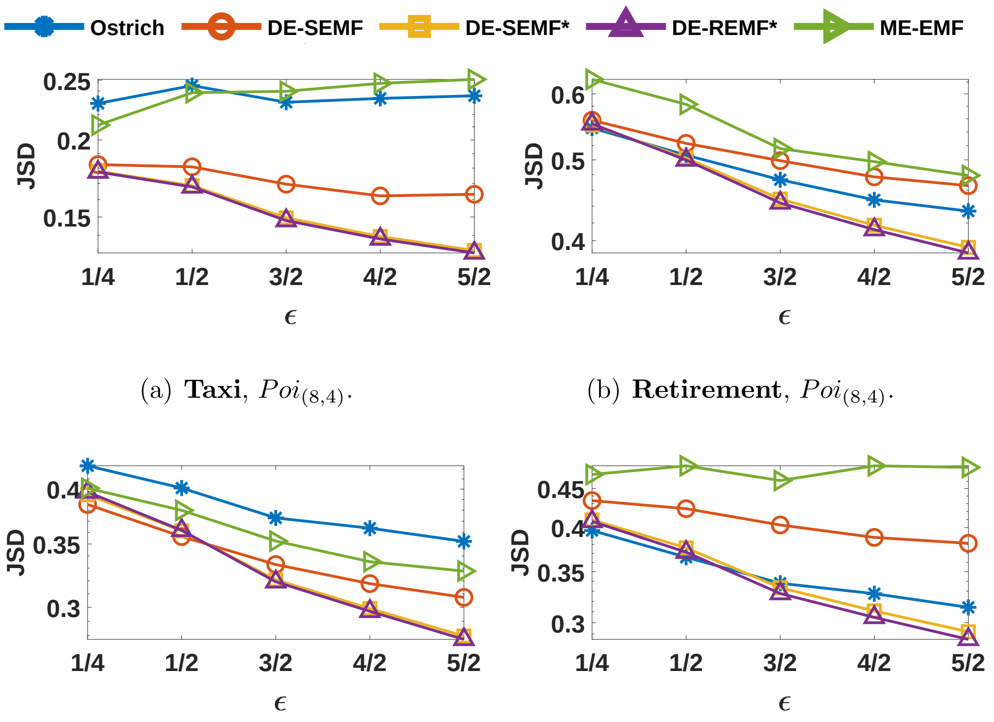
(d) **Beta(5,2)**, $Poi_{(8,2)}$.

Figure 6.8: The results for JSD w.r.t. ϵ

almost uniformly distributed across the domain, reducing the impact of the attack, probing becomes challenging and thus Ostrich performs best. Nonetheless, the proposed schemes achieve comparable performance to Ostrich. This shows the robustness of the scheme in cases where poison values are difficult to distinguish from normal values.

In Fig. 6.15 (h), n_d is set to 8, and the poison patterns are set between $Poi_{(8,1)}$ and $Poi_{(8,7)}$ inclusive. Ostrich performs better as n_p increases and the poison values become sparser. However, DE-EMF* and DE-REMF* still outperform Ostrich. Particularly, the proposed schemes perform well even when the poison values almost occupy the entire domain in the $Poi_{(8,7)}$ poison pattern.

Robustness on Distribution of Poison Values. Fig. 6.15 (i)-(l) show how JSD varies with respect to different distributions of poison values. It can be observed that the performance greatly varies with different distributions, and the proposed schemes consistently outperform the others. It is noteworthy that all schemes have a higher JSD when the attack



(a) **Taxi**, $Poi_{(8,4)}$.

(b) **Retirement**, $Poi_{(8,4)}$.

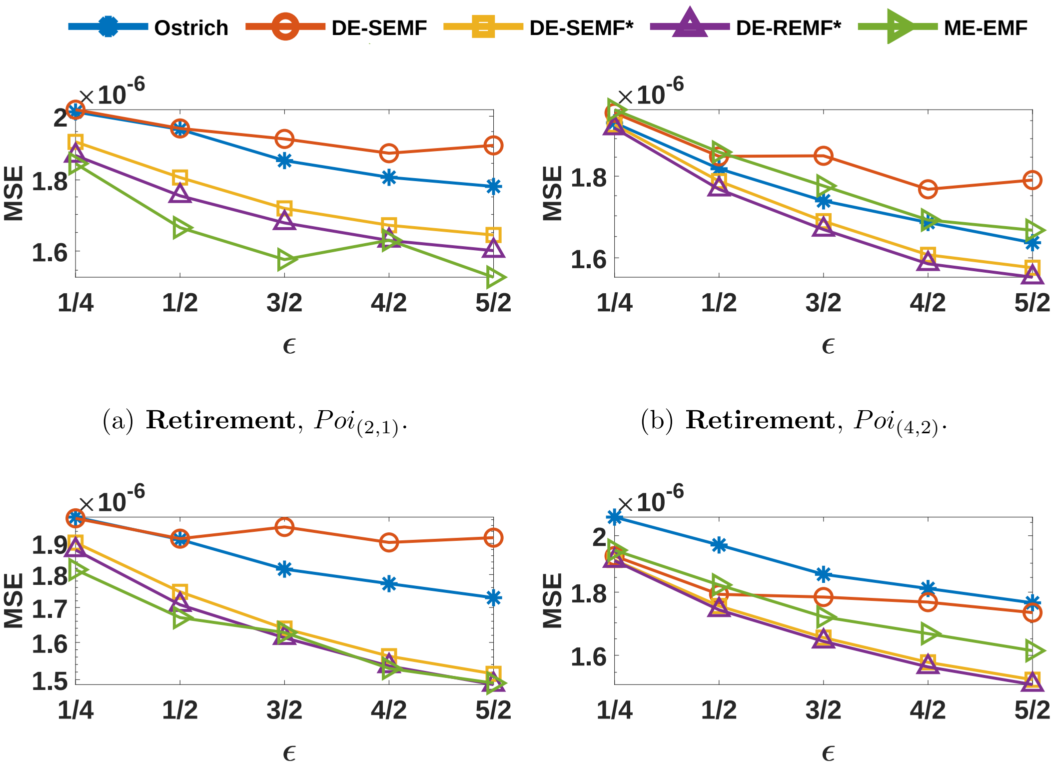
(c) **Beta(2,5)**, $Poi_{(8,4)}$.

(d) **Beta(5,2)**, $Poi_{(8,4)}$.

Figure 6.9: The results for JSD w.r.t. ϵ

distribution is Gaussian, whose unique shape can be characterized by a high peak in the middle that tapers off to lower values at the tails. It becomes significantly challenging to effectively identify and probe the poison values that are located near the peak. In contrast, the nearly flat uniform distribution simplifies the identification of poison values, leading to a lower JSD. Similarly, Beta(1,6) and Beta(6,1) distributions, which do not have any peak, also result in a better performance.

Robustness on Proportion of Byzantine Users. Fig. 6.15 (m)-(p) demonstrate that, even with increasing Byzantine users, the proposed schemes still effectively eliminate the impact of Byzantine attacks. As the number of Byzantine users increases (even when the proportion $\gamma = 0.5$), the performance of Ostrich worsens while the proposed schemes are quite stable. In Fig. 6.15 (n) and (o), DE-EMF performs worse than Ostrich. This is due to the bias introduced in the distribution for normal users when estimating the distribution for Byzantine users. However, both the optimized methods DE-EMF* and DE-REMF*



(a) **Retirement**, $Poi_{(2,1)}$. (b) **Retirement**, $Poi_{(4,2)}$.

(c) **Retirement**, $Poi_{(8,1)}$. (d) **Retirement**, $Poi_{(8,2)}$.

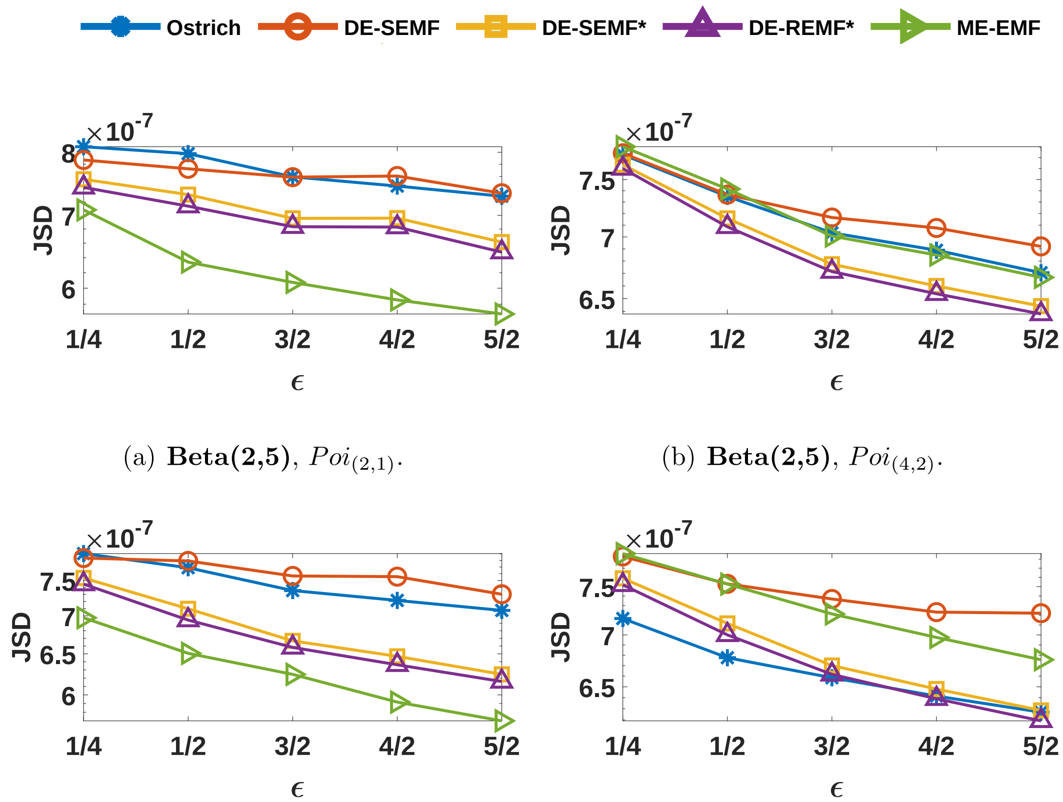
Figure 6.10: The results for JSD w.r.t. ϵ

correct these errors and achieve lower JSD.

6.5.4 Discussion.

Combating Input Manipulation Attacks. Let's consider the scenario where Byzantine users are aware of our proposed DAP technique and attempt to evade it by employing an input manipulation attack (IMA) [70]. Specifically, Byzantine users generate poison data within the range of $[0, 1]$ and then perturb them using an LDP mechanism to make them less detectable before sending to the data collector.

In Fig. 6.16 (a), the number of Byzantine users $\hat{\gamma}$ is evaluated under the IMA. More specifically, when Byzantine users inject their poison values as 1 and then perturb them like normal users. It is observed that for all four datasets with $\epsilon_0 = 1/16$, the false positives range from 0.02 to 0.04, indicating a relatively low rate. This suggests that there are either no



(a) **Beta(2,5)**, $Poi_{(2,1)}$.

(b) **Beta(2,5)**, $Poi_{(4,2)}$.

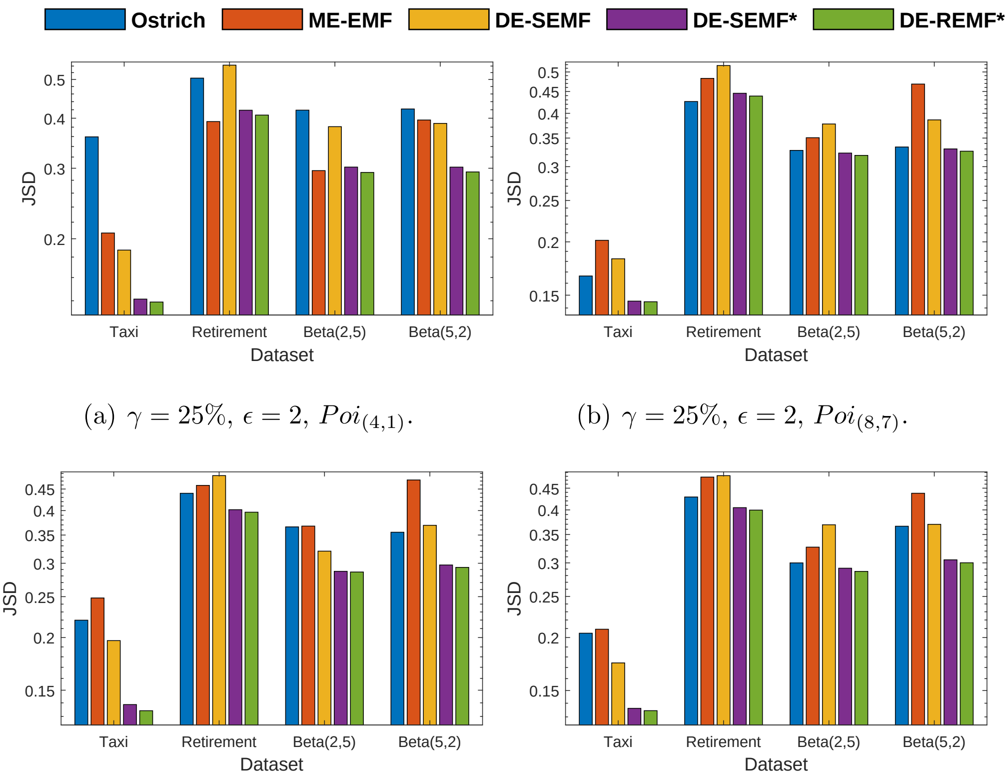
(c) **Beta(2,5)**, $Poi_{(8,1)}$.

(d) **Beta(2,5)**, $Poi_{(8,2)}$.

Figure 6.11: The results for mean estimation w.r.t. ϵ

Byzantine users or the Byzantine users successfully evade, making it challenging for EMF to filter out these Byzantine users due to the perturbation. However, the utility can be further enhanced by utilizing existing detection techniques, such as k-means clustering [70], as illustrated in Fig. 6.16 (b).

Fig. 6.16 (b) evaluates the IMA on Taxi, i.e., Byzantine users generating an input poison value g and then strictly following the perturbation mechanism to make it less detectable. To integrate EMF with the existing k-means-based defense, EMF is first used to determine whether $\hat{\gamma}$ is relatively small (i.e., evading), as shown in Fig. 6.16 (d). Then, EMF is employed to estimate the input distribution by setting $\hat{\gamma} = 0$, and finally, the mean is evaluated using k-means. Fig. 6.16 (b) demonstrates that through this integration (referred to as EMF-based), the estimation accuracy under IMA can be further enhanced. When $g = -1$, the mean squared error (MSE) of the EMF-based approach ranges between 1.40×10^{-7}



(a) $\gamma = 25\%$, $\epsilon = 2$, $Poi_{(4,1)}$.

(b) $\gamma = 25\%$, $\epsilon = 2$, $Poi_{(8,7)}$.

(c) $\gamma = 25\%$, $\epsilon = 2$, $Poi_{(16,4)}$.

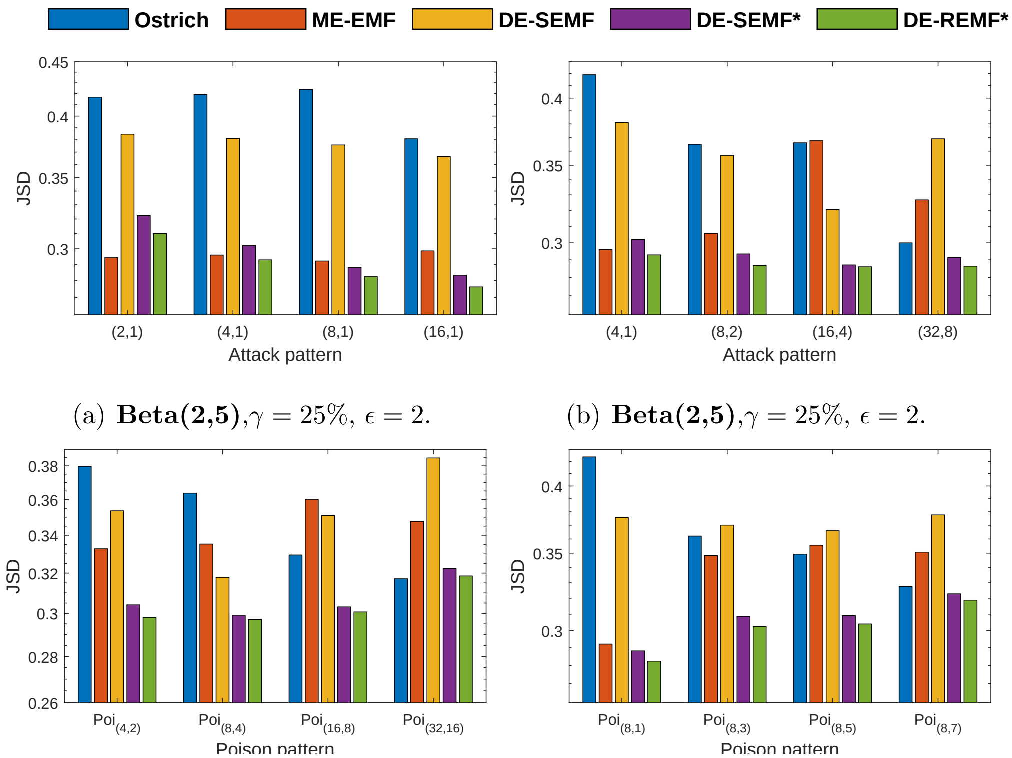
(d) $\gamma = 25\%$, $\epsilon = 2$, $Poi_{(32,8)}$.

Figure 6.12: The results for JSD with different parameters

to 1.44×10^{-7}, while the MSE of k-means alone ranges between 1.77×10^{-7} to 1.88×10^{-7}, resulting in a 28

Frequency Estimation on Categorical Data. DAP is not limited to numerical data and can be generalized to categorical data, e.g., frequency estimation for categorical data. This is because $\hat{x}$ derived in Algorithms 6 and 8 is essentially the frequency histogram of numerical/categorical data. To exemplify how to use DAP to estimate the frequency of categorical data, let's consider k-RR [62, 112], a common LDP mechanism for categorical data.

The schemes are evaluated under frequency estimation on categorical data using the COVID-19 dataset, which records the number of coronavirus disease 2019 deaths for females in California as of December 14, 2022, by age [106]. All death records are divided into 15 age groups, and every record is perturbed locally by k-RR. In Fig. 6.16 (c), Byzantine users



(a) **Beta(2,5)**,$\gamma = 25\%$, $\epsilon = 2$. (b) **Beta(2,5)**,$\gamma = 25\%$, $\epsilon = 2$.

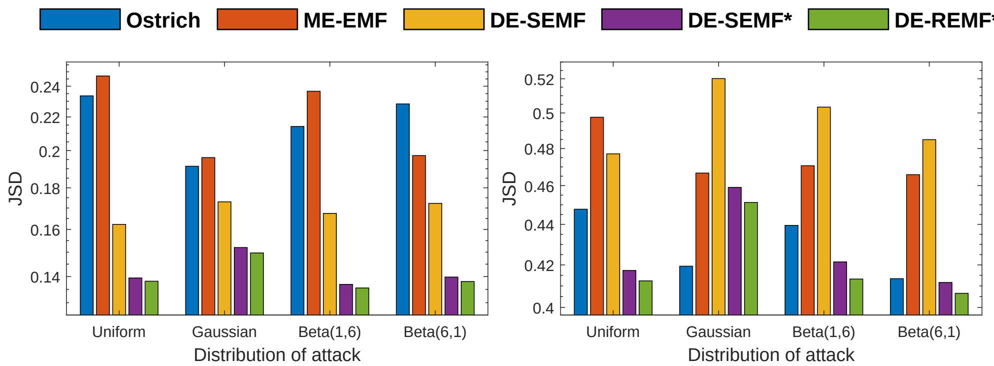
(c) **Beta(2,5)**,$\gamma = 25\%$, $\epsilon = 2$. (d) **Beta(2,5)**,$\gamma = 25\%$, $\epsilon = 2$.

Figure 6.13: The results for JSD with different parameters

are injected into the 10th group only, and it is observed that the MSE of Ostrich is about 0.1 and keeps steady regardless of ϵ, while that of the proposed schemes is lower than 0.01 and decreases significantly with respect to ϵ. When Byzantine users uniformly inject poison values into the 10th, 11th, and 12th groups, as shown in Fig. 6.16 (d), the MSE of Ostrich still significantly underperforms the proposed schemes. This experiment shows that the DAP schemes can also work well in other statistics and data types than mean estimation on numerical data.

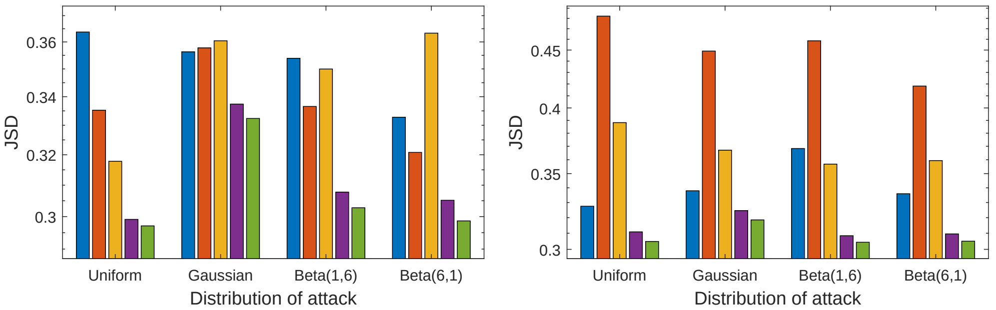
6.6 Summary

This chapter studies the problem of distribution estimation while addressing arbitrarily distributed Byzantine attacks in LDP. Unlike previous solutions, the proposed approach



(a) **Taxi**, $\epsilon = 2, \gamma = 25\%$.

(b) **Retirement**, $\epsilon = 2, \gamma = 25\%$.



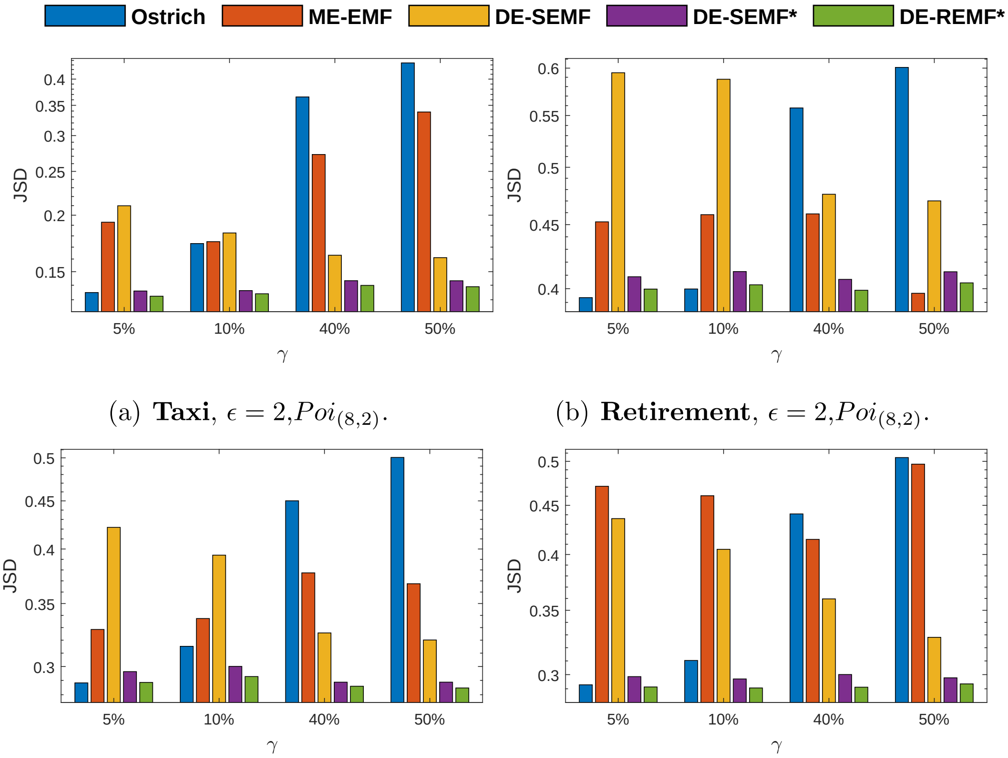
(c) **Beta(2,5)**, $\epsilon = 2, \gamma = 25\%$.

(d) **Beta(5,2)**, $\epsilon = 2, \gamma = 25\%$.

Figure 6.14: The results for JSD with different parameters

does not require any prior knowledge about the attack methods or the distribution of poison values. A new algorithm, SEMF, is developed that can approximately identify the range of these poison values. Based on this information, the distribution is estimated through DE-EMF and two additional optimized methods, DE-EMF* and DE-REMF*.

To further improve performance and security, a group-based method called DAP is introduced, which optimizes the estimates and reduces MSE. Through extensive experiments with both simulated and real-world data, it is demonstrated that DAP is effective, accurate, and robust against various attack settings in LDP systems.



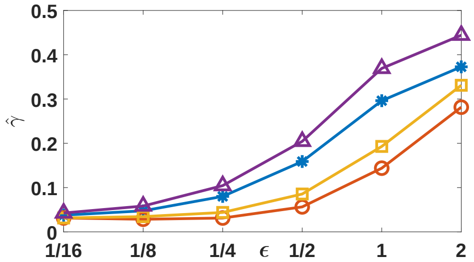
(a) **Taxi**, $\epsilon = 2$,$Poi_{(8,2)}$.

(b) **Retirement**, $\epsilon = 2$,$Poi_{(8,2)}$.

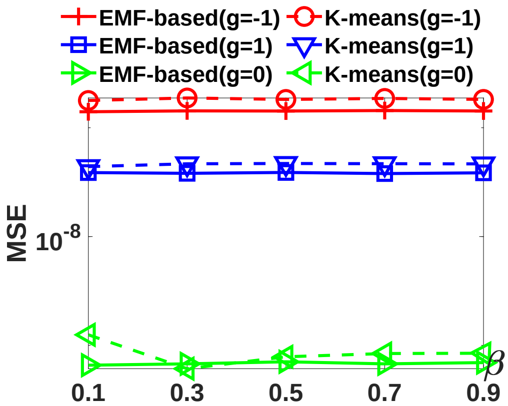
(c) **Beta(2,5)**, $\epsilon = 2$,$Poi_{(8,2)}$.

(d) **Beta(5,2)**, $\epsilon = 2$,$Poi_{(8,2)}$.

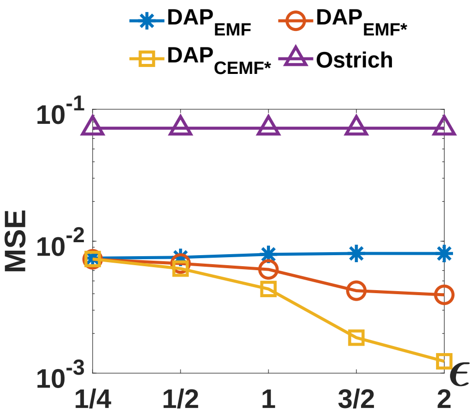
Figure 6.15: The results for JSD with different parameters



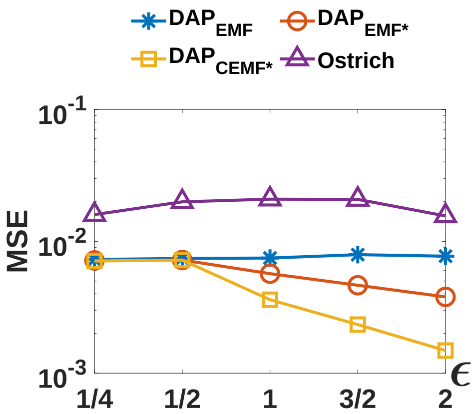
(a) IMA, $\gamma = 0.25$.



(b) **Taxi**, IMA, $\gamma = 0.25$.



(c) **COVID-19**, Poi_{10}.



(d) **COVID-19**, $Poi_{10,11,12}$.

Figure 6.16: Comparison with k-means-based defense for (a) (b) and extension to frequency estimation for (c) (d)

Chapter 7

Conclusions and Future Works

In this thesis, we have conducted an in-depth study on LDP under adverse circumstances, focusing on enhancing both utility and security. Our research addresses several key challenges that represent challenges for LDP systems: the utility degradation in high-dimensional data scenarios, where we optimized privacy budget allocation among correlated attributes; the inefficiency in processing sparse data with low-frequency values, where we developed methods to identify top-k values through budget allocation and reinforcement learning; and the vulnerability to Byzantine attacks, where we established robust LDP protocols by filtering out poisoned data based on varying user behaviors. Our comprehensive study demonstrates that our proposed solutions significantly outperform existing approaches, both theoretically and experimentally. These methods effectively improve utility and enhance the security of data collection in LDP systems under adverse circumstances.

Looking ahead, we aim to emphasize three critical directions that will shape the trajectory of future research on LDP, focusing on both utility and security:

Personalized Privacy Developing adaptive LDP frameworks that allow for personalized privacy settings based on individual user preferences or legal requirements can ensure a more tailored privacy experience.

Interplay with Other Privacy Technologies Exploring how LDP can work in conjunction with other privacy-preserving technologies, such as secure multi-party computation

(MPC) [55] and homomorphic encryption (HE) [131], could lead to hybrid systems that leverage the strengths of multiple approaches.

Robustness against Inference Attacks Although LDP provides strong privacy guarantees, sophisticated inference attacks [119] can still pose risks. Investigating new threats and enhancing the robustness of LDP mechanisms against such attacks is vital for maintaining user trust.

References

[1] Breast cancer wisconsin (diagnostic) data set. https://www.kaggle.com/datasets/uciml/breast-cancer-wisconsin-data, 2016. Accessed: September, 2016, [Online].

[2] Cambridge analytica and facebook: The scandal and the fall-out so far. https://www.nytimes.com/2018/04/04/us/politics/cambridge-analytica-scandal-fallout.html, 2018. Accessed: April 4, 2018. [Online].

[3] Marriott hacking exposes data of up to 500 million guests. https://www.nytimes.com/2018/11/30/business/marriott-data-breach.html, 2018. Accessed: Nov. 30, 2018. [Online].

[4] Yahoo's 2013 email hack actually compromised three billion accounts. https://www.wired.com/story/yahoo-breach-three-billion-accounts/, 2018. Accessed: OCT 3, 2017. [Online].

[5] Iot signals report. https://azure.microsoft.com/mediahandler/files/resourcefiles/iot-signals/IoT-Signals-Microsoft-072019.pdf, July 2019.

[6] Equifax data breach faq: What happened, who was affected, what was the impact? https://www.csoonline.com/article/567833/equifax-data-breach-faq-what-happened-who-was-affected-what-was-the-impact.html, 2020. Accessed: Feb 12, 2020, [Online].

[7] U. S. Vaishampayan G. Kapoor J. Freudinger V. V. Prakash A. Legendre A. G. Thakurta, A. H. Vyrros and S. Duplinsky. Emoji frequency detection and deep link frequency, us patent.

[8] Jayadev Acharya, Ziteng Sun, and Huanyu Zhang. Hadamard response: Estimating distributions privately, efficiently, and with little communication. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1120–1129. PMLR, 2019.

[9] Arpit Agarwal, Shivani Agarwal, Sepehr Assadi, and Sanjeev Khanna. Learning with limited rounds of adaptivity: Coin tossing, multi-armed bandits, and ranking from pairwise comparisons. In *Conference on Learning Theory*, pages 39–75. PMLR, 2017.

[10] Shipra Agrawal and Navin Goyal. Analysis of thompson sampling for the multi-armed bandit problem. In *Conference on learning theory*, pages 39–1. JMLR Workshop and Conference Proceedings, 2012.

[11] P Arthur. Maximum likelihood from incomplete data via the em algorithm. *Journal of The Royal Statistical Society Series B*, 39(1):1–38, 1977.

[12] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2):235–256, 2002.

[13] Peter Auer and Ronald Ortner. Ucb revisited: Improved regret bounds for the stochastic multi-armed bandit problem. *Periodica Mathematica Hungarica*, 61(1-2):55–65, 2010.

[14] Baruch Awerbuch, Reza Curtmola, David Holmer, Cristina Nita-Rotaru, and Herbert Rubens. Mitigating byzantine attacks in ad hoc wireless networks. *Department of Computer Science, Johns Hopkins University, Tech. Rep. Version*, 1:16, 2004.

[15] A.T. Azar and A.E. Hassanien. Dimensionality reduction of medical big data using neural-fuzzy classifier. *Soft Computing*, 19:1115–1127, 2015.

[16] Raef Bassily, Kobbi Nissim, Uri Stemmer, and Abhradeep Guha Thakurta. Practical locally private heavy hitters. In *Advances in Neural Information Processing Systems*, pages 2288–2296, 2017.

[17] Raef Bassily and Adam Smith. Local, private, efficient protocols for succinct histograms. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 127–135, 2015.

[18] Donald A Berry and Stephen G Eick. Adaptive assignment versus balanced randomization in clinical trials: a decision analysis. *Statistics in medicine*, 14(3):231–246, 1995.

[19] Donald A Berry and Bert Fristedt. Bandit problems: sequential allocation of experiments (monographs on statistics and applied probability). *London: Chapman and Hall*, 5(71-87):7–7, 1985.

[20] Jeff A Bilmes et al. A gentle tutorial of the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models. *International Computer Science Institute*, 4(510):126, 1998.

[21] Andrea Bittau, Ulfar Erlingsson, Petros Maniatis, Ilya Mironov, Ananth Raghunathan, David Lie, Mitch Rudominer, Ushasree Kode, Julien Tinnes, and Bernhard Seefeld. Prochlo: Strong privacy for analytics in the crowd. In *Proceedings of the 26th Symposium on Operating Systems Principles*, pages 441–459, 2017.

[22] Sébastien Bubeck, Nicolo Cesa-Bianchi, et al. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 5(1):1–122, 2012.

[23] Swapna Buccapatnam, Fang Liu, Atilla Eryilmaz, and Ness B Shroff. Reward maximization under uncertainty: Leveraging side-observations on networks. *arXiv preprint arXiv:1704.07943*, 2017.

[24] Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. Data poisoning attacks to local differential privacy protocols. In *30th USENIX Security Symposium (USENIX Security 21)*, 2021.

[25] Nicholas Carlini. Poisoning the unlabeled dataset of {Semi-Supervised} learning. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 1577–1592, 2021.

[26] Shih-Hao Chang and Zhi-Rong Chen. Protecting mobile crowd sensing against sybil attacks using cloud based trust management system. *Mobile Information Systems*, 2016, 2016.

[27] Chun-Hung Chen, Donghai He, Michael Fu, and Loo Hay Lee. Efficient simulation budget allocation for selecting an optimal subset. *INFORMS Journal on Computing*, 20(4):579–595, 2008.

[28] Rui Chen, Benjamin CM Fung, S Yu Philip, and Bipin C Desai. Correlated network data publication via differential privacy. *The VLDB Journal*, 23(4):653–676, 2014.

[29] Rui Chen, Haoran Li, A Kai Qin, Shiva Prasad Kasiviswanathan, and Hongxia Jin. Private spatial data aggregation in the local setting. In *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*, pages 289–300. IEEE, 2016.

[30] Albert Cheu, Adam Smith, and Jonathan Ullman. Manipulation attacks in local differential privacy. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 883–900. IEEE, 2021.

[31] G. Cormode, T. Kulkarni, and D. Srivastava. Marginal release under local differential privacy. In *SIGMOD*, pages 131–146. ACM, 2018.

[32] Imre Csiszár. I-divergence geometry of probability distributions and minimization problems. *The annals of probability*, pages 146–158, 1975.

[33] Sabyasachi Dash, Sushil Kumar Shakyawar, Mohit Sharma, and Sandeep Kaushik. Big data in healthcare: management, analysis and future prospects. *Journal of big data*, 6(1):1–25, 2019.

[34] Frederik Michel Dekking, Cornelis Kraaikamp, Hendrik Paul Lopuhaä, and Ludolf Erwin Meester. *A Modern Introduction to Probability and Statistics: Understanding why and how*, volume 488. Springer, 2005.

[35] Bolin Ding, Janardhan Kulkarni, and Sergey Yekhanin. Collecting telemetry data privately. *arXiv preprint arXiv:1712.01524*, 2017.

[36] Rong Du. The source code for differential aggregation protocol. `https://github.com/RONGDUGithub/DAP`, 2023.

[37] Rong Du, Qingqing Ye, Yue Fu, and Haibo Hu. Collecting high-dimensional and correlation-constrained data with local differential privacy. In *2021 18th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, pages 1–9. IEEE, 2021.

[38] Rong Du, Qingqing Ye, Yue Fu, Haibo Hu, Jin Li, Chengfang Fang, and Jie Shi. Differential aggregation against general colluding attackers. *arXiv preprint arXiv:2302.09315*, 2023.

[39] John C Duchi, Michael I Jordan, and Martin J Wainwright. Local privacy and statistical minimax rates. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 429–438. IEEE, 2013.

[40] John C Duchi, Michael I Jordan, and Martin J Wainwright. Privacy aware learning. *Journal of the ACM (JACM)*, 61(6):1–57, 2014.

[41] John C Duchi, Michael I Jordan, and Martin J Wainwright. Minimax optimal procedures for locally private estimation. *Journal of the American Statistical Association*, 113(521):182–201, 2018.

[42] Cynthia Dwork. Differential privacy: A survey of results. In *International conference on theory and applications of models of computation*, pages 1–19. Springer, 2008.

[43] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer, 2006.

[44] The San Francisco employee retirement plans in 2013. Retirement dataset. `https://www.kaggle.com/datasets/san-francisco/sf-employee-compensation`, 2013.

[45] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*, pages 1054–1067, 2014.

[46] Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Gong. Local model poisoning attacks to {Byzantine-Robust} federated learning. In *29th USENIX Security Symposium (USENIX Security 20)*, pages 1605–1622, 2020.

[47] Minghong Fang, Neil Zhenqiang Gong, and Jia Liu. Influence function based data poisoning attacks to top-n recommender systems. In *Proceedings of The Web Conference 2020*, pages 3019–3025, 2020.

[48] Giulia Fanti, Vasyl Pihur, and Úlfar Erlingsson. Building a rappor with the unknown: Privacy-preserving learning of associations and data dictionaries. *arXiv preprint arXiv:1503.01214*, 2015.

[49] Christian Fischer, Zachary A Pardos, Ryan Shaun Baker, Joseph Jay Williams, Padhraic Smyth, Renzhe Yu, Stefan Slater, Rachel Baker, and Mark Warschauer. Mining big data in education: Affordances and challenges. *Review of Research in Education*, 44(1):130–160, 2020.

[50] Roger Fletcher. *Practical methods of optimization.* John Wiley & Sons, 2013.

[51] Ujwal Gadiraju, Ricardo Kawase, Stefan Dietze, and Gianluca Demartini. Understanding malicious behavior in crowdsourcing platforms: The case of online surveys. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 1631–1640, 2015.

[52] Pratik Gajane, Tanguy Urvoy, and Emilie Kaufmann. Corrupt bandits for preserving local privacy. In *Algorithmic Learning Theory*, pages 387–412. PMLR, 2018.

[53] Aurélien Garivier and Olivier Cappé. The kl-ucb algorithm for bounded stochastic bandits and beyond. In *Proceedings of the 24th annual conference on learning theory*, pages 359–376. JMLR Workshop and Conference Proceedings, 2011.

[54] Quan Geng, Peter Kairouz, Sewoong Oh, and Pramod Viswanath. The staircase mechanism in differential privacy. *IEEE Journal of Selected Topics in Signal Processing*, 9(7):1176–1184, 2015.

[55] Oded Goldreich. Secure multi-party computation. *Manuscript. Preliminary version*, 78(110):1–108, 1998.

[56] Xiaolan Gu, Ming Li, Yueqiang Cheng, Li Xiong, and Yang Cao. PCKV: locally differentially private correlated key-value data collection with optimized utility. In *USENIX Security Symposium*, 2020.

[57] Godfrey H Hardy, John E Littlewood, and George Pólya. *Inequalities (Cambridge mathematical library).* cambridge university press, 1934.

[58] Nan Hu, Ling Liu, and Vallabh Sambamurthy. Fraud detection in online consumer reviews. *Decision Support Systems*, 50(3):614–626, 2011.

[59] Vittorio P Illiano and Emil C Lupu. Detecting malicious data injections in wireless sensor networks: A survey. *ACM Computing Surveys (CSUR)*, 48(2):1–33, 2015.

[60] Norman L Johnson, Samuel Kotz, and Narayanaswamy Balakrishnan. *Continuous univariate distributions, volume 2*, volume 289. John wiley & sons, 1995.

[61] Peter Kairouz, Keith Bonawitz, and Daniel Ramage. Discrete distribution estimation under local privacy. *arXiv preprint arXiv:1602.07387*, 2016.

[62] Peter Kairouz, Sewoong Oh, and Pramod Viswanath. Extremal mechanisms for local differential privacy. In *Advances in neural information processing systems*, pages 2879–2887, 2014.

[63] Shivaram Kalyanakrishnan and Peter Stone. Efficient selection of multiple bandit arms: Theory and practice. In *ICML*, 2010.

[64] Shiva Prasad Kasiviswanathan, Homin K Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. What can we learn privately? *SIAM Journal on Computing*, 40(3):793–826, 2011.

[65] Fumiyuki Kato, Yang Cao, and Masatoshi Yoshikawa. Preventing manipulation attack in local differential privacy using verifiable randomization mechanism. *arXiv preprint arXiv:2104.06569*, 2021.

[66] Stéphane Lavertu. We all need help:big data and the mismeasure of public administration. *Public administration review*, 76(6):864–872, 2016.

[67] Husheng Li and Zhu Han. Catching attacker (s) for collaborative spectrum sensing in cognitive radio systems: An abnormality detection approach. In *2010 IEEE Symposium on New Frontiers in Dynamic Spectrum (DySPAN)*, pages 1–12. IEEE, 2010.

[68] Lihong Li, Wei Chu, John Langford, and Robert E Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 661–670, 2010.

[69] Ninghui Li, Min Lyu, Dong Su, and Weining Yang. Differential privacy: From theory to practice. *Synthesis Lectures on Information Security, Privacy, & Trust*, 8(4):1–138, 2016.

[70] Xiaoguang Li, Neil Zhenqiang Gong, Ninghui Li, Wenhai Sun, and Hui Li. Fine-grained poisoning attacks to local differential privacy protocols for mean and variance estimation. *arXiv preprint arXiv:2205.11782*, 2022.

[71] Zitao Li, Tianhao Wang, Milan Lopuhaä-Zwakenberg, Ninghui Li, and Boris Škoric. Estimating numerical distributions under local differential privacy. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, pages 621–635, 2020.

[72] Fang Liu. Generalized gaussian mechanism for differential privacy. *IEEE Transactions on Knowledge and Data Engineering*, 31(4):747–756, 2018.

[73] Fang Liu, Xiuzhen Cheng, and Dechang Chen. Insider attacker detection in wireless sensor networks. In *IEEE INFOCOM 2007-26th IEEE International Conference on Computer Communications*, pages 1937–1945. IEEE, 2007.

[74] Fang Liu, Sinong Wang, Swapna Buccapatnam, and Ness Shroff. Ucboost: a boosting approach to tame complexity and optimality for stochastic bandits. *arXiv preprint arXiv:1804.05929*, 2018.

[75] Yao Liu, Peng Ning, and Michael K Reiter. False data injection attacks against state estimation in electric power grids. *ACM Transactions on Information and System Security (TISSEC)*, 14(1):1–33, 2011.

[76] Michael Luca and Georgios Zervas. Fake it till you make it: Reputation, competition, and yelp review fraud. *Management Science*, 62(12):3412–3427, 2016.

[77] Xu Ma, Xiaoqian Sun, Yuduo Wu, Zheli Liu, Xiaofeng Chen, and Changyu Dong. Differentially private byzantine-robust federated learning. *IEEE Transactions on Parallel and Distributed Systems*, 33(12):3690–3701, 2022.

[78] Christopher Manning and Hinrich Schutze. *Foundations of statistical natural language processing.* MIT press, 1999.

[79] Dina Mayzlin, Yaniv Dover, and Judith Chevalier. Promotional reviews: An empirical investigation of online review manipulation. *American Economic Review*, 104(8):2421–55, 2014.

[80] Frank McSherry and Kunal Talwar. Mechanism design via differential privacy. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*, pages 94–103. IEEE, 2007.

[81] Volodymyr Mnih, Csaba Szepesvári, and Jean-Yves Audibert. Empirical bernstein stopping. In *Proceedings of the 25th international conference on Machine learning*, pages 672–679, 2008.

[82] Alexander McFarlane Mood. Introduction to the theory of statistics. 1950.

[83] Mc Farlane Mood. *Introduction to the theory of statistics.* McGraw-Hill, 1963.

[84] Thông T Nguyên, Xiaokui Xiao, Yin Yang, Siu Cheung Hui, Hyejin Shin, and Junbum Shin. Collecting and analyzing data from smart device users with local differential privacy. *arXiv preprint arXiv:1606.05053*, 2016.

[85] World Health Organization. *Air quality guidelines: global update 2005: particulate matter, ozone, nitrogen dioxide, and sulfur dioxide*. World Health Organization, 2006.

[86] The pick-up time in a day extracted from 2018 January New York Taxi data. Taxi dataset. `https://www.kaggle.com/code/wti200/exploratory-analysis-nyc-taxi-trip`, 2018.

[87] Saurav Prakash and Amir Salman Avestimehr. Mitigating byzantine attacks in federated learning. *arXiv preprint arXiv:2010.07541*, 2020.

[88] Zhan Qin, Yin Yang, Ting Yu, Issa Khalil, Xiaokui Xiao, and Kui Ren. Heavy hitter estimation over set-valued data with local differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 192–203, 2016.

[89] Ankit Singh Rawat, Priyank Anand, Hao Chen, and Pramod K Varshney. Collaborative spectrum sensing in the presence of byzantine attacks in cognitive radio networks. *IEEE Transactions on Signal Processing*, 59(2):774–786, 2010.

[90] Ankit Singh Rawat, Priyank Anand, Hao Chen, and Pramod K Varshney. Countering byzantine attacks in cognitive radio networks. In *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 3098–3101. IEEE, 2010.

[91] Wenbo Ren, Xingyu Zhou, Jia Liu, and Ness B Shroff. Multi-armed bandits with local differential privacy. *arXiv preprint arXiv:2007.03121*, 2020.

[92] Xuebin Ren, Chia-Mu Yu, Weiren Yu, Shusen Yang, Xinyu Yang, Julie A McCann, and S Yu Philip. Lopub: High-dimensional crowdsourced data publication with local differential privacy. *IEEE Transactions on Information Forensics and Security*, 13(9):2151–2166, 2018.

[93] Mohsen Rezvani, Aleksandar Ignjatovic, Elisa Bertino, and Sanjay Jha. Secure data aggregation technique for wireless sensor networks in the presence of collusion attacks. *IEEE transactions on Dependable and Secure Computing*, 12(1):98–110, 2014.

[94] John A Rice. *Mathematical statistics and data analysis*. Cengage Learning, 2006.

[95] Richard J Rossi. *Mathematical statistics: an introduction to likelihood based inference*. John Wiley & Sons, 2018.

[96] Dwijen Rudrapal, Smita Das, Nikhil Debbarma, and Swapan Debbarma. Internal attacker detection by analyzing user keystroke credential. *Lecture Notes on Software Engineering*, 1(1):49, 2013.

[97] Bill Schmarzo. *Big Data: Understanding how data powers big business*. John Wiley & Sons, 2013.

[98] Doug Semenick. Tests and measurements: The t-test. *Strength & Conditioning Journal*, 12(1):36–37, 1990.

[99] Hyejin Shin, Sungwook Kim, Junbum Shin, and Xiaokui Xiao. Privacy enhanced matrix factorization for recommendation with local differential privacy. *IEEE Transactions on Knowledge and Data Engineering*, 30(9):1770–1782, 2018.

[100] Shaorui Song, Lei Xu, and Liehuang Zhu. Efficient defenses against output poisoning attacks on local differential privacy. *IEEE Transactions on Information Forensics and Security*, 18:5506–5521, 2023.

[101] Harald Steck. Training and testing of recommender systems on data missing not at random. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 713–722, 2010.

[102] P Street, N Haven, and D Mayzlin. Promotional chat on the internet. 25 (2), 155-163, 2006.

[103] Haipei Sun, Xiaokui Xiao, Issa Khalil, Yin Yang, Zhan Qin, Hui Wendy Wang, and Ting Yu. Analyzing subgraph statistics from extended local views with decentralized differential privacy. In *CCS*, pages 703–717. ACM, 2019.

[104] Youming Tao, Yulian Wu, Peng Zhao, and Di Wang. Optimal rates of (locally) differentially private heavy-tailed multi-armed bandits. In *International Conference on Artificial Intelligence and Statistics*, pages 1546–1574. PMLR, 2022.

[105] ADP Team et al. Learning with privacy at scale. *Apple Mach. Learn. J*, 1(8):1–25, 2017.

[106] 2022 The number of coronavirus disease 2019 deaths for females in California as of December 14. Covid-19 dataset. `https://data.cdc.gov/NCHS/Provisional-COVID-19-Deaths-by-Sex-and-Age/9bhg-hcku/data`, 2022.

[107] Mar. 4 2020 The Wall Streat Journal. Startup biofourmis using ai-based system to battle coronavirus.

[108] Ana Isabel Torre-Bastida, Javier Del Ser, Ibai Laña, Maitena Ilardia, Miren Nekane Bilbao, and Sergio Campos-Cordobés. Big data for transportation and mobility: recent advances, trends and challenges. *IET Intelligent Transport Systems*, 12(8):742–755, 2018.

[109] Salil Vadhan. Learning with privacy at scale. In *Apple Machine Learning,Journal,vol. 1, no. 8*, pages 1–25. 2017.

[110] Ning Wang, Xiaokui Xiao, Yin Yang, Jun Zhao, Siu Cheung Hui, Hyejin Shin, Junbum Shin, and Ge Yu. Collecting and analyzing multidimensional data with local differential privacy. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*, pages 638–649. IEEE, 2019.

[111] Shaowei Wang, Liusheng Huang, Yiwen Nie, Xinyuan Zhang, Pengzhan Wang, Hongli Xu, and Wei Yang. Local differential private data aggregation for discrete distribution estimation. *IEEE Transactions on Parallel and Distributed Systems*, 30(9):2046–2059, 2019.

[112] Shaowei Wang, Liusheng Huang, Pengzhan Wang, Hou Deng, Hongli Xu, and Wei Yang. Private weighted histogram aggregation in crowdsourcing. In *International Conference on Wireless Algorithms, Systems, and Applications*, pages 250–261. Springer, 2016.

[113] Shaowei Wang, Xuandi Luo, Yuqiu Qian, Youwen Zhu, Kongyang Chen, Qi Chen, Bangzhou Xin, and Wei Yang. Shuffle differential private data aggregation for random

population. *IEEE Transactions on Parallel and Distributed Systems*, 34(5):1667–1681, 2023.

[114] Tianhao Wang, Jeremiah Blocki, Ninghui Li, and Somesh Jha. Locally differentially private protocols for frequency estimation. In *26th USENIX Security Symposium (USENIX Security 17)*, pages 729–745, 2017.

[115] Tianhao Wang, Ninghui Li, and Somesh Jha. Locally differentially private frequent itemset mining. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 127–143. IEEE, 2018.

[116] Tianhao Wang, Ninghui Li, and Somesh Jha. Locally differentially private heavy hitter identification. *IEEE Transactions on Dependable and Secure Computing*, 18(2):982–993, 2019.

[117] Stanley L Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60(309):63–69, 1965.

[118] Chuhan Wu, Fangzhao Wu, Yang Cao, Yongfeng Huang, and Xing Xie. Fedgnn: Federated graph neural network for privacy-preserving recommendation. *arXiv preprint arXiv:2102.04925*, 2021.

[119] Ruihan Wu, Jin Peng Zhou, Kilian Q Weinberger, and Chuan Guo. Does label differential privacy prevent label inference attacks? *arXiv preprint arXiv:2202.12968*, 2022.

[120] Yongji Wu, Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. Poisoning attacks to local differential privacy protocols for key-value data. *arXiv preprint arXiv:2111.11534*, 2021.

[121] GAO Xiao-xia. The control of indoor environment pollution from the decoration [j]. *Journal of Inner Mongolia Institute of Agriculture and Animal Husbandry*, 2, 2005.

[122] Chang Xu, Yu Jia, Liehuang Zhu, Chuan Zhang, Guoxie Jin, and Kashif Sharif. Tdfl: Truth discovery based byzantine robust federated learning. *IEEE Transactions on Parallel and Distributed Systems*, 33(12):4835–4848, 2022.

[123] Jianyu Yang, Xiang Cheng, Sen Su, Rui Chen, Qiyu Ren, and Yuhan Liu. Collecting preference rankings under local differential privacy. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*, pages 1598–1601. IEEE, 2019.

[124] Q. Ye, H. Hu, X. Meng, and H. Zheng. Privkv: Key-value data collection with local differential privacy. In *2019 IEEE Symposium on Security and Privacy (SP)*, 2019.

[125] Qingqing Ye, Haibo Hu, Man Ho Au, Xiaofeng Meng, and Xiaokui Xiao. Lf-gdpr: A framework for estimating graph metrics with local differential privacy. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–1, 2020.

[126] Qingqing Ye, Haibo Hu, Man Ho Au, Xiaofeng Meng, and Xiaokui Xiao. Towards locally differentially private generic graph metric estimation. In *ICDE*, pages 1922–1925. IEEE, 2020.

[127] Qingqing Ye, Haibo Hu, Kai Huang, Man Ho Au, and Qiao Xue. Stateful switch: Optimized time series release with local differential privacy. *arXiv preprint arXiv:2212.08792*, 2022.

[128] Qingqing Ye, Haibo Hu, Ninghui Li, Xiaofeng Meng, Huadi Zheng, and Haotian Yan. Beyond value perturbation: Local differential privacy in the temporal setting. In *INFOCOM*, pages 1–10. IEEE, 2021.

[129] Qingqing Ye, Haibo Hu, Xiaofeng Meng, and Huadi Zheng. PrivKV: Key-value data collection with local differential privacy. In *S&P*, pages 317–331. IEEE, 2019.

[130] Qingqing Ye, Haibo Hu, Xiaofeng Meng, Huadi Zheng, Kai Huang, Chengfang Fang, and Jie Shi. PrivKVM*: Revisiting key-value statistics estimation with local differential privacy. *IEEE Transactions on Dependable and Secure Computing*, 2021.

[131] Xun Yi, Russell Paulet, Elisa Bertino, Xun Yi, Russell Paulet, and Elisa Bertino. *Homomorphic encryption*. Springer, 2014.

[132] NIE Yiwen, Wei Yang, Liusheng Huang, Xike Xie, Zhenhua Zhao, and Shaowei Wang. A utility-optimized framework for personalized private histogram estimation. *IEEE Transactions on Knowledge and Data Engineering*, 31(4):655–669, 2018.

[133] Xiao-Jian Zhang and Xiao-Feng Meng. Differential privacy in data publication and analysis. *Chinese journal of computers*, 37(4):927–949, 2014.

[134] Zhikun Zhang, Tianhao Wang, Ninghui Li, Shibo He, and Jiming Chen. CALM: Consistent adaptive local marginal for marginal release under local differential privacy. In *CCS*, pages 212–229. ACM, 2018.

[135] Ziqi Zhang and Lei Luo. Hate speech detection: A solved problem? the challenging case of long tail on twitter. *Semantic Web*, 10(5):925–945, 2019.

[136] Huadi Zheng, Qingqing Ye, Haibo Hu, Chengfang Fang, and Jie Shi. BDPL: A boundary differentially private layer against machine learning model extraction attacks. In *ESORICS*, pages 66–83. Springer, 2019.

[137] Huadi Zheng, Qingqing Ye, Haibo Hu, Chengfang Fang, and Jie Shi. Protecting decision boundary of machine learning model with differentially private perturbation. *IEEE Transactions on Dependable and Secure Computing*, 2020.

[138] Youwen Zhu, Yiran Cao, Qiao Xue, Qihui Wu, and Yushu Zhang. Heavy hitter identification over large-domain set-valued data with local differential privacy. *IEEE Transactions on Information Forensics and Security*, 2023.