# Local Differential Privacy: Refined Mechanism Design and Utility Analysis

by

Ye Zheng

A proposal submitted in partial fulfillment of the
requirements for the degree of Doctor of Philosophy
in Computing and Information Sciences

B. Thomas Golisano College of Computing and Information Sciences
Rochester Institute of Technology

[Month and year of Dissertation Acceptance was signed]

GCCIS Ph.D. PROGRAM IN COMPUTING AND INFORMAITION SCIENCES
ROCHESTER INSTITUTE OF TECHNOLOGY
ROCHESTER, NEW YORK

CERTIFICATE OF APPROVAL

---

Ph.D. DEGREE PROPOSAL

---

The Ph.D. Degree Proposal of Ye Zheng
has been examined and approved by the
proposal committee as satisfactory for the
proposal required for the
Ph.D. degree in Computing and Information Sciences

| | |
|---|---|
| Ph.D. Program Director | Date |
| Yidan Hu, Proposal Advisor | Date |
| [External Chair's name], External Chair | Date |
| [Committee member's name] | Date |
| [Committee member's name] | Date |

# Local Differential Privacy: Refined Mechanism Design and Utility Analysis

by
Ye Zheng

Submitted to the
B. Thomas Golisano College of Computing and Information Sciences Ph.D. Program in Computing and Information Sciences
in partial fulfillment of the requirements for the
**Doctor of Philosophy Degree**
at the Rochester Institute of Technology

## Abstract

Local Differential Privacy (LDP) is a privacy model that enables users to perturb their data locally before sharing it with untrusted data collectors for analysis. This privacy model provides provable privacy guarantees for each individual user. Owing to these guarantees, LDP has been widely deployed by major technology companies, including Apple, Google, and Microsoft, to protect user privacy while still enabling data collection for analytics and machine learning. However, a fundamental challenge in LDP is the tradeoff between privacy and data utility: stronger privacy guarantees typically lead to lower utility for the data collector. To address this challenge, one central direction in theoretical LDP research is to design mechanisms that achieve better utility under the same privacy guarantee. This dissertation advances this direction by refining mechanism design and utility analysis under LDP.

This dissertation proceeds in four steps: (i) It examines whether correlated perturbation can improve the privacy–utility tradeoff beyond the standard assumption of independent per-user perturbations. (ii) Since optimizing this tradeoff is the central objective of mechanism design, it investigates optimal designs within a widely used class of LDP mechanisms for bounded numerical data. (iii) Motivated by trajectory data—a representative form of bounded numerical data that prior LDP studies often handle via discretization—it shows that operating directly in continuous space avoids privacy and efficiency issues. (iv) Moving beyond mechanism-level utility metrics toward downstream tasks, it studies how to theoretically quantify the utility of classifiers under LDP-perturbed inputs, taking a first step toward connecting LDP mechanisms with robustness.

## Acknowledgments

This is the acknowledgements text

*To the three cold, sober years I lived in Rochester,*
*whose quiet formed me and whose memory I shall always carry.*

# Contents

# List of Figures

# List of Tables

# Terminology

**Privacy and utility.** Privacy refers to protecting individuals' sensitive information from untrusted entities or adversaries. Utility refers to the usefulness or accuracy of the outputs obtained after applying privacy-preserving mechanisms.

**Privacy guarantee and privacy parameter.** These terms are used to describe the strength of privacy protection provided by a specific $\varepsilon$ value in Local Differential Privacy (LDP) mechanisms. Among them, "privacy parameter" specifically refers to the numerical value $\varepsilon$ that quantifies the privacy guarantee.

**Randomization and perturbation.** These terms are used interchangeably to describe the process of adding randomness or noise to data to protect privacy.

**Domain and space.** A domain is the set of admissible inputs for a function or mechanism (e.g. the input domain of an LDP mechanism). A space is a structured set of elements (e.g. continuous spaces, the distance space in TraCS (Chapter 6), etc).

**Original data, raw data, and sensitive data.** These terms refer to the data held by individuals before any privacy-preserving mechanism is applied. Raw data is an umbrella term for unprocessed data, whereas sensitive data specifically denotes information that is private or confidential.

**Perturbed data, randomized data, and reported data.** These terms refer to data that has been modified by adding randomness or noise through privacy-preserving mechanisms. They are used interchangeably for consistency with the context in which they appear in.

**User.** A user is an individual or entity that holds data and applies an LDP mechanism to protect privacy before sharing data with a server or data collector.

**Data collector and server.** A data collector (or server) is an entity that collects data from users, typically for analysis or aggregation. The term "server" is broader and may refer to entities beyond data collection.

**Adversary model.** An adversary is an entity that attempts to compromise users' privacy by analyzing collected data or exploiting system vulnerabilities. In the context of LDP, the server and other users may be potential adversaries. An adversary model specifies the adversary's capabilities and knowledge in a given scenario (i.e. what information they can access and what actions they can take).

# Notations

General notations and their meanings.

| **Notation** | **Meaning** |
|---|---|
| $\mathcal{M}_\varepsilon(x)$ | LDP mechanism with privacy parameter $\varepsilon$ applied to input $x$ |
| $y$ | Output of an LDP mechanism, i.e. $y \leftarrow \mathcal{M}_\varepsilon(x)$ |
| $\mathcal{X}, \mathcal{Y}$ | Input and output domains of an LDP mechanism |
| $\mathbb{R}$ | Real number domain |
| $n$ | Number of users in the system |
| $\mathrm{pdf}[\cdot]$ | Probability density function of a random variable |
| $\mathrm{MSE}[\cdot]$ | Mean Squared Error of a random variable |

Notations in Chapter 4 (JRR).

| **Notation** | **Meaning** |
|---|---|
| $n_x$, $\hat{n}_x$ | Number of users with original value $x$, and its estimator |
| $u_i$ | User $i$, where $i \in \{1, \dots, n\}$ |
| $I_x$ | The number of $x$ in the reported values from all users |
| $p$ and $q \coloneqq 1 - p$ | Probabilities in RR, i.e. reporting truthfully with probability $p$ |
| $T_{2i}$, $T_{2i-1}$ | Reporting truthfulness indicators of users $2i$ and $2i-1$ in JRR |
| $\rho$ | Correlation coefficient between two users in a group in JRR |
| $\mathrm{Var}[\cdot]$ | Variance of a random variable |
| $\mathrm{Cov}[\cdot, \cdot]$ | Covariance of two random variables |
| $\mathcal{C}$ | Set of colluding users with size $m = \lvert\mathcal{C}\rvert$ |

Notations in Chapter 5 (OGPM).

| **Notation** | **Meaning** |
|---|---|
| $\mathcal{L}(y, x)$ | A loss metric measuring the "distance" between $y$ and $x$ |
| $Err(x)$ | Expected error (of $y$) w.r.t. ground truth $x$ |
| $\mathcal{P}_{\mathcal{M}(x)}$ | Probability density function of $y = \mathcal{M}(x)$ |
| $p_\varepsilon$ | Sampling probability w.r.t. $\varepsilon$ |
| $[l_{x,\varepsilon}, r_{x,\varepsilon})$ | Sampling interval w.r.t. $x$ and $\varepsilon$ |
| $[l^{\text{mod}}_{x,\varepsilon}, r^{\text{mod}}_{x,\varepsilon})$ | Sampling interval in the circular domain $[0, 2\pi)$ |

Notations in Chapter 6 (TraCS).

| **Notation** | **Meaning** |
|---|---|
| $\mathcal{T}$ | Sensitive trajectory, $\mathcal{T} \coloneqq \{\tau_1, \ldots, \tau_n\}$ |
| $\mathcal{T}'$ | Perturbed trajectory, $\mathcal{T}' \coloneqq \{\tau_1', \ldots, \tau_n'\}$ |
| $\mathcal{S}$ | Rectangular continuous location space, $\mathcal{S} \coloneqq [a_{\text{sta}}, a_{\text{end}}] \times [b_{\text{sta}}, b_{\text{end}}]$ |
| $\varphi \in \mathcal{D}_\varphi \coloneqq [0, 2\pi)$ | Direction & direction space |
| $\varphi'$ | Perturbed direction |
| $r(\varphi) \in \mathcal{D}_{r(\varphi)}$ | Distance & distance space along $\varphi$ |
| $r'(\varphi)$ | Perturbed distance along $\varphi$ |
| $\overline{r}(\varphi) \in [0, 1)$ | Normalized distance over $\mathcal{D}_{r(\varphi)}$ |
| $\mathcal{M}_\circ(\varphi)$ | Direction perturbation mechanism applied on direction $\varphi$ |
| $\mathcal{M}_-(r)$ | Distance perturbation mechanism applied on distance $r$ |

Notations in Chapter 7.

| **Notation** | **Meaning** |
|---|---|
| $h$ | Classifier function; $h : \mathbb{R}^d \to \{1, \ldots, K\}$ is called a $d$-dimensional classifier |
| $B_\theta(x)$ | $\theta$-ball centered at $x$ |
| $\tilde{x}$ | Perturbed version of $x$ |
| $\rho(\varepsilon, \theta)$ | Utility quantification for a classifier (w.r.t. $\varepsilon$ and $\theta$) |
| $F_{\mathcal{M}}(\cdot)$ | Cumulative distribution function of LDP mechanism $\mathcal{M}$ |
| $\omega$ | Confidence parameter; $1 - \omega$ indicates the confidence level |
| $\tau$ | Robustness tolerance |

# Chapter 1

# Introduction

Local differential privacy (LDP) mechanisms protect individual users' data privacy against untrusted data collectors by allowing each user to locally perturb their data before sharing it [42, 48, 81]. Though the data collector receives only perturbed data, they can still learn valuable statistics while being unable to infer much about any individual user's original data that would compromise privacy, with privacy guarantees quantified by the privacy parameter $\varepsilon$. Due to these provable privacy guarantees, LDP mechanisms have been widely adopted by major technology companies, including Apple's operating systems [132], Google Chrome [50], and Microsoft Office [137] for collecting user statistics on-device. Furthermore, LDP is a key privacy-enhancing component in federated learning, a decentralized machine learning paradigm where users collaboratively train a global model without sharing sensitive data with a central server [4, 85].

Figure 1.1 illustrates the LDP system model, in which an untrusted data collector may attempt to infer users' true (sensitive) data. The LDP mechanism $\mathcal{M}$ acts as a guard at the trust boundary,

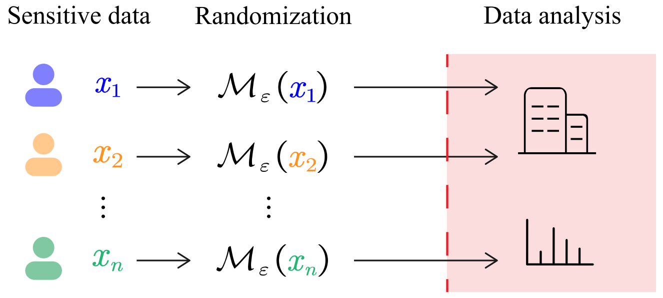

Sensitive data
Randomization
Data analysis
$x_1$
$\mathcal{M}_\varepsilon(x_1)$
$x_2$
$\mathcal{M}_\varepsilon(x_2)$
$x_n$
$\mathcal{M}_\varepsilon(x_n)$

Figure 1.1: LDP system model. The dashed red line indicates the trust boundary. Each user locally perturbs their original data $x_i$ using an LDP mechanism $\mathcal{M}$ before sending it to the untrusted data collector. The data collector has access only to the perturbed data $y_i = \mathcal{M}_\varepsilon(x_i)$, which it uses to perform statistical analysis.

Table 1.1: Comparison of typical privacy-enhancing techniques (PETs).

| **Threat Model** | **Technique** | **Privacy Guarantee** | **Complexity** | **Data Utility** |
|---|---|---|---|---|
| Trusted Collector | $k$-anonymity [143] | Syntactic | Medium | High |
| | Central DP [47] | Semantic ($\varepsilon$-DP) | Simple | $\varepsilon$-dependent |
| Untrusted Collector[a] | HE[b] [133] | Semantic (IND-CPA) | Complex | High |
| | MPC[c] [173] | Semantic (Real/Ideal) | Complex | High |
| | LDP [42] | Semantic ($\varepsilon$-LDP) | Simple | $\varepsilon$-dependent |

[a] In practice, the untrusted data collector may need to be *honest-but-curious* (HBC) to ensure correct *external* service functionality. Note that this assumption is not always required for the *internal* privacy guarantees.

[b] Homomorphic Encryption (HE) typically requires that the service functionality be expressible using operations supported by the encryption scheme.

[c] Secure Multi-Party Computation (MPC) usually assumes a threshold of dishonest parties to guarantee privacy.

preserving privacy by injecting *unreversible* randomness (quantified by $\varepsilon$) into each user's original data $x_i$ before it leaves their device.

**Advantages over other privacy-enhancing techniques.** Orthogonal to LDP, various privacy-enhancing techniques (PETs) have been proposed to protect user data privacy in data collection and analysis. They are designed under different threat models and offer different senses of privacy guarantee. Table 1.1 summarizes and compares typical PETs.

$k$-anonymity [143] and central DP [47] assume a trusted data collector with direct access to users' original data, and place the responsibility for safeguarding individual privacy against external inference attacks on the collector. Among them, (i) $k$-anonymity is a *syntactic* privacy model that protects the linkage between users' identities and their data by masking quasi-identifiers in the dataset, rather than directly protecting the sensitive data themselves. Designing an effective masking scheme is often difficult for high-dimensional data, and such syntactic models have been shown to be vulnerable to various attacks [83, 89, 118]. (ii) Central DP, in contrast, provides a *semantic* privacy guarantee by adding randomness to aggregated statistics before releasing them, making it difficult to infer any individual user's data from the published statistics. A subtle but important issue in central DP is the definition of "neighboring datasets" (e.g. record removal vs record replacement), which can lead to different required noise levels under the same privacy parameter $\varepsilon$ [119, 125, 127].

Homomorphic Encryption (HE) [133] and Secure Multi-Party Computation (MPC) [173] are cryptographic techniques that can also protect user data privacy when the data collector is untrusted. (i) HE enables certain function evaluations to be performed directly on encrypted data without decryption. Its privacy guarantee relies on indistinguishability under chosen-plaintext attack (IND-

CPA), which ensures that an adversary cannot distinguish the encryptions of any two chosen plaintexts. (ii) MPC allows multiple parties to jointly compute a function over their private inputs without revealing those inputs to one another, typically by operating on secret shares. Its privacy guarantee is formalized via the real/ideal simulation paradigm, which requires that an adversary's (e.g. a malicious party's) view during protocol execution can be simulated using only its own input and output, implying that the protocol leaks no additional information beyond what is inherently revealed by the prescribed outputs. Both HE and MPC typically require sophisticated protocol design and often rely on honest assumptions about the participating parties in practice.

Compared with these PETs, LDP offers several advantages. *It provides semantic privacy guarantees for individual users' data, quantified by the clean $\varepsilon$-LDP notion, agnostic to the data collector and has low computational complexity, making it suitable for resource-constrained devices.*

**Privacy–utility tradeoff in LDP.** Despite these advantages, LDP mechanisms are fundamentally constrained by a privacy–utility tradeoff: achieving stronger privacy guarantees (i.e. smaller $\varepsilon$) requires injecting more randomness into users' data, which in turn degrades the utility of the collected data for downstream statistical analysis. Figure 1.2 illustrates this tradeoff by showing the worst-case expected error of $\mathcal{M}_\varepsilon(x)$ for four LDP mechanisms on a numerical data domain $x \in [0, 1]$. As $\varepsilon$ increases, the worst-case expected error decreases for all mechanisms, indicating improved data utility. However, for a fixed privacy parameter $\varepsilon$, the utility (error) achieved by different mechanisms can vary significantly. Some mechanisms, such as OGPM, achieve lower expected error across most $\varepsilon$ values than others, demonstrating a better privacy–utility tradeoff. One central direction of theoretical LDP research is therefore to design mechanisms that optimize this tradeoff, as they directly determine the building blocks of practical LDP systems deployed in real-world applications. Beyond this, it is also important to quantify the utility of complex data analysis tasks under LDP, such as classifier performance. Such utility cannot be inferred solely from the expected error of the underlying LDP mechanism, since many classifiers can tolerate moderate perturbations and may degrade non-linearly as noise increases. A clear understanding of these implications is essential for selecting and configuring LDP mechanisms in practice.

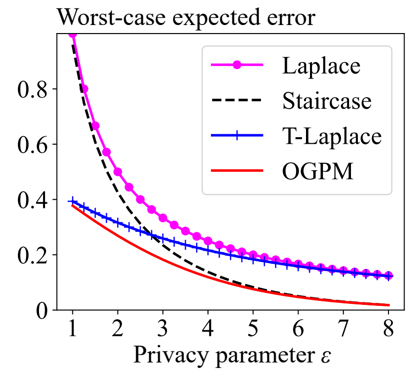


Figure 1.2: Privacy–utility curves.

To summarize, there are two fundamental challenges in advancing current LDP research:

- Designing LDP mechanisms with optimized privacy–utility tradeoffs.
- Quantifying the utility of complex data analysis tasks under LDP.

To address these fundamental challenges, this dissertation proceeds in four steps. (i) It examines whether correlated perturbation can improve the privacy–utility tradeoff beyond the standard assumption of independent per-user perturbations. (ii) Since optimizing this tradeoff is the central objective of mechanism design, it investigates optimal designs within a widely used class of LDP mechanisms for bounded numerical data. (iii) Motivated by trajectory data—a representative form of bounded numerical data that prior LDP studies often handle via discretization—it shows that operating directly in continuous space avoids privacy and efficiency issues. (iv) Moving beyond mechanism-level utility metrics toward downstream tasks, it studies how to theoretically quantify the utility of classifiers under LDP-perturbed inputs.

**Contributions of this dissertation.** In summary, to advance mechanism design and utility analysis in LDP research, this dissertation makes the following contributions:

- Chapter 4 introduces *correlated perturbation* into LDP mechanisms for multiple users' data, which generalizes existing LDP mechanisms that perturb each user's data independently and achieves improved privacy–utility tradeoffs.
- Chapter 5 establishes the *optimality* of piecewise-based mechanisms, state-of-the-art category of LDP mechanisms for collecting bounded numerical data.
- Chapter 6 proposes two mechanisms for collecting individual trajectory data, which achieve higher efficiency and data utility by operating in *continuous space* instead of previously studied discrete space.
- Chapter 7 provides a *quantification framework* for theoretically analyzing data utility of classifiers under LDP-perturbed inputs, making a first step towards connecting LDP mechanisms with robustness.

**Open source resources.** All implementations and evaluations are available at `https://github.com/ZhengYeah/`.

**Dissertation structure.** The remainder of this dissertation presents the necessary background on LDP in Chapter 2, briefly overviews the research questions and key ideas underlying each contribution in Chapter 3, followed by four chapters that each address one of the research questions in detail. Each chapter begins by formulating the problem and outlining the key ideas, followed by the proposed methods, discussion, and experimental evaluation. Related work for each research question is reviewed at the end of the corresponding chapter.

# Chapter 2

# Background

This chapter presents necessary background on LDP, including its formal definition, common mechanisms, and properties used in this dissertation.

## 2.1 Local Differential Privacy

Local Differential Privacy (LDP) [42, 81] is a formal privacy definition providing provable privacy guarantees for data collection in scenarios where users do not trust the data collector. Such privacy guarantees are defined in terms of the *indistinguishability* of outputs from a randomized algorithm when given different inputs.

**Definition 1** ($\varepsilon$-LDP)**.** *A randomized algorithm $\mathcal{M} : \mathcal{X} \to \mathcal{Y}$ satisfies $\varepsilon$-LDP if, for any two inputs $x_1, x_2 \in \mathcal{X}$ and any measurable output event $\mathcal{Y}_{\text{sub}} \subseteq \mathcal{Y}$:*

$$\forall x_1, x_2 \in \mathcal{X}, \forall \mathcal{Y}_{\text{sub}} \subseteq \mathcal{Y}, \quad \Pr[\mathcal{M}(x_1) \in \mathcal{Y}_{\text{sub}}] \leq e^{\varepsilon} \cdot \Pr[\mathcal{M}(x_2) \in \mathcal{Y}_{\text{sub}}].$$

Intuitively, Definition 5 requires that for any two inputs $x_1$ and $x_2$, the probability of the mechanism $\mathcal{M}$ producing any output in $\mathcal{Y}_{\text{sub}}$ should be similar (thus indistinguishable) up to a multiplicative factor of $e^{\varepsilon}$.

For guidance of mechanism design and cleanness of proof, Definition 5 is often equivalently stated in terms of individual output $y \in \mathcal{Y}$ in discrete or continuous output domains.

**Definition 2** (Equivalent definitions of $\varepsilon$-LDP)**.** *A randomized algorithm $\mathcal{M} : \mathcal{X} \to \mathcal{Y}$ with discrete* *output domain $\mathcal{Y}$ satisfies $\varepsilon$-LDP if,* [*]

$$\forall x_1, x_2 \in \mathcal{X}, \forall y \in \mathcal{Y}, \quad \frac{\Pr[\mathcal{M}(x_1) = y]}{\Pr[\mathcal{M}(x_2) = y]} \leq e^{\varepsilon}.$$

*Similarly, a randomized algorithm $\mathcal{M} : \mathcal{X} \to \mathcal{Y}$ with continuous output domain $\mathcal{Y}$ satisfies $\varepsilon$-LDP if,*

$$\forall x_1, x_2 \in \mathcal{X}, \forall y \in \mathcal{Y}, \quad \frac{\mathrm{pdf}[\mathcal{M}(x_1) = y]}{\mathrm{pdf}[\mathcal{M}(x_2) = y]} \leq e^{\varepsilon}.$$

**Proposition 1.** *Definition 5 is equivalent to the discrete-domain and continuous-domain definitions in Definition 2.*

*Proof.* (Sketch) Enumerate over all possible output events $\mathcal{Y}_{\text{sub}}$ (discrete case) or integrate over all possible output events $\mathcal{Y}_{\text{sub}}$ (continuous case). Refer to Appendix A.1.1 for full proof. □

Under Definition 2, we refer to $\mathcal{M}$ as a *discrete-domain* LDP mechanism when its input domain $\mathcal{X}$ is discrete, and as a *continuous-domain* LDP mechanism when $\mathcal{X}$ is continuous. Unless stated otherwise, we use discrete-domain mechanisms for exposition and proofs; the corresponding results typically extend to the continuous case by replacing probabilities with probability density functions.

**Interpretations of LDP.** The privacy guarantee of $\varepsilon$-LDP can be interpreted in multiple ways. (i) In the definition view, the output distribution is somewhat independent of the true input. (ii) In a Bayesian-adversarial view, an adversary observing output $y$ and trying to distinguish between two possible inputs $x_1$ and $x_2$, has limited likelihood of success, i.e. $\Pr[y|x_1]/\Pr[y|x_2] \leq e^{\varepsilon}$. (iii) In an information-theoretic view, the mutual information between the input and output is bounded by $\mathcal{O}(\varepsilon)$ [32]. All these interpretations point to the same conclusion: the output reveals limited information about the input, despite the mechanism and the privacy parameter $\varepsilon$ being known to the adversary.

**Choice of $\varepsilon$.** The privacy parameter $\varepsilon$ controls the strength of the privacy guarantee provided by an LDP mechanism. A smaller value of $\varepsilon$ indicates a stronger privacy guarantee, i.e. smaller distinguishability between outputs corresponding to different inputs. Conversely, a larger value of $\varepsilon$ indicates a weaker privacy guarantee, but preserves more information about the original data. Generally, we consider $\varepsilon \geq 0$:[†]

- When $\varepsilon = 0$, it means $\Pr[\mathcal{M}(x_1) = y] = \Pr[\mathcal{M}(x_2) = y]$ for any inputs $x_1, x_2$ and output $y$, which implies that the mechanism's output distribution is identical for all possible inputs, providing perfect privacy.

[*]This formula adopts conventions $0/0 \coloneqq 0$ and $c/0 \coloneqq \infty$ for any $c > 0$.

[†]$\varepsilon < 0$ makes the definition trivial; see Appendix A.2.1.

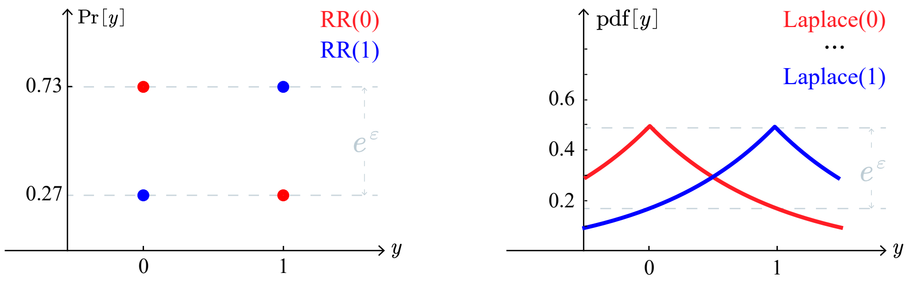


Figure 2.1: The RR mechanism (left) on binary inputs $x \in \{0, 1\}$ when $\varepsilon = 1$ and the Laplace mechanism (right) on real-valued inputs $x \in [0, 1]$ when $\varepsilon = 1$. Note that the Laplace mechanism's output domain is continuous and larger than the input domain. In practice, its outputs are often post-processed (e.g. truncation or discretization) to fit the application needs while preserving LDP guarantees, as discussed in Theorem 1.

- When $\varepsilon \to \infty$, it allows $\Pr[\mathcal{M}(x_1) = y]$ or $\Pr[\mathcal{M}(x_2) = y]$ approaches to zero or one, which means the mechanism can reveal the input deterministically without any privacy protection.

Note that although $\varepsilon$ provides a unified measure of privacy across different LDP mechanisms and data domains, the *effective* privacy guarantee may vary depending on the domain size. For example, $\varepsilon = 1$ leaks noticeable information when the input domain size is 2, while it provides strong uncertainty when the input domain size is $10^2$.[‡] In literature and practice, $\varepsilon$ is often chosen between 0.01 and 10, but the exact value depends on the specific application.

**Typical LDP mechanisms.** The most classical discrete-domain LDP mechanism is the Randomized Response (RR) mechanism [159] for Yes/No questions. Given a binary input $x \in \{0, 1\}$, RR outputs the true value $x$ with probability $p = e^{\varepsilon}/(e^{\varepsilon}+1)$, and outputs the flipped value $1-x$ with probability $1 - p = 1/(e^{\varepsilon} + 1)$. RR satisfies $\varepsilon$-LDP because

$$\forall x_1, x_2 \in \{0, 1\}, \forall y \in \{0, 1\}, \quad \frac{\Pr[\mathcal{M}_{\mathrm{RR}}(x_1) = y]}{\Pr[\mathcal{M}_{\mathrm{RR}}(x_2) = y]} \leq \frac{e^{\varepsilon}}{1} = e^{\varepsilon}.$$

The most classical continuous-domain LDP mechanism is the Laplace mechanism [47]. It can be applied to real-valued inputs $x \in [0, 1]$[§] by adding Laplace noise with scale $1/\varepsilon$, i.e. outputting $x + \eta$, where $\eta \sim \mathrm{Lap}(0, 1/\varepsilon)$. The Laplace mechanism satisfies $\varepsilon$-LDP because

$$\forall x_1, x_2 \in [0, 1], \forall y \in \mathbb{R}, \quad \frac{\mathrm{pdf}[\mathcal{M}_{\mathrm{Lap}}(x_1) = y]}{\mathrm{pdf}[\mathcal{M}_{\mathrm{Lap}}(x_2) = y]} \leq e^{\varepsilon|x_1 - x_2|} \leq e^{\varepsilon}.$$

[‡]This is because LDP enforce pairwise indistinguishability, but does not accumulate uncertainty across many alternatives. From the information-theoretic view, the privacy guarantee is domain-dependent, and a more precise bound is $\mathcal{I}(x; \mathcal{M}(x)) \leq \min\{\log|\mathcal{X}|, \mathcal{O}(\varepsilon)\}$. See Appendix A.2.2 for details.

[§]The Laplace mechanism can also be applied to discrete domain $x \in \{0, 1\}$ by treating it as a continuous domain, and then post-processing the output to discrete values.

## 2.2 Properties of LDP Mechanisms

LDP mechanism as a privacy primitive has several important properties that facilitate complex mechanism design and privacy analysis.

**Theorem 1** (Post-processing)**.** *If a randomized algorithm $\mathcal{M} : \mathcal{X} \to \mathcal{Y}$ satisfies $\varepsilon$-LDP, then for any (possibly randomized) function $f : \mathcal{Y} \to \mathcal{Y}'$, the composed algorithm $f \circ \mathcal{M} : \mathcal{X} \to \mathcal{Y}'$ also satisfies $\varepsilon$-LDP.*

*Proof.* (Sketch) If $f$ is deterministic, the proof is straightforward by substituting the output $y' \in \mathcal{Y}'$ with its pre-image $f^{-1}(y')$ in Definition 5. If $f$ is randomized, we can condition on $f(y)$ and substitute $y$ term by term to reach the at-least $\varepsilon$-LDP guarantee. Refer to Appendix A.1.2 for full proof. □

The intuition behind Theorem 1 is that post-processing $f$ can only discard or remap information in $y = \mathcal{M}(x)$;[¶] it can't create new dependence on $x$. So distinguishability between $x_1$ and $x_2$ can't increase. Via Theorem 1, an LDP mechanism's output can be freely transformed, truncated, or discretized to fit application needs without sacrificing its privacy guarantee.

**Theorem 2** (Sequential (adaptive) composition)**.** *Let $\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_k$ be randomized mechanisms applied sequentially to an input $x \in \mathcal{X}$.*

- *$\mathcal{M}_1(x)$ is $\varepsilon_1$-LDP.*
- *For each $i \geq 2$, $\mathcal{M}_i(x, y_1, \ldots, y_{i-1})$ may depend on all previous outputs, and for every fixed history $(y_1, \ldots, y_{i-1})$, the mechanism $\mathcal{M}_i(x, y_1, \ldots, y_{i-1})$ is $\varepsilon_i$-LDP.*

*Define the composed mechanism*

$$\mathcal{M}(x) := (\mathcal{Y}_1, \mathcal{Y}_2, \ldots, \mathcal{Y}_k), \quad \mathcal{Y}_i = \mathcal{M}_i(x, Y_1, \ldots, Y_{i-1}),$$

*Then $\mathcal{M}$ satisfies $(\varepsilon_1 + \varepsilon_2 + \cdots + \varepsilon_k)$-LDP.*

*Proof.* (Sketch) Use the law of total probability to expand the joint output probability, then apply the LDP inequalities for each mechanism $\mathcal{M}_i$ one by one. Refer to Appendix A.1.3 for full proof. □

[¶]On the other hand, if $f$ uses information about $x$ directly, it is not a post-processing, and may violate LDP.

The intuition behind Theorem 2 is that each mechanism $\mathcal{M}_i$ contributes $\varepsilon_i$ distinguishability between any two inputs, and the total distinguishability accumulates additively. Via Theorem 2, multiple one-dimensional LDP mechanisms can be composed to build high-dimensional mechanisms for complex data structures, particularly when each mechanism accesses different dimensions or aspects of the data.

**Example 1.** *(Build a 2D LDP mechanism from 1D mechanisms) Consider a 2D input $x = (x^{(1)}, x^{(2)}) \in \mathcal{X}_1 \times \mathcal{X}_2$, and two 1D LDP mechanisms $\mathcal{M}_1 : \mathcal{X}_1 \to \mathcal{Y}_1$ and $\mathcal{M}_2 : \mathcal{X}_2 \to \mathcal{Y}_2$, where $\mathcal{M}_1$ is $\varepsilon_1$-LDP and $\mathcal{M}_2$ is $\varepsilon_2$-LDP. Here is a special case that $\mathcal{M}_1$ and $\mathcal{M}_2$ are independent of each other, i.e. output $y_1$ of $\mathcal{M}_1(x^{(1)})$ does not affect the operation of $\mathcal{M}_2(x^{(2)})$, thus Theorem 2 applicable. We can design a 2D LDP mechanism $\mathcal{M} : \mathcal{X}_1 \times \mathcal{X}_2 \to \mathcal{Y}_1 \times \mathcal{Y}_2$ as*

$$\mathcal{M}(x) := \left(\mathcal{M}_1(x^{(1)}), \mathcal{M}_2(x^{(2)})\right).$$

*By Theorem 2, $\mathcal{M}$ satisfies $(\varepsilon_1 + \varepsilon_2)$-LDP.*

# Chapter 3

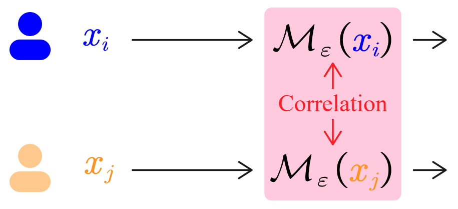
# Research Questions and Technical Overview

This chapter overviews the research questions and key technical ideas underlying each contribution of this dissertation.

## 3.1 Correlated Perturbation for LDP

The most classical and widely used application of LDP is frequency estimation, where the data collector aims to estimate the number (or proportion) of users possessing a certain attribute or data value. Randomized Response (RR) [159] is the first known and most classical LDP protocol for frequency estimation on binary data (e.g. yes/no questions). In RR, each user perturbs their true binary data independently by flipping it with a probability determined by the privacy parameter $\varepsilon$. Due to its simplicity and effectiveness, RR has been widely adopted as a building block in many LDP mechanisms for diverse data types and analysis tasks [12, 29, 154, 155]. A common feature of these RR-based mechanisms, and of LDP mechanisms more broadly, is that each user's data is perturbed independently, resulting in a large amount of total randomness. This raises a natural research question: *Can the data utility of RR be improved by introducing correlations among the perturbations performed by different users?*

Chapter 4 investigates correlated perturbations for frequency estimation to improve data utility without weakening LDP guarantees. The key insight is that the total randomness injected into users' data can be reduced by partitioning data users into disjoint groups and introducing carefully designed correlations into each group's random perturbations, as illustrated in Figure 3.1. A novel Joint Randomized Response (JRR) mechanism is proposed based on this idea. With appropriately chosen parameters, JRR achieves substantially higher data utility in the vast majority of cases, while providing the same level of LDP protection as classical RR.



$x_i$
$\mathcal{M}_\varepsilon(x_i)$
Correlation
$x_j$
$\mathcal{M}_\varepsilon(x_j)$

Figure 3.1: Correlated perturbation.

Specifically, Chapter 4 makes the following contributions:

- *Correlated perturbation.* It makes the first attempt in the LDP literature to introduce correlated perturbations into LDP mechanisms, thereby improving the data utility of frequency estimation.
- *The JRR mechanism.* It proposes a general JRR mechanism that provides the same level of LDP protection as classical RR, while substantially improving data utility in most cases, particularly when the number of data users is large.
- *Deployable instantiations.* It presents a deployable instantiation of JRR that leverages MPC protocols to conceal group membership and enable secure correlated perturbations.

## 3.2 Optimal Piecewise-based Mechanism

Moving beyond last section's binary data, this section focuses on numerical data. Numerical data with bounded domains is a fundamental data type in personal devices and sensor networks. These bounded domains can be categorized into two types: linear ranges, such as sensor readings in $[0, 1)$, referred to as the *classical domain*;[*] and cyclic ranges, such as angular measurements in $[0, 2\pi)$, referred to as the *circular domain*. The Laplace mechanism [47] is the most classical LDP mechanism for numerical data privacy: it adds random noise drawn from a Laplace distribution determined by $\varepsilon$ to the sensitive data. However, the unbounded support (i.e. the entire real line) of the Laplace noise makes it unsuitable for bounded domains.

State-of-the-art LDP mechanisms for numerical data on bounded domains are *piecewise-based mechanisms* [95, 101, 150]. These mechanisms randomize sensitive data to values drawn from carefully designed piecewise probability distributions. Existing instantiations use different pieces and probabilities, but are all designed for classical (linear) domains. Their applicability to other

[*]To ease interval operations (e.g. union and intersection), this dissertation uses left-closed right-open intervals, e.g. $[0, 1)$ instead of $[0, 1]$. They are equivalent to closed intervals in implementation and practical applications.

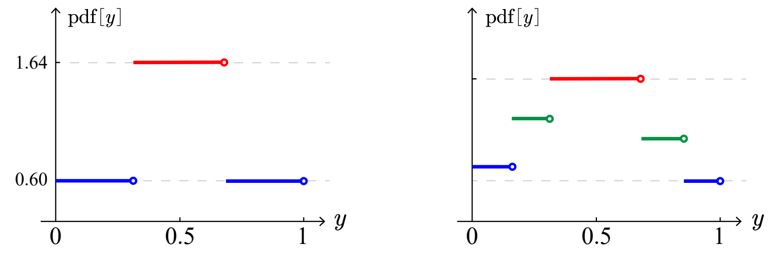


Figure 3.2: Illustration of piecewise distributions. The output $y = \mathcal{M}_\varepsilon(0.5)$ is drawn from a piecewise distribution, which ensures $\varepsilon$-LDP. Existing piecewise-based mechanisms [95, 101, 150] are special cases of 3-piece distributions (left). In contrast, a general piecewise distribution can have an arbitrary number of pieces and locations (right), which has potential to improve data utility as existing evidence [60, 148] suggests.

bounded domains, such as the circular domains of angular sensors that frequently arise in personal devices, remains unexplored.

The optimality of piecewise-based mechanisms remains an open problem. Existing instantiations can be viewed as heuristic forms of the 3-piece mechanism (TPM). As a special case with exactly 3 pieces and pre-defined functional forms, TPM is too restrictive to fully characterize the optimality of piecewise-based mechanisms. Evidence from the staircase Laplace mechanism [60] for unbounded numerical data shows that the asymptotically optimal mechanism has a staircase (multi-piece) structure. For categorical data, the Staircase Randomized Response mechanism (SRR) [148] improves data utility in location collection compared to classical RR. These and other results suggest that increasing the diversity of probabilities over the data domain, i.e. using more pieces, can improve data utility. Motivated by this, a fundamental question for piecewise-based mechanisms is: *What is the optimal instantiation of a piecewise-based mechanism?* In designing such mechanisms, the number of pieces, as well as their probabilities and sizes, can be arbitrary. Finding the optimal instantiation within this large design space is challenging, as it requires jointly optimizing the number of pieces and the associated probabilities and sizes.

Chapter 5 studies the optimality of piecewise-based mechanisms in their most general form. It extends TPM to a generalized piecewise-based mechanism (GPM) with $m$ pieces, where each piece has no predefined functional form. Within this GPM framework, it formulates an optimization problem that minimizes the distance between the sensitive and randomized data. By combining numerical solutions of this optimization problem with analytical proofs, it derives a closed-form characterization of the optimal GPM for classical domains. For circular domains, where the distance metric is periodic (e.g. the distance between 0 and $2\pi$ is zero), it explicitly incorporates this property into the mechanism design and reduce the search for the optimal mechanism to related problems on classical domains.

Specifically, Chapter 5 makes the following contributions:

- *Solving framework.* It is the first work to study the closed-form optimal piecewise-based mechanism in its most general form. A framework is proposed that combines analytical proofs with off-the-shelf optimization solvers to derive the closed-form optimal mechanism, providing a practical foundation for achieving optimal data utility under LDP for bounded numerical data.
- *Closed-form instantiations.* It derives closed-form optimal mechanisms for both the classical and circular domains. These mechanisms can be directly used as building blocks in applications such as sensor networks and federated learning.

## 3.3 Trajectory Collection in Continuous Space under LDP

A representative form of bounded numerical data in last section is trajectory data. Trajectory data from users—sequences of locations that describe movement over time—are a fundamental resource for activity analysis and location-based services, such as activity classification and routine detection. Existing LDP methods for trajectory collection are primarily designed for discrete spaces. They rely on discrete-domain mechanisms, such as the Exponential mechanism [110], to perturb trajectory data. Because discrete LDP mechanisms are explicitly defined with respect to the size of the location space, these methods either partition the continuous space into grids [148] or assume that the location space is a finite set of labeled locations (points of interest) [34, 181].

**Limitations of discrete-domain mechanisms.** (i) Their privacy guarantees are inherently tied to the discrete set. For example, a space with 10 locations offers weaker effective privacy than one with 100 locations: even a trivial inference strategy that always outputs a fixed location succeeds with probability at least 1/10 in the former, regardless of the privacy parameter $\varepsilon$.[†] (ii) Their efficacy and efficiency are often limited by the domain size. As the number of candidate locations grows, the probability of a mechanism outputting the true location diminishes. Furthermore, the widely used Exponential mechanism incurs linear sampling complexity in the domain size, making the generation of each perturbed location computationally expensive.[‡] (iii) Discrete methods are not directly applicable to inherently continuous location spaces, such as those arising from flying and sailing trajectories or sensor data from wearable devices. Although one can discretize a continuous space before applying discrete methods, this inherits the aforementioned limitations. Moreover, choosing

[†]Appendix D.2.1 provides details on the space-dependence of indistinguishability.

[‡]Appendix D.2.2 details the Exponential mechanism's efficiency and data utility (efficacy) limitations.

a suitable discretization granularity that balances privacy, utility, and efficiency is non-trivial.[§]

Chapter 6 addresses these limitations by *shifting the focus from discrete to continuous spaces for trajectory collection under LDP*. A continuous space represents locations as real-valued points, such as GPS coordinates in $[-180, 180] \times [-90, 90]$, and thus contains infinitely many candidate locations. Collecting trajectory data directly in continuous spaces is natural for many applications and offers three key advantages. (i) The privacy guarantee does not depend on the cardinality of a discretized location set (ii) The sampling mechanism operates directly on the continuous space, so its efficacy and efficiency are not constrained by the space size. (iii) Perturbed locations can be post-processed (e.g. rounded) to any discrete space embedded within the continuous space, making the approach applicable to both continuous and discrete settings.

This chapter proposes two LDP methods for trajectory collection in continuous spaces. The main insight is to decompose the 2D continuous space into 1D subspaces and design mechanisms for each subspace. These two mechanisms are built on existing utility-optimized piecewise-based mechanisms [184] for 1D bounded numerical domains.[¶] Based on two different decompositions of the continuous space, we obtain two methods: TraCS-D and TraCS-C.

Specifically, Chapter 6 makes the following contributions:

- This is the first work to develop trajectory collection methods for continuous spaces under pure LDP. It highlights the benefits of operating directly in continuous spaces (rather than discretizing) and proposes TraCS-D and TraCS-C accordingly. The key insight is to decompose the 2D continuous space into two 1D subspaces and to build 2D trajectory perturbation mechanisms from existing utility-optimized 1D piecewise-based mechanisms, leveraging the direction and coordinate information in continuous trajectories.
- TraCS also applies to discrete spaces. Compared with existing approaches for discrete spaces, TraCS has substantially lower computational complexity for generating perturbed locations.

[§]Appendix D.2.3 further discusses the challenges of adapting discrete mechanisms to continuous spaces.

[¶]Another category of LDP mechanisms for bounded numerical domains is truncated mechanisms (e.g. the truncated Laplace mechanism [73, 97]). While such mechanisms can be incorporated into TraCS, they are more complex and typically less effective than piecewise-based mechanisms. Section 6.2.5 provides details, and Section 6.3 includes experimental comparisons.

## 3.4 Quantification of Classifier Utility under LDP

The above sections focus on designing LDP mechanisms that optimize data utility at the mechanism level. Moving beyond mechanism-level utility metrics toward downstream tasks, this section studies how to quantify the utility of classifiers under LDP. Classifiers map input data to class labels and underpin a wide range of industrial applications, including predictive modeling, data analysis, and image recognition [75, 87, 99]. When deployed as services, classifiers require users to submit input data that often contain sensitive information, such as medical records or financial attributes, thereby raising serious privacy concerns. While users seek to benefit from classification services, they may be unwilling to disclose their sensitive data. A common mitigation strategy is user-side data perturbation, such as adding noise to numerical data or applying blurring and other obfuscation techniques to images before sending them to the classifier. Data perturbation remains a lightweight and intuitive solution for privacy-preserving classification [24, 179, 186]. Among these approaches, LDP mechanisms provide users with provable privacy guarantees. However, such perturbations inevitably degrade the utility of the classifier, leading to a fundamental research question: *How can we quantify the utility of classifiers when their inputs are perturbed by LDP mechanisms?*

**Utility of classifiers under LDP.** For simple queries such as summation, utility can be quantified analytically via the mean squared error (MSE) of LDP mechanisms [150,153]. For classification tasks, however, utility is typically measured by classification accuracy, which cannot be written analytically in terms of MSE. A straightforward way to evaluate classifier utility under an LDP mechanism with a given privacy parameter $\varepsilon$ is to repeatedly perturb the data and measure the proportion of correctly classified instances, as illustrated in Figure 3.3. Although practical, this empirical approach has significant limitations: it applies only to specific choices of $\varepsilon$ and particular perturbed datasets; changing $\varepsilon$ requires time-consuming re-evaluation, and different random perturbations lead to different results. Moreover, it does not reveal how utility depends on the privacy parameter, making it unsuitable for systematic comparison of LDP mechanisms. In contrast, an analytical utility quantification framework—analogous to MSE for summation queries—would provide principled guidance for the design of classifiers under LDP, but such a framework is currently lacking.

Quantifying the relationship between privacy and classifier utility presents significant analytical challenges. These challenges stem from two aspects: (i) LDP mechanisms inherently introduce perturbations across the entire data domain, potentially causing zero utility for classifiers. (ii) Classifiers are often complex or even black-box functions, making it difficult to analyze their behavior under perturbations.

Chapter 7 develops a framework to theoretically quantify classifier utility under LDP mechanisms. This framework addresses the above challenges via two key insights: (i) LDP mechanisms generate

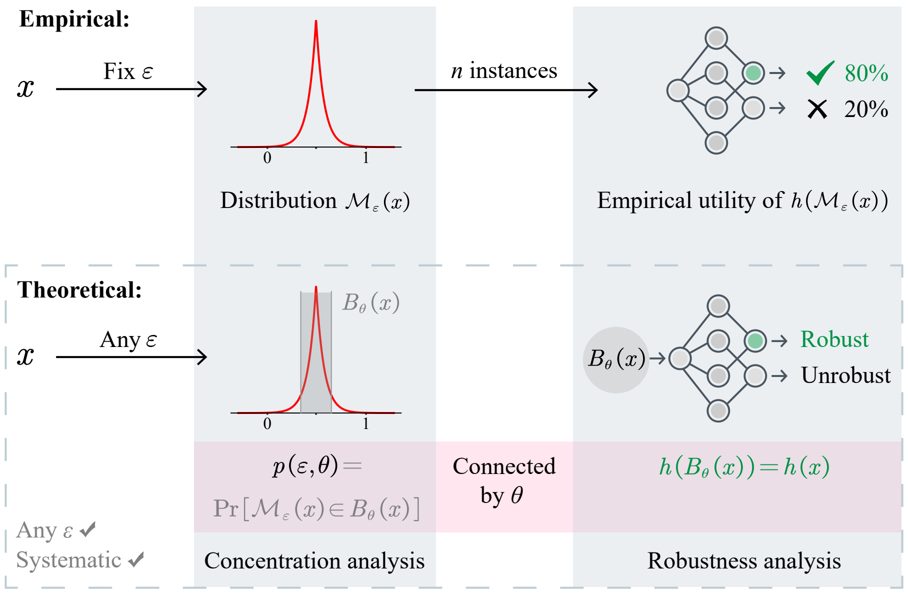


Figure 3.3: From empirical to theoretical utility analysis. Chapter 7 analytically quantifies classifier utility under LDP mechanisms by linking the *concentration analysis* of LDP mechanisms with the *robustness analysis* of classifiers.

perturbed data that, with high probability, concentrates within a bounded region around the original data, making extreme perturbations rare. (ii) Within this concentration region, utility analysis of a classifier can be reformulated as a robustness analysis problem. Here, robustness characterizes how reliably a classifier preserves its predictions under input perturbations. Established robustness analysis techniques can determine the maximum permissible perturbation that leaves the classifier's output unchanged, thereby preserving utility. By combining the concentration analysis of LDP mechanisms with the robustness analysis of classifiers, a theoretical characterization of data utility can be obtained as a function of the privacy parameter $\varepsilon$, formalized as: *Given classifier $h$, LDP mechanism $\mathcal{M}_\varepsilon$, and raw data $x$, with probability at least $p(\varepsilon, \theta)$, $h$ preserves its correct classification result under $\mathcal{M}_\varepsilon(x)$.*

**Applications.** The proposed utility quantification framework has direct applications in privacy-preserving classification systems: (i) It enables a comparative analysis of different LDP mechanisms for a given classifier by evaluating their probability guarantees $p(\varepsilon, \theta)$. An LDP mechanism that provides a higher $p(\varepsilon, \theta)$ ensures better utility at the same privacy parameter. (ii) The framework facilitates the selection of an appropriate privacy parameter $\varepsilon$ to meet specific utility requirements. For example, given a utility threshold $p^*$, the framework identifies the $\varepsilon$ that satisfies $p(\varepsilon, \theta) \geq p^*$, achieving a precise privacy–utility balance when using the classifier.

Specifically, Chapter 7 makes the following contributions:

- *Quantification framework.* It introduces the first analytical framework for quantifying classifier utility under LDP-perturbed inputs by linking the concentration analysis of LDP mechanisms with the robustness analysis of classifiers, enabling principled utility evaluation.

- *Refinement techniques.* It develops two refinement techniques that enhance utility quantification. The first extends robustness from a scalar "robustness radius" to an axis-aligned "robustness hyperrectangle", enabling tighter robustness analysis. The second adapts the PAC privacy notion by introducing a new privacy indicator and an extended Gaussian mechanism that is applicable to any $\varepsilon$.

# Chapter 4

# Correlated Perturbation for LDP

## 4.1 Preliminaries

This section formulates the problem of frequency estimation for binary data under LDP, then reviews the classical randomized response (RR) mechanism, which serves as the basis for the proposed joint randomized response (JRR) mechanism.

### 4.1.1 Problem Formulation

Consider a system comprising a data collector and a set of users $\{u_1, u_2, \cdots, u_n\}$. Each user holds a binary value $x_i \in \mathcal{X} = \{0, 1\}$. For each $x \in \mathcal{X}$, the data collector aims to estimate the count $n_x$ of users whose value equals $x \in \{0, 1\}$. To protect privacy, users do not submit $x_i$ directly. Instead, each user $u_i$ applies an $\varepsilon$-LDP mechanism $\mathcal{M}$ to produce a perturbed report $y_i = \mathcal{M}(x_i)$ and sends $y_i$ to the data collector. The objective is to design an LDP frequency-estimation scheme that improves the estimation accuracy of $n_x$ while maintaining the per-user $\varepsilon$-LDP guarantee as the classical RR mechanism.

### 4.1.2 Review of Randomized Response

Randomized Response (RR) [159] was originally proposed to provide plausible deniability for respondents answering a sensitive binary question in surveys. Under RR, each user reports the true value with probability $p$, and reports the opposite value with probability $q = 1 - p$. The RR mechanism satisfies $\varepsilon$-LDP when $p \leq e^{\varepsilon}/(1 + e^{\varepsilon})$.

Assume there are $n$ users in total, let $n_x$ denote the number of users whose true value equals $x$ for each $x \in \mathcal{X}$, $I_x$ be the number of perturbed reports equal to $x$ received by the data collector. An unbiased estimator of $n_x$ is

$$\hat{n}_x = \frac{I_x - nq}{p - q},$$

which is standard in RR-based frequency estimation [153, 159].

Data utility is commonly measured by the variance of the unbiased estimator $\hat{n}_x$, given by

$$\mathrm{Var}[\hat{n}_x] = \frac{\mathrm{Var}[I_x]}{(p-q)^2} = \frac{npq}{(p-q)^2}.$$

Smaller variance corresponds to higher estimation accuracy (and thus higher utility).

## 4.2 Impact of Correlated Perturbation

This section discusses how introducing correlation among different users' perturbations can affect both data privacy and data utility, using illustrative examples.

In traditional LDP protocols, each user perturbs their data independently. As a result, the relevant aggregate statistic (e.g. the count of reported 1's) can be written as a sum of individual perturbed values, and the variance of the corresponding estimator is simply the sum of the individual variances.

By contrast, when multiple users jointly perturb their data, the variance of the estimator also depends on the covariance between their perturbed outputs. By carefully designing the joint perturbation to induce negative covariance, it is possible to reduce the estimator variance and thus improve data utility. Two examples are presented below to illustrate this idea.

**Example 2.** *(Independent Randomized Response) Suppose there are two users, $u_1$ and $u_2$, with values $x_1 = 1$ and $x_2 = 1$, respectively. Each user independently perturbs their value using RR with parameter $p = 0.8$ (thus $q = 1 - p = 0.2$). Let $T_j$ be the indicator of whether user $u_j$ reports truthfully, i.e. $T_j = 1$ if $y_j = x_j$ and $T_j = 0$ otherwise. Then*

$$T_j = \begin{cases} 1, & \textit{with probability } p = 0.8, \\ 0, & \textit{with probability } q = 0.2. \end{cases}$$

**Estimation $\hat{\boldsymbol{n}}_{\mathbf{1}}$:** Let $I_1$ be the number of perturbed reports equal to 1 received by the data collector. An unbiased estimator of $n_1$ is

$$\hat{n}_1 = \frac{I_1 - 2q}{p - q} = \frac{I_1 - 0.4}{0.6}.$$

**Data privacy:** Since $p/q = 0.8/0.2 = 4$, the RR mechanism in this example satisfies $(\ln 4)$-LDP.

**Data utility:** The variance of $\hat{n}_1$ is

$$\mathrm{Var}[\hat{n}_1] = \frac{npq}{(p-q)^2} = \frac{2 \cdot 0.8 \cdot 0.2}{0.6^2} = \frac{0.32}{0.36} \approx 0.89.$$

**Example 3.** *(Joint Randomized Response) Consider the same two users as in Example 2, i.e. $u_1$ and $u_2$ with values $x_1 = 1$ and $x_2 = 1$, respectively. Let $T_j$ be a binary indicator of whether user $u_j$ reports truthfully, i.e. $T_j = 1$ if $y_j = x_j$ and $T_j = 0$ otherwise. The two users jointly perturb their data according to the joint distribution in Table 4.1.*

Table 4.1: Joint reporting probability in Example 3.

| | $T_1 = 1$ | $T_1 = 0$ |
|---|---|---|
| $T_2 = 1$ | 0.60 | 0.2 |
| $T_2 = 0$ | 0.2 | 0 |

**Estimation $\hat{n}_1$:** From Table 4.1, it follows that $\Pr[T_j = 1] = 0.8$ and $\Pr[T_j = 0] = 0.2$ for both $j \in \{1, 2\}$. To derive an unbiased estimator of $n_1$, compute the expectation of $I_1 := y_1 + y_2$, i.e. the number of perturbed reports equal to 1 received by the data collector:

$$\begin{aligned} \mathrm{E}[I_1] &= \mathrm{E}\left[\sum_{j=1}^{2} y_j\right] = \sum_{j=1}^{2} \mathrm{E}[y_j] = \sum_{j=1}^{2} \Pr[y_j = 1] \\ &= n_1 \cdot \Pr[T_j = 1] + (2 - n_1) \cdot \Pr[T_j = 0] = 0.8n_1 + 0.2(2 - n_1) \\ &= 0.4 + 0.6n_1. \end{aligned}$$

Therefore, an unbiased estimator of $n_1$ is $\hat{n}_1 = (I_1 - 0.4)/0.6$, which is identical to the estimator in Example 2.

**Data privacy:** If privacy is evaluated solely from each user's marginal reporting distribution, the mechanism matches RR with $p = 0.8$ and $q = 0.2$, and thus satisfies $(\ln 4)$-LDP under the standard single-user view. However, correlation between users in the same group can lead to additional leakage in the presence of collusion. A detailed analysis is provided later in Section 4.3.2.

**Data utility:** The variance of the unbiased estimator is

$$\mathrm{Var}[\hat{n}_1] = \frac{\mathrm{Var}[I_1]}{0.36} = \frac{25}{9}\mathrm{Var}[y_1 + y_2] = \frac{25}{9}\big(\mathrm{Var}[y_1] + \mathrm{Var}[y_2] + 2\mathrm{Cov}[y_1, y_2]\big),$$

where $\mathrm{Cov}[y_1, y_2]$ denotes the covariance between $y_1$ and $y_2$.

(i) The terms $\mathrm{Var}[y_1]$ and $\mathrm{Var}[y_2]$ are equal due to identical marginals. Since $x_1 = x_2 = 1$, we have $\mathrm{E}[y_j] = \Pr[y_j = 1] = \Pr[T_j = 1] = 0.8$ and $\mathrm{E}[y_j^2] = \Pr[y_j = 1] = 0.8$. Hence,

$$\mathrm{Var}[y_1] + \mathrm{Var}[y_2] = 2\mathrm{Var}[y_1] = 2(\mathrm{E}[y_1^2] - \mathrm{E}^2[y_1]) = 2(0.8 - 0.8^2) = 0.32.$$

(ii) To compute $\mathrm{Cov}[y_1, y_2]$, note that when $x_1 = x_2 = 1$, $y_1y_2 = 1$ iff $(T_1, T_2) = (1, 1)$, and $y_1y_2 = 0$ otherwise. Therefore,

$$\mathrm{E}[y_1y_2] = 1 \cdot \Pr[T_1 = 1, T_2 = 1] + 0 \cdot \Pr[T_1 = 0, T_2 = 0] = 0.60,$$

and

$$\mathrm{Cov}[y_1, y_2] = \mathrm{E}[y_1y_2] - \mathrm{E}[y_1]\mathrm{E}[y_2] = 0.60 - 0.8 \cdot 0.8 = -0.04.$$

Combining (i) and (ii) gives

$$\mathrm{Var}[\hat{n}_1] = \frac{1}{0.36}\big(0.32 + 2(-0.04)\big) = \frac{0.24}{0.36} \approx 0.67,$$

which is smaller than the variance 0.89 obtained in Example 2.

From the two examples above, we observe that data utility can be improved by introducing negative correlation between $y_1$ and $y_2$ through joint perturbation of two users. The next section provides a theoretical analysis that generalizes these examples to arbitrary $n$ and user inputs $x_i$. We will also address the following design questions for joint perturbation mechanisms:

- How can the joint perturbation mechanism in Example 3 be generalized? (Section 4.3.1)
- Can the joint perturbation mechanism provide the same level of data privacy as RR? If so, under what conditions? (Section 4.3.2) How should the data utility of such a mechanism be quantified, and when can it outperform RR? (Section 4.3.3)
- How can a joint perturbation mechanism be instantiated in a deployable and secure manner? (Section 4.4)

## 4.3 Joint Randomized Response

This section formalizes the general JRR mechanism and summarizes its key properties, then analyzes its privacy and utility, discuss how to choose parameters to maximize utility under a target privacy level, and finally presents a deployable instantiation.

Table 4.2: Joint reporting probability in JRR. $T_{2i-1}$ and $T_{2i}$ denote the truthfulness indicators for users $u_{2i-1}$ and $u_{2i}$, respectively. Parameters satisfy $0.5 < p \leq 1$, $q = 1 - p$, and $1 - 1/p \leq \rho \leq 1$ to ensure all probabilities are valid.

| | $T_{2i-1} = 1$ | $T_{2i-1} = 0$ |
|---|---|---|
| $T_{2i} = 1$ | $p^2 + \rho pq$ | $(1-\rho)pq$ |
| $T_{2i} = 0$ | $(1-\rho)pq$ | $q^2 + \rho pq$ |

### 4.3.1 General JRR Mechanism

Consider $n$ users $\{u_1, \ldots, u_n\}$, where each user holds a binary value. The users are partitioned uniformly at random into $n/2$ disjoint groups of size two, denoted by $G_1, \ldots, G_{n/2}$.[*] Without loss of generality, assume each group $G_i$ consists of users $u_{2i-1}$ and $u_{2i}$ for all $1 \leq i \leq n/2$. Within each group $G_i$, the two users jointly perturb their values according to the joint distribution in Table 4.2.

#### Properties of JRR

The general JRR mechanism has several key properties, summarized below.

**Marginal probabilities ($p$ and $q$).** In each group $G_i = \{u_{2i-1}, u_{2i}\}$, the marginal distribution of each truthfulness indicator matches RR with parameter $p$ (and $q = 1 - p$). Specifically,

$$\begin{aligned}
\Pr[T_{2i-1} = 1] &= \Pr[T_{2i-1} = 1, T_{2i} = 1] + \Pr[T_{2i-1} = 1, T_{2i} = 0] \\
&= (p^2 + \rho pq) + (1-\rho)pq = p, \\
\Pr[T_{2i-1} = 0] &= 1 - \Pr[T_{2i-1} = 1] = q,
\end{aligned}$$

and symmetrically $\Pr[T_{2i} = 1] = p$ and $\Pr[T_{2i} = 0] = q$. Hence, each user reports truthfully with probability $p$ (and untruthfully with probability $q$).

**Correlation coefficient ($\rho$).** The parameter $\rho$ is the correlation coefficient between $T_{2i-1}$ and $T_{2i}$, i.e.

$$\mathrm{Corr}[T_{2i-1}, T_{2i}] = \frac{\mathrm{Cov}[T_{2i-1}, T_{2i}]}{\sigma_{T_{2i-1}} \sigma_{T_{2i}}} = \rho,$$

where $\mathrm{Cov}[\cdot, \cdot]$ denotes covariance and $\sigma_{\cdot}$ denotes standard deviation. A proof is given in Appendix B.1.1.

[*]Random grouping is more general and practical in real-world applications. JRR can also be applied under a fixed grouping, in which case the analysis is simpler. For completeness, Section 4.4 discusses secure and deployable random-grouping methods against malicious users and servers.

**RR as a special case of JRR.** When $\rho = 0$, $T_{2i-1}$ and $T_{2i}$ become independent Bernoulli($p$) variables. In this case, JRR reduces to independently applying RR to each user.

**Same frequency estimator ($\hat{n}_x$).** JRR uses the same unbiased frequency estimator as RR. Let $I_x$ be the number of received reports equal to $x \in \{0, 1\}$. Then the data collector estimates

$$\hat{n}_x = \frac{I_x - nq}{p - q}.$$

Appendix B.1.2 shows that this estimator remains unbiased under JRR.

### 4.3.2 Data Privacy Analysis

Because the two users in a group perturb their data jointly, one user's report distribution depends on the other's truthfulness indicator. Consequently, if a user colludes with the data collector, then knowing that user's raw data and truthfulness indicator may increase the collector's confidence about the partner user's report, thereby weakening privacy compared to independent RR. On the other hand, the random grouping ensures that each user is paired uniformly at random, so a target (non-colluding) user's partner is colluding only with limited probability.

**Adversary model.** We consider honest-but-curious adversaries, consisting of the data collector together with a subset of colluding users, who follow the JRR protocol but attempt to infer as much as possible about non-colluding users' raw data. The adversary observes all perturbed reports and any protocol messages exchanged among users, and additionally knows the raw data and truthfulness indicators of all colluding users. For the random-grouping, we assume that whenever a group contains a colluding user, the adversary also learns the identity of the other user in that group.[†]

Let $\mathcal{C}$ denote the set of users colluding with the collector. In addition to the perturbed reports $y_1, \ldots, y_n$, the adversary learns the collection of truthfulness indicators of colluding users, $\mathcal{T}_c \coloneqq \{T_j : j \in \mathcal{C}\}$. Under this adversary model, the following theorem gives a (worst-case) lower bound on the privacy level provided by JRR.

**Theorem 3.** *Assume that $n$ users are divided into $n/2$ groups uniformly at random, and the adversary knows $\mathcal{T}_c$ for a subset of colluding users $\mathcal{C}$. Then for any user $u_i$, the JRR mechanism $\mathcal{M}$ satisfies*

$$\frac{\Pr[\mathcal{M}(x_i) = y_i \mid \mathcal{T}_c]}{\Pr[\mathcal{M}(x_i') = y_i \mid \mathcal{T}_c]} \leq e^{\varepsilon}$$

[†]This can happen when joint perturbation is implemented via user-to-user communication, in which case communication metadata may reveal identities.

*for any pair of inputs* $x_i, x'_i \in \{0,1\}$ *and any output* $y_i \in \{0,1\}$, *where*

$$\varepsilon = \ln \frac{mp_{\max} + (n-m-1)p}{mp_{\min} + (n-m-1)q},$$

*with* $m := |\mathcal{C}|$, $p_{\max} := \max\{(1-\rho)p, p+\rho q\}$, *and* $p_{\min} := \min\{(1-\rho)q, q+\rho p\}$.

*Proof.* (Sketch) The key step is to bound $\Pr[\mathcal{M}(x_i) = y_i \mid \mathcal{T}_c]$ by identifying its maximum and minimum values given the adversary's side information. Fix a user $u_i$ and let $u_j$ be its (randomly chosen) partner. (i) If $u_j$ is non-colluding, then conditioning on $\mathcal{T}_c$ does not affect the distribution of $y_i$ (beyond the marginal RR behavior). (ii) If $u_j \in \mathcal{C}$, which happens with probability $m/(n-1)$ under uniform random grouping, then $\Pr[\mathcal{M}(x_i) = y_i \mid \mathcal{T}_c]$ depends on the realized value of $T_j$ and can be computed from the joint table in Table 4.2. Combining these two cases yields an upper bound on the conditional probability ratio, giving the stated $\varepsilon$. See Appendix B.1.3 for the full proof. $\square$

Privacy result in Theorem 3 involves $p_{\max}$ and $p_{\min}$, which are determined by $\rho$ given $p$ and $q$. It is straightforward to verify that

- if $\rho \le 0$, then $p_{\max} = (1-\rho)p$ and $p_{\min} = q + \rho p$;
- if $\rho \ge 0$, then $p_{\max} = p + \rho q$ and $p_{\min} = (1-\rho)q$.

Substituting these into (3) yields

$$e^{\varepsilon} = \begin{cases} -1 + \dfrac{n-1}{n-1-(n-1-m\rho)p} & \text{if } \rho \le 0, \\ -1 + \dfrac{n-1}{(n-1-m\rho)q} & \text{if } \rho \ge 0. \end{cases}$$

**Effect of $\boldsymbol{\rho}$.** For fixed $n$, $p$, and $m$, the quantity $e^{\varepsilon}$ depends on $\rho$ only through the term $(n-1-m\rho)$ in the denominator. In particular, when $m > 0$, strengthening the correlation, i.e. making $\rho$ more negative when $\rho \le 0$, or larger when $\rho \ge 0$, reduces $(n-1-m\rho)$ and thus increases $e^{\varepsilon}$, which corresponds to a weaker privacy guarantee.

**Effect of $\boldsymbol{m}$.** For fixed $n$, $p$, and $\rho \ne 0$, a larger number of colluding users $m$ decreases $(n-1-m\rho)$ in the denominator (in both cases $\rho \le 0$ and $\rho \ge 0$), thereby increasing $e^{\varepsilon}$ (and thus $\varepsilon$). Intuitively, more collusion amplifies the extra leakage introduced by correlation. If $\rho = 0$, then $m$ has no effect.

**Comparison with RR.** When $\rho = 0$, JRR degenerates to independent RR. In this case, $e^{\varepsilon} = p/q$, which matches the privacy guarantee of standard RR and is independent of $m$.

### 4.3.3 Data Utility Analysis

Example 3 shows that JRR can achieve higher data utility than RR. This subsection provides a general variance analysis for JRR's unbiased frequency estimator $\hat{n}_x$.

**Theorem 4.** *Assume that $n$ users are divided into $n/2$ groups uniformly at random. The variance of JRR's frequency estimator $\hat{n}_x$ is*

$$\mathrm{Var}[\hat{n}_x] = \frac{pq}{(p-q)^2} \cdot \left( n + \frac{\rho\left((2n_1 - n)^2 - n\right)}{n-1} \right),$$

*where $n_1$ denotes the number of users whose raw data equals 1 (and $n_0 = n - n_1$).*

*Proof.* (Sketch) The total variance can be reduced to the sum of the contributions from the $n/2$ random groups. Let $g_{1,1}$ be the (random) number of $(1,1)$-groups, i.e. groups in which both users have raw data 1. Since $g_{1,1}$ determines how many groups induce covariance, we can apply the law of total variance and compute the conditional variance over the three group types given $g_{1,1}$. This yields the stated expression. See Appendix B.1.4 for the full proof. □

The expression in Theorem 4 can be decomposed into two terms:

$$\mathrm{Var}[\hat{n}_x] = \underbrace{\frac{pqn}{(p-q)^2}}_{\text{RR baseline}} + \underbrace{\frac{pq\,\rho\left((2n_1 - n)^2 - n\right)}{(p-q)^2(n-1)}}_{\text{correlated perturbation}},$$

where the first term equals the variance under independent RR, and the second term captures the effect of correlation.

Treating RR as the baseline, both $\rho$ and $n_1$ determine the sign and magnitude of the correlated-perturbation term (assuming $p > 0$ and $n > 1$). In particular, the factor $(2n_1 - n)^2 - n$ satisfies

$$(2n_1 - n)^2 - n \begin{cases} \geq 0 & \text{if } \dfrac{n_1}{n} \in \left[0, \dfrac{1}{2} - \dfrac{1}{2\sqrt{n}}\right] \cup \left[\dfrac{1}{2} + \dfrac{1}{2\sqrt{n}}, 1\right], \\ \leq 0 & \text{if } \dfrac{n_1}{n} \in \left[\dfrac{1}{2} - \dfrac{1}{2\sqrt{n}}, \dfrac{1}{2} + \dfrac{1}{2\sqrt{n}}\right]. \end{cases}$$

In typical data-collection settings, $n$ is large, so the region where $(2n_1 - n)^2 - n \leq 0$ is quite small. For example, when $n = 10^4$, this region has length $1/\sqrt{n} = 0.01 = 1\%$.

**Effect of $\boldsymbol{\rho}$.** For most values of $n_1$ (i.e. when $(2n_1 - n)^2 - n \geq 0$), negative correlation $\rho < 0$ reduces the variance relative to RR, and a smaller $\rho$ yields a smaller variance. Specifically,

$$\mathrm{Var}[\hat{n}_x] \propto \begin{cases} \rho & \text{for most values of } n_1, \\ -\rho & \text{when } n_1/n \text{ is close to } 1/2. \end{cases}$$

Recall that the smallest feasible value of $\rho$ is $\rho \geq 1 - 1/p$; hence $\rho = 1 - 1/p$ minimizes the variance for most $n_1$. Conversely, when $n_1/n$ is close to $1/2$, the variance is minimized at $\rho = 1$.

**Effect of $\boldsymbol{n_1/n}$.** When $\rho < 0$, the correlated-perturbation term scales with $(2n_1 - n)^2$, and thus depends quadratically on $n_1$:

$$\mathrm{Var}[\hat{n}_x] \propto \begin{cases} n_1^2 & \text{if } \dfrac{n_1}{n} \in \left[0, \dfrac{1}{2} - \dfrac{1}{2\sqrt{n}}\right], \\ -n_1^2 & \text{if } \dfrac{n_1}{n} \in \left[\dfrac{1}{2} + \dfrac{1}{2\sqrt{n}}, 1\right]. \end{cases}$$

In this regime, the variance is smaller when $n_1$ is close to 0 or close to $n$. Since the intermediate interval has length $1/\sqrt{n}$, this behavior holds for nearly all $n_1$ when $n$ is sufficiently large.

**Comparison with RR.** When $\rho = 0$, JRR reduces to independent RR and Theorem 4 matches the classical RR variance. Moreover, JRR can achieve strictly smaller variance; although this improvement depends on the unknown $n_1$, choosing $\rho < 0$ heuristically improves utility for nearly all values of $n_1$.

### 4.3.4 Choosing $p$ and $\rho$ for JRR

This subsection discusses how to choose the reporting probability $p$ and the correlation parameter $\rho$ for JRR under a target privacy level $\varepsilon$.

#### Optimization Problem Formulation

Suppose we want JRR to provide the same $\varepsilon$-LDP level as an RR scheme. We therefore need to select $p$ and $\rho$ such that the privacy bound in Theorem 3 is at most $\varepsilon$.

Using the privacy and utility results above, we can formalize the parameter selection as the following constrained optimization problem:

$$\begin{aligned} \arg\min_{p,\rho} \quad & Var(p, \rho) \\ \text{s.t.} \quad & \frac{mp_{\max} + p(n - m - 1)}{mp_{\min} + q(n - m - 1)} \leq e^{\varepsilon}, \\ & 1 - \frac{1}{p} \leq \rho \leq 1, \end{aligned}$$

where $Var(p, \rho)$ is the variance of JRR in Theorem 4, the first constraint enforces privacy (no weaker than $\varepsilon$), and the second constraint specifies the feasible range of $\rho$.

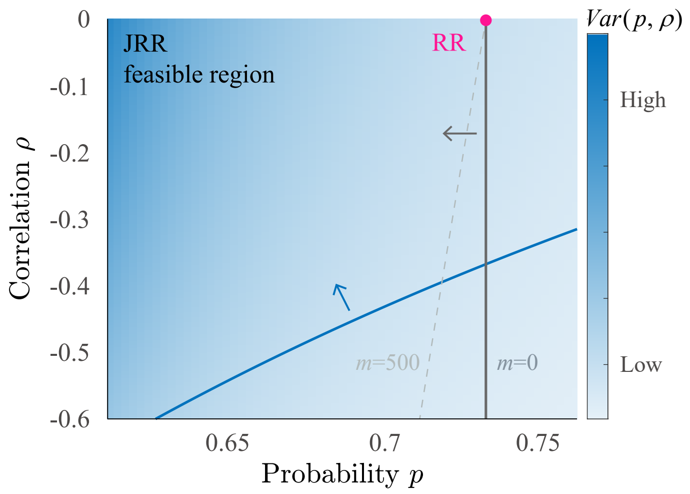


Figure 4.1: Heatmap of $Var(p,\rho)$ and the feasible region when $m = 0$ and $m = 500$ (with $\varepsilon = 1$, $n = 10^4$, and $n_1 = 100$). Lighter blue indicates lower variance; RR is marked by the pink point.

**Unobservable parameters $n_1$ and $m$.** The objective $Var(p,\rho)$ depends on $n_1$, which is the quantity we aim to estimate, and its value affects the direction in which $Var(p,\rho)$ is optimized (as discussed in the previous subsection). Another unobservable parameter is the number of colluding users $m$, which affects the feasible region of $(p,\rho)$ but is generally unknown and hard to estimate. As a result, solving for a closed-form optimum that is simultaneously optimal for all $n_1$ and $m$ is intractable.

Fortunately, we showed that choosing $\rho < 0$ improves utility for nearly all values of $n_1$. Moreover, for fixed $(p,\rho)$, the privacy constraint becomes tighter as $m$ increases (i.e. larger $m$ yields a smaller feasible region). These observations motivate a simpler heuristic.

### Heuristic Solution

We show that restricting to $\rho < 0$ and treating $m$ as small matches common scenarios and yields a simple closed-form choice of $(p,\rho)$.

**Restrict to $\rho < 0$.** If we optimize over $\rho < 0$, then the resulting $(p,\rho)$ is also optimal for

$$\forall n_1 : \ \frac{n_1}{n} \in \left[0, \frac{1}{2} - \frac{1}{2\sqrt{n}}\right] \cup \left[\frac{1}{2} + \frac{1}{2\sqrt{n}}, 1\right],$$

which covers almost the entire range of $n_1/n \in [0,1]$. For example, if private bits are roughly uniformly distributed among $10^4$ users, JRR improves over RR with probability $1 - 1/\sqrt{10^4} = 99\%$.[‡]

[‡]JRR can also improve over RR when $n_1/n$ is close to $1/2$ by choosing $\rho > 0$. However, this region has width

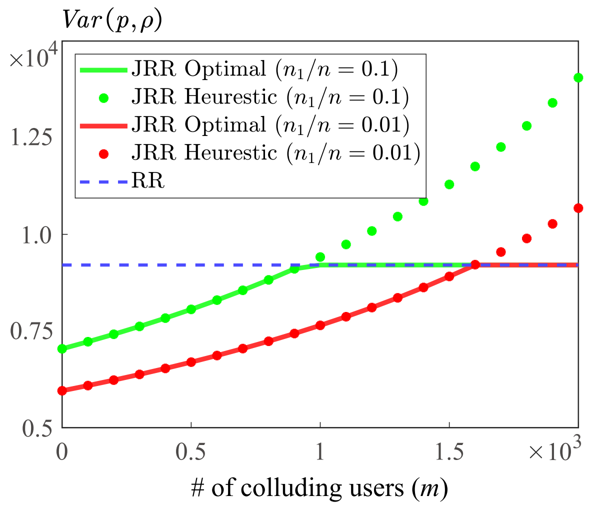


Figure 4.2: Comparison of the variance $Var(p, \rho)$ of JRR (optimal and heuristic) with RR as functions of $m$ and $n_1$ when $\varepsilon = 1$ and $n = 10^4$.

In many real-world settings (e.g. social-media telemetry), $n$ is large, and estimation accuracy is typically analyzed under such large-$n$ assumptions.

Under $\rho < 0$, the privacy constraint simplifies to

$$p \leq \frac{(n-1)e^{\varepsilon}}{(e^{\varepsilon}+1)(n-1-\rho m)},$$

which still depends on the (unknown) collusion size $m$.

**Treat $m$ as small.** If $m$ were known, we could solve the constrained problem directly. In practice, it is useful to distinguish between a *weak-adversary* regime (small $m$) and a *strong-adversary* regime (large $m$). The weak-adversary assumption is often reasonable in privacy-preserving data collection, since large-scale collusion is typically difficult to coordinate and may be illegal. It is also consistent with the fact that Theorem 3 gives a worst-case lower bound on privacy. For intuition, Figure 4.1 visualizes the objective and the feasible region for $m = 0$ and $m = 500$ when $\varepsilon = 1$ and $n = 10^4$.

With these two heuristics, we have $n - 1 - \rho m \approx n - 1$, yielding the closed-form choice

$$p = \frac{e^{\varepsilon}}{e^{\varepsilon}+1}, \quad \rho = 1 - \frac{1}{p},$$

which maximizes utility in common scenarios by pushing $\rho$ to its smallest feasible value.

This heuristic is generally effective for moderate collusion levels. Figure 4.2 compares $Var(p, \rho)$ for JRR (optimal and heuristic) against RR as functions of $m$ and $n_1$ when $\varepsilon = 1$ and $n = 10^4$. The

$1/\sqrt{n}$ and is typically negligible when $n$ is large. Thus, we focus on $\rho < 0$ for practicality.

heuristic matches the optimum up to approximately 17% colluding users when $n_1/n = 0.1$, and up to approximately 10% colluding users when $n_1/n = 0.9$. Additional settings of $n_1/n$ under the heuristic solution are reported in Appendix B.2.1.

## 4.4 Deployable Instantiation of JRR

This subsection presents a deployable instantiation of the JRR mechanism. The main challenges of implementing JRR in practice lies the following two steps:

- Secure random grouping:[§] How to divide $n$ users into $n/2$ disjoint groups of two uniformly at random without a trusted server.
- Secure correlated perturbation: How to let two users in each group jointly perturb their data according to the joint probability distribution in Table 4.2 without leaking their reporting truthfulness to each other.

We solve the first challenge by designing a secure shuffling protocol leveraging secure multi-party computation (MPC) techniques, and the second challenge by designing a correlated randomness generation protocol based on two-party computation (2PC) techniques.

### 4.4.1 Secure Random Grouping via MPC

If a trusted server is available, then it can simply generate a secure random grouping and send each user its group assignment. In practice, the trusted server can be replaced by a set of untrusted users. The key insight comes from *multi-party computation* (MPC) [173], which allows a set of users to jointly compute a function over their inputs while keeping those inputs private. In our case, the function is the grouping and the inputs are the individual users' identities. The MPC protocol should ensure that no user has access to other users' grouping information, thus preventing leakage of user identities.

Our approach is to treat users' identities as a list of secret-shared items, and then securely shuffle this list using an oblivious sorting network. Specifically, the protocol works as follows:

1. Each user $i$ generates a secret-shared key $[r_i]$ (e.g. over the field $\{0, 1\}^{64}$).

[§]In JRR, "secure" requires that the random grouping is at least not controlled by a single party, so that it cannot bias the grouping to assign a target user to a colluding partner.

2. (Oblivious sorting) For a pair $(u_i, u_j)$ in the sorting network (e.g. a bitonic sorting network), compute a secret bit $[b] = [r_i < r_j]$.¶ Then, obliviously swap the two users' positions if $b = 1$.
3. Form a secure grouping by pairing adjacent users in the shuffled list, i.e. $\{(u_{\pi_1}, u_{\pi_2}), (u_{\pi_3}, u_{\pi_4}), \ldots, (u_{\pi_{n-1}}, u_{\pi_n})\}$.

**View of the adversary.** This method is similar to the *secure multi-party shuffling* (SMPS) protocol [116]. In the above protocol, each user $u_i$ knows only its own share $[r_i]$ and its own new position $u_{\pi_i}$ after shuffling. It learns nothing about other users' positions,‖ since the messages received from other users are merely shares that information-theoretically independent of other users' positions on their own.

### 4.4.2 Secure Correlated Perturbation via 2PC

To enable two users in each group to jointly perturb their data according to the joint probability distribution in Table 4.2, we design a correlated randomness generation protocol based on two-party computation (2PC). In this scenario, the joint-computed function outputs are $T_{2i-1}, T_{2i} \in \{0, 1\}$ and the inputs are two random numbers from users $u_{2i-1}$ and $u_{2i}$, respectively. The 2PC protocol should ensure that no user has access to the other user's random number, thus preventing inference of the other's $T$ value.

**Probability-table encoding.** To ease presentation, we encode the joint probabilities in Table 4.2 into cumulative probabilities. Specifically, we define four scaled cumulative probabilities over the field $\{0, 1\}^{64}$ as follows:

$$\begin{aligned} R_1 &:= \lfloor (p^2 + \rho pq) \cdot 2^{64} \rfloor, \\ R_2 &:= R_1 + \lfloor (1-\rho)pq \cdot 2^{64} \rfloor, \\ R_3 &:= R_2 + \lfloor (1-\rho)pq \cdot 2^{64} \rfloor. \end{aligned}$$

Our direction is to sample $(T_{2i-1}, T_{2i})$ according a *secret-shared* random key $R \in \{0, 1\}^{64}$ and the above scaled cumulative probabilities. The protocol works as follows:

1. Users $u_{2i-1}$ and $u_{2i}$ independently sample random key $r_{2i-1}$ and $r_{2i}$ over the field $\{0, 1\}^{64}$,

¶A single share is information-theoretically meaningless on its own (as a placeholder), and becomes a well-defined bit in $\{0, 1\}$ only after being combined with the other party's share. For brevity, we omit the details of boolean MPC for secure comparisons such as $[r_i < r_j]$ from multiple shares here and treat them as atomic operations.

‖In fact, this protocol provides a stronger guarantee than what JRR's privacy analysis requires: it suffices that no single party can control (or bias) the random grouping; here, moreover, the shuffled positions remain hidden.

respectively. Let

$$R \coloneqq (r_{2i-1} + r_{2i}) \bmod 2^{64}.$$

We view $r_{2i-1}$ and $r_{2i}$ as additive secret shares of the *implicit* value $R$.

2. Define the following (secret-shared) indicator bits:

$$I_{11} = [R < R_1],\ \ I_{10} = [R < R_2] - [R < R_1],\ \ I_{01} = [R < R_3] - [R < R_2],\ \ I_{00} = 1 - [R < R_3],$$

where $[P]$ equals 1 if predicate $P$ holds and 0 otherwise. Then set the outputs as

$$T_{2i-1} = I_{11} + I_{10}, \quad T_{2i} = I_{11} + I_{01}.$$

3. (Selective opening) User $u_{2i}$ sends its share of $T_{2i-1}$ to $u_{2i-1}$, and $u_{2i-1}$ reconstructs $T_{2i-1}$ by adding the received share to its own share. Symmetrically, $u_{2i-1}$ sends its share of $T_{2i}$ to $u_{2i}$, who reconstructs $T_{2i}$.

**Theorem 5** (Correctness)**.** *The above protocol correctly generates* $(T_{2i-1}, T_{2i})$ *with the same joint distribution as in Table 4.2.*

*Proof.* (Sketch) The probability of $(T_{2i-1}, T_{2i})$ in the above protocol can be computed, which coincides with that in Table 4.2. See Appendix B.1.6 for the full proof □

**View of the adversary.** In the above protocol, each user (e.g. $u_{2i-1}$) learns only its locally sampled random value $r_{2i-1}$ and the reconstructed output bit $T_{2i-1}$. It learns nothing about the other party's output $T_{2i}$, since the message received from $u_{2i}$ is merely an additive share and independent of $T_{2i}$.

### 4.4.3 Correspondence to JRR's Privacy Proof

The above instantiation subsumes the adversary model in Theorem 3. Specifically, it assumes an honest-but-curious adversary consisting of the data collector and a subset of colluding users: they execute the secure grouping and correlated perturbation protocols correctly, but attempt to infer the raw data of non-colluding users.

The secure random grouping protocol is designed to be non-malleable and to hide the grouping information of non-colluding users from the adversary. However, correlated perturbation is implemented via a 2PC protocol between the two users in each group. Consequently, if a group contains a colluding user, the communication (or its metadata) may reveal the identity of that user's partner to the adversary. This is precisely the additional side information captured in Theorem 3.

## 4.5 Extensions and Discussions

This section discusses extensions to non-binary domains and how JRR can be combined with advanced LDP mechanisms, and contrasts JRR's random grouping with the shuffle model.

### Extension to Non-Binary Data

JRR can be extended to non-binary data by redesigning the joint reporting distribution in Table 4.2. Consider a group of two users with values $x_1, x_2 \in [k]$, where $k \geq 2$ is the domain size. Let $(y_1, y_2)$ denote their reported values. Define the joint reporting distribution as

$$\Pr[(y_1, y_2)] = \begin{cases} p^2 + \rho pq & \text{if } y_1 = x_1,\ y_2 = x_2, \\ pq - \frac{1}{k-1}\rho pq & \text{if } y_1 = x_1,\ y_2 \neq x_2, \\ pq - \frac{1}{k-1}\rho pq & \text{if } y_1 \neq x_1,\ y_2 = x_2, \\ q^2 + \frac{1}{(k-1)^2}\rho pq & \text{if } y_1 \neq x_1,\ y_2 \neq x_2, \end{cases}$$

where $p + (k-1)q = 1$, and $\rho$ is the correlation coefficient between the two users' truthfulness indicators. This construction is analogous to the extension from RR to Generalized RR (GRR).

It is straightforward to verify that each user's marginal reporting distribution matches the binary case: a value $x$ is reported as itself with probability $p$, and as any other value with probability $q$. Consequently, the standard estimator $\hat{n}_x = (I_x - nq)/(p-q)$ remains unbiased for $n_x$. Appendix B.1.5 proves unbiasedness. Privacy and utility can be analyzed similarly to the binary setting.

### Integration with Advanced LDP Mechanisms

JRR can be combined with LDP mechanisms built on top of RR/GRR to improve the privacy–utility tradeoff. Below we illustrate this integration with Optimized Unary Encoding (OUE) [153] and Optimized Local Hashing (OLH) [153].

**Integration with OUE.** OUE encodes a value $x \in [k]$ as a $k$-bit unary vector $B = \mathsf{Encode}(x)$ with a single 1 at position $x$ and 0 elsewhere. It then applies RR independently to each bit:

$$\Pr[B'[j] = 1] = \begin{cases} p, & \text{if } B[j] = 1, \\ q, & \text{if } B[j] = 0, \end{cases}$$

where $B[j]$ denotes the $j$-th bit. The data collector aggregates the perturbed vectors to estimate frequencies.

JRR can be incorporated by modifying the perturbation step: instead of perturbing each user's bits independently, two users jointly perturb each corresponding bit of their unary vectors using the binary JRR mechanism. This induces negative correlation across paired users while preserving the intended per-user marginal behavior.

**Integration with OLH.** OLH uses the following encode–perturb–aggregate structure: (i) each user samples a hash function $H(\cdot)$ from a universal hash family and maps the original value $x$ to $x = H(x)$ in a smaller domain $[k]$; (ii) the user perturbs $x$ to a report $y$ using GRR:

$$\Pr[y = i] = \begin{cases} p, & \text{if } i = x, \\ q, & \text{for each } i \in [k] \setminus \{x\}; \end{cases}$$

(iii) the data collector aggregates reports to estimate frequencies in the hashed domain.

To integrate JRR with OLH, Step (ii) can be replaced with the non-binary JRR mechanism defined early on. Two users in a group jointly perturb their hashed values according to this joint distribution, improving estimation accuracy while maintaining the same marginal behavior as GRR.

### Extension to Larger-size Group

While this dissertation focuses on two-user groups, JRR can in principle be extended to larger groups. For a group of $k > 2$ users, the joint distribution of truthful reporting can be described by a $k$-dimensional probability table. Intuitively, increasing the group size may further improve the privacy–utility tradeoff by enabling richer (e.g. stronger negative) correlation structures that reduce estimator variance. However, as $k$ grows, the probability that a group contains one or more colluding users increases substantially, which can amplify privacy leakage and limit achievable utility gains.

Designing such a scheme is also increasingly complex. Consider $k = 3$ as an example. To preserve the same per-user marginal behavior as RR/JRR (so that each report, viewed in isolation, is indistinguishable from standard RR), each user must remain truthful with the same marginal probability $p$. Yet specifying the full joint distribution typically requires not only the three pairwise correlation coefficients $(\rho_{12}, \rho_{13}, \rho_{23})$, but also a third-order interaction term (e.g. a triple-correlation parameter $\rho_{123}$) to capture higher-order dependence. In general, the number of required dependence parameters grows rapidly with $k$. Even under symmetry assumptions that reduce the number of free parameters, selecting feasible values that simultaneously (i) keep all probabilities nonnegative, (ii) preserve the desired marginals, and (iii) maintain strong privacy guarantees remains challenging.

**Relationship with the Shuffle Model**

In the shuffle model [10, 26, 49, 61, 62, 100], users first encode (i.e. locally perturb) their data using an LDP mechanism and then send the perturbed data to a *trusted* shuffler, which randomly permutes all received messages before forwarding them to the data collector. Random shuffling can amplify privacy without degrading utility.

When each user can send only one bit (the single-message shuffle model), frequency estimation reduces to summing bits. In this setting, Ghazi et al. [61] show an optimal error of $\Omega(n^{1/4})$ by combining results of Erlingsson et al. [49] and Balle et al. [10]. A key technical tool in these analyses is the *Chernoff bound*. For more general real-valued summation, allowing multiple messages per user or longer messages can further improve accuracy [26, 62, 100].

**Different strengths of the results.** Our analysis of JRR yields error $\Theta(n^{1/2})$ for fixed $m$ and $\rho$, which matches RR in order and is worse than $\Omega(n^{1/4})$. However, the shuffle-model guarantees rely on substantially stronger assumptions than those used for JRR: (1) they are typically proved under $(\varepsilon, \delta)$-DP, and the Chernoff-bound-based privacy amplification crucially uses $\delta > 0$; (2) the optimal error depends on a restricted privacy regime (e.g. Ghazi et al. require $\varepsilon \leq 1$ [61]); and (3) the results are often stated in terms of $(\alpha, \beta)$-*accuracy*, which is weaker than reporting a worst-case (or variance-based) error bound. Beyond theory, we also evaluate JRR empirically on real datasets to validate the robustness of its gains.

**Different adversaries.** In the JRR setting, the adversary may partially learn the random pairing (e.g. through colluding users and communication metadata), although this is difficult in practice. As $m$ increases, anonymity from random grouping degrades; in contrast, shuffle-model results assume a perfectly trusted shuffler that provides full anonymity. This assumption is stronger than the adversary model considered for JRR.

Table 4.3: Summary of the datasets.

| **Dataset** | **Total** ($n$) | **Number of "1"** ($n_1$) | **Pct. of "1"** ($n_1/n$) |
|---|---:|---:|---:|
| Kosarak | $2 \times 10^4$ | 659 | 0.033 |
| Amazon | $1 \times 10^4$ | 762 | 0.076 |
| E-commerce | 23 486 | 19 314 | 0.822 |
| Census | $1 \times 10^4$ | 9 528 | 0.953 |
| Synthetic | $20 \sim 2 \times 10^6$ | $0 \sim 2 \times 10^6$ | $0 \sim 1.0$ |

## 4.6 Experiments

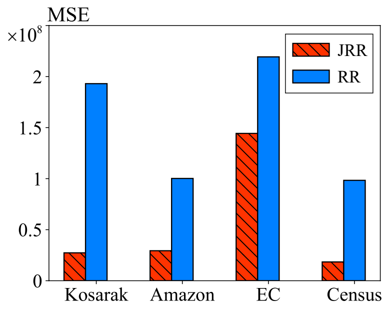
This section evaluates the utility of the JRR mechanism and analyzes how its parameters affect performance.

### 4.6.1 Setup

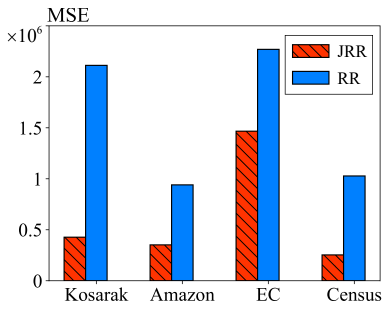
JRR's performance is evaluated on four real-world datasets: Kosarak [14], Amazon Rating [5], E-commerce [126], and Census [135]. Detailed descriptions are provided in Appendix B.2.2. In addition, synthetic datasets are generated with the total number of users $n$ ranging from 20 to $2 \times 10^6$ and the fraction of "1"s, $n_1/n$, ranging from 0 to 1. Table 4.3 summarizes all datasets used in the evaluation.

The RR mechanism is used as the benchmark because it is both the classical LDP protocol for frequency estimation and a special case of JRR. A comparison between JRR and the shuffle model is omitted for two reasons. (i) The privacy guarantee of the shuffle model is typically analyzed under $(\varepsilon, \delta)$-DP with $\delta \neq 0$ [10,26,49,100], whereas JRR provides pure $\varepsilon$-LDP (i.e. $\delta = 0$), making accuracy comparisons under matched guarantees nontrivial. (ii) The shuffle model often characterizes error via the $(\alpha, \beta)$-accuracy notion [61], which is weaker than the mean squared error used in this evaluation. A direct comparison with JRR would require extending these mechanisms to incorporate JRR and is therefore omitted. Nevertheless, JRR can potentially replace RR as a building block in these mechanisms and improve utility.

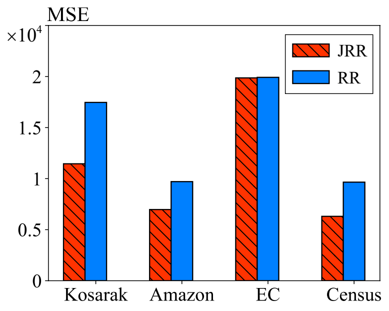
Data-utility comparisons are conducted under the same privacy parameter $\varepsilon$. For RR, utility is maximized by setting $p = e^\varepsilon/(1+e^\varepsilon)$. For JRR, the privacy parameter is characterized by Theorem 3, which generally yields a different $p$ than RR. Evaluations of data utility are based on the following two metrics:



(a) Privacy parameter $\varepsilon = 0.01$.

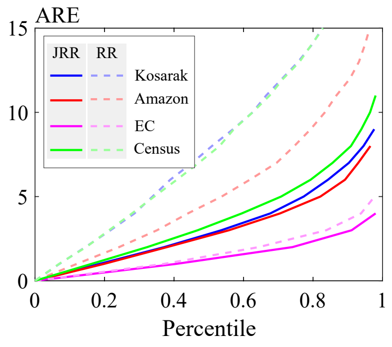


(b) Privacy parameter $\varepsilon = 0.1$.

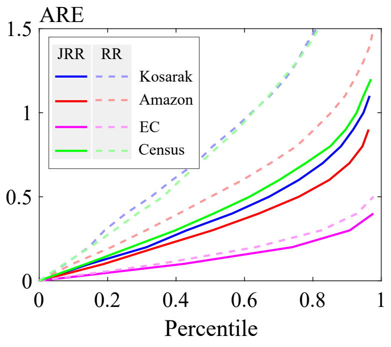


(c) Privacy parameter $\varepsilon = 1$.

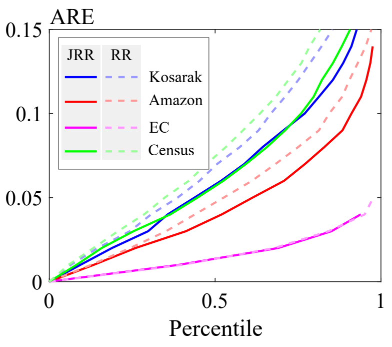
Figure 4.3: Comparison of MSE under RR and JRR on four real-world datasets when the privacy parameter $\varepsilon = 0.01$, 0.1 and 1.



(a) Privacy parameter $\varepsilon = 0.01$.



(b) Privacy parameter $\varepsilon = 0.1$.



(c) Privacy parameter $\varepsilon = 1$.

Figure 4.4: Percentiles of ARE under RR and JRR on four real-world datasets when the privacy parameter $\varepsilon = 0.01$, 0.1 and 1. A point on the curve indicates the percentage of error values that are lower than the corresponding ARE.

- Mean-squared error (MSE) [67, 153]: the mean squared error of the estimate $\hat{n}_x$ with respect to the ground truth $n_x$, averaged over all values, defined as

$$\text{MSE} = \frac{1}{|D|} \sum_{x \in D} (\hat{n}_x - n_x)^2. \tag{4.1}$$

- Average relative error (ARE) [90, 174]: the mean relative error over all values, defined as

$$\text{ARE} = \frac{1}{|D|} \sum_{x \in D} \frac{|\hat{n}_x - n_x|}{n_x}. \tag{4.2}$$

In the above formulas, $|D| = 2$ is the size of $D = \{0, 1\}$. Unlike MSE, ARE is affected by $n_x$ because it measures relative error; in particular, larger $n_x$ typically leads to smaller ARE.

Table 4.3 lists the default settings. All evaluations are performed in MATLAB. Each point in the figures below is averaged over 1 000 runs with distinct random seeds.

### 4.6.2 Results on Real-world Datasets

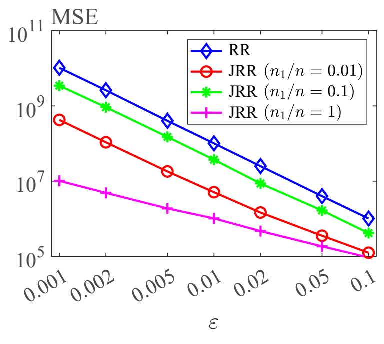
Figure 4.3 presents the MSE of JRR and RR on the four real-world datasets for $\varepsilon \in \{0.01, 0.1, 1\}$. Across all datasets and privacy settings, JRR achieves a lower MSE than RR. This is expected because the negative correlation between two users' perturbations under JRR can effectively reduce the expected MSE when the ratio $n_1/n$ is not close to 0.5. This condition holds for all four datasets, where $n_1/n$ is either below 0.1 or above 0.8. Moreover, for both mechanisms, the MSE decreases as $\varepsilon$ increases: with a larger privacy parameter, users report truthfully with higher probability, which reduces estimation error. It can be observed that JRR outperforms RR by a larger margin on the Kosarak and Census datasets than on the Amazon Rating and E-commerce datasets, especially when $\varepsilon$ is small (e.g. $\varepsilon = 0.01$). This is because $n_1/n$ in the Kosarak and Census datasets is farther from 0.5 than in the other two datasets.

Figure 4.4 shows the distributions of ARE over 1 000 runs under JRR and RR on the four real-world datasets, for $\varepsilon \in \{0.01, 0.1, 1\}$. Each point on a curve represents a percentile: the fraction of runs whose ARE is at most the corresponding value. For any fixed percentile, JRR consistently achieves a lower ARE than RR across all four datasets. For instance, when $\varepsilon = 0.1$ on the Kosarak dataset, the 80th percentile under JRR is 0.7, meaning that 800 out of 1 000 runs have ARE at most 0.7. In contrast, the 80th percentile under RR is 1.45. These results indicate that JRR provides both improved accuracy and more consistently low error than RR.

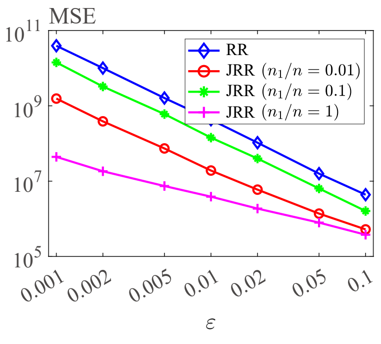
### 4.6.3 Results on Synthetic Datasets

Figure 4.5 reports the MSE of JRR and RR on synthetic datasets with $n \in \{10^4, 4 \times 10^4, 8 \times 10^4\}$. For a fixed ratio $n_1/n$ (each colored curve), the MSE decreases as $\varepsilon$ increases, and JRR consistently achieves a lower MSE than RR. The improvement is most pronounced in the high-privacy regime (small $\varepsilon$). For example, when $\varepsilon = 0.01$ and $n_1/n = 1$ (the first point on the purple curves), JRR's MSE is approximately 1% of that of RR; when $\varepsilon = 0.1$, it is about 10%. The results also confirm that the gain of JRR depends on $n_1/n$: when $n_1/n$ is far from 0.5 (e.g. the purple and red curves), JRR provides a much larger reduction in MSE than when $n_1/n$ is close to 0.5 (e.g. the green curves).

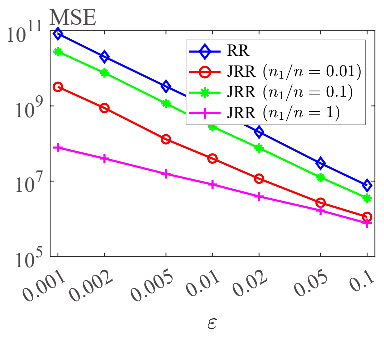
Figure 4.6 presents the corresponding ARE when $n_1/n = 0.01$ and 0.1 (blue and red curves, respectively). As expected, $n_1/n$ affects ARE of both mechanisms: a smaller $n_1$ typically leads to a larger ARE. Meanwhile, JRR (solid curves) consistently yields a lower ARE than RR (dashed curves). In particular, when $n_1/n = 0.01$, ARE of JRR is, on average, 21.1%, 20.3%, and 20.2% of that of RR for $n = 10^4$, $4 \times 10^4$, and $8 \times 10^4$, respectively.



(a) $n = 1 \times 10^4$.

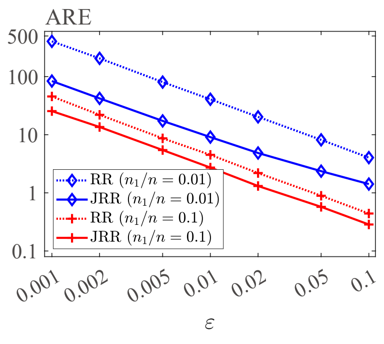


(b) $n = 4 \times 10^4$.

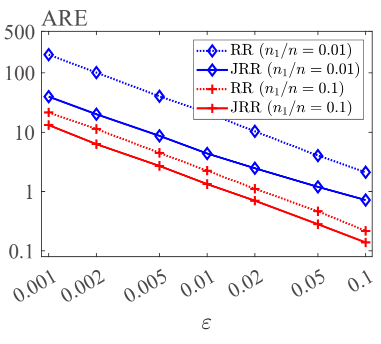


(c) $n = 8 \times 10^4$.

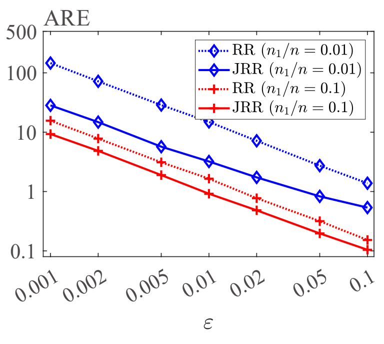
Figure 4.5: Comparison of MSE under RR and JRR with privacy parameter $\varepsilon = 0.01$ to 1.



(a) $n = 1 \times 10^4$.



(b) $n = 4 \times 10^4$.



(c) $n = 8 \times 10^4$.

Figure 4.6: Comparison of ARE under RR and JRR with privacy parameter $\varepsilon = 0.01$ to 1.

### 4.6.4 Summary of Evaluation Results

The evaluation on both real-world and synthetic datasets leads to the following conclusions:

- Across all real-world datasets and all privacy parameters, JRR consistently outperforms RR in terms of MSE and ARE, with larger gains when $\varepsilon$ is small.
- On synthetic datasets, JRR outperforms RR across all tested $\varepsilon$ and $n$ settings; the improvement becomes more significant as $\varepsilon$ decreases and $n$ increases.

## 4.7 Related Work

**Frequency estimation under LDP.** The concept of privacy-preserving frequency estimation dates back to Warner [159], who introduced the Randomized Response (RR) mechanism for collecting sensitive data. RR has been widely used in social science research to collect statistical information about embarrassing or illegal behaviors. RAPPOR [50] extends RR to handle non-binary data by encoding each user's value into a $d$-bit vector and applying RR independently to each bit. OLH [153] further refines this approach by introducing a *local hash* to compress the $d$-bit vector, thereby reducing communication cost. A comparative analysis of these frequency-estimation mechanisms and their variants is provided in [30].

Significant efforts have been made to improve the privacy–utility tradeoff for data analysis under LDP. A variance-analysis framework was proposed in [153] to optimize the parameters of RR-based mechanisms, thereby improving utility. Post-processing can further improve utility. For example, the non-negativity and sum-to-one constraints were applied in [156], where they were referred to as *consistency*. As another example, the convolution framework in [51] incorporates Wiener-filter-based deconvolution into existing LDP protocols to improve utility. Interactive protocols such as PrivKV [174] can iteratively improve estimation accuracy. Estimating the most frequent items (i.e. *heavy hitters*) can be accomplished via random projection, as shown in [12]. Cryptographic techniques can also strengthen privacy and thereby improve utility for a fixed privacy parameter. For example, Crypt$\varepsilon$ [28] leverages cryptography to achieve the same utility as centralized DP. However, none of these approaches consider correlated perturbations across different users. Moreover, some of these techniques are complementary to JRR and can be integrated with it to further improve utility (e.g. post-processing).

**Privacy for correlated data.** Privacy leakage due to correlations among data records has long been a concern. A line of research [23, 84, 122, 140, 170] studied this problem from both theoretical and practical perspectives. The Pufferfish framework [84] allows users to define customized privacy notions tailored to specific applications, offering several formal definitions for releasing correlated data. A practical mechanism for releasing correlated data [140] was subsequently adapted to the Pufferfish framework. Bayesian differential privacy [170] was proposed to analyze privacy under correlations from a Bayesian perspective. Under this definition, learning data correlations was studied in [23], which derived the minimum required noise using Bayesian networks. Moreover, applications such as graph data publication [91], correlated trajectory release [123], network data release [23], and trading statistics aggregation [122] have been studied in the context of differential privacy for correlated data. However, these works primarily focus on protecting privacy when the underlying data are correlated; none of them consider correlations among different users' random perturbations.

A separate line of research focuses on designing LDP mechanisms for various data types, including real-valued data [44, 150], multidimensional data [131, 150, 169], set-valued data [128, 151, 152], time-series data [158], social graph data [142], key–value pairs [66, 174], and directional data [160]. However, similar to existing LDP frequency-estimation techniques, these works do not consider correlated perturbations.

## 4.8 Conclusions

This chapter studies correlated random perturbations for locally differentially private frequency estimation and examines their effect on the utility–privacy tradeoff. A general Joint Randomized Response (JRR) mechanism is introduced, together with a practical instantiation. The proposed approach achieves the same local differential privacy guarantee as the classical Randomized Response (RR) mechanism while providing improved utility in most cases. These advantages are supported by theoretical analysis and extensive simulation studies conducted on both real-world and synthetic datasets.

# Chapter 5

# Optimal Piecewise-based Mechanism

## 5.1 Preliminaries

This section formulates the problem and presents existing instantiations of three-piecewise mechanisms (TPM) for collecting bounded numerical data under LDP, and their limitations, which motivate the proposed optimal generalized piecewise-based mechanism (OGPM).

### 5.1.1 Problem Formulation

We consider a typical data collection schema that consists of a set of *users* and one *collector*. Each user has a numerical sensitive data $x_i \in \mathcal{X}$, where $\mathcal{X}$ is a continuous and bounded domain. The collector needs to collect data from users for statistical estimations, such as the mean value and distribution of the data.

However, the collector is untrusted and may attempt to infer users' sensitive data. To protect privacy, each user locally randomizes their sensitive data using a privacy mechanism $\mathcal{M} : \mathcal{X} \to \mathcal{Y}$, then sends $y_i = \mathcal{M}(x_i)$ to the collector.

We seek to design an optimal $\mathcal{M}$ that maximizes the data utility by minimizing the distance between the sensitive data $x_i$ and the perturbed data $y_i$, while ensuring $\varepsilon$-LDP.

Table 5.1: This dissertation (OGPM) vs existing instantiations of TPM.

| **Mechanism** | **Domain** | **Optimality** | **Closed form** | **Estimation** |
|---|---|---|---|---|
| PM [150] | | No | Yes | Mean |
| SW [95] | Classical | No | Yes | Distribution |
| PTT [101] | | Partly[a] | No | Mean |
| OGPM | Classical & circular | Yes | Yes | Mean & Distribution |

[a] Proved the existence of the optimal under TPM, but did not give closed-form instantiations. Appendix C.2.1 provides detailed discussion.

### 5.1.2 Piecewise-based Mechanisms

When the input domain $\mathcal{X}$ is both continuous and bounded, the state-of-the-art mechanisms to achieve $\varepsilon$-LDP are piecewise-based mechanisms. These mechanisms are heuristic instances of the following definition.

**Definition 3.** 3-*piecewise mechanism (TPM) $\mathcal{M} : \mathcal{X} \to \mathcal{Y}_\varepsilon$ is a family of probability density functions that, given input $x \in \mathcal{X}$, outputs $y \in \mathcal{Y}$ according to*

$$\mathrm{pdf}[\mathcal{M}(x) = y] = \begin{cases} p_\varepsilon & \textit{if } y \in [l_{x,\varepsilon}, r_{x,\varepsilon}], \\ p_\varepsilon / e^\varepsilon & \textit{if } y \in \mathcal{Y}_\varepsilon \setminus [l_{x,\varepsilon}, r_{x,\varepsilon}], \end{cases}$$

*where $p_\varepsilon$ is a variable determined solely by $\varepsilon$,* while $l_{x,\varepsilon}$ and $r_{x,\varepsilon}$ depend on $x$ and $\varepsilon$. The output domain $\mathcal{Y}_\varepsilon \supset \mathcal{X}$ is an enlarged domain depending on $\varepsilon$.*

TPM samples the output $y$ for each $x$ from a piecewise distribution. This sampling is with a higher probability $p_\varepsilon$ within $[l_{x,\varepsilon}, r_{x,\varepsilon}]$ and a lower probability $p_\varepsilon/e^\varepsilon$ within the remaining two pieces $\mathcal{Y} \setminus [l_{x,\varepsilon}, r_{x,\varepsilon}]$, satisfying the $\varepsilon$-LDP constraint.

**Instantiations.** In TPM, the parameters are the central interval $[l_{x,\varepsilon}, r_{x,\varepsilon}]$, its probability $p_\varepsilon$, and the output domain $\mathcal{Y}_\varepsilon$. Different instantiations of those parameters yield different existing mechanisms [95, 101, 150]. For example, PM [150] is the first instantiation of TPM, defined on $[-1, 1] \to [-C_\varepsilon, C_\varepsilon]$, where $C_\varepsilon$ is a variable determined solely by $\varepsilon$ and the central interval has a fixed length $r_{x,\varepsilon} - l_{x,\varepsilon} = C_\varepsilon - 1$.

*Otherwise, if $p_\varepsilon$ varies with $x$, it violates the $\varepsilon$-LDP constraint because the probability ratio outputting the same $y$ from $x_1$ and $x_2$ is not bounded by $e^\varepsilon$.

**Data utility metric.** To quantify the data utility of different instantiations, we consider the general $\ell_p$-similar error metric (i.e. $|y-x|^p$) as a loss function $\mathcal{L}:\mathbb{R}\to\mathbb{R}$. Thus, the error is:

$$Err(x) = \int_{\mathcal{Y}} \mathcal{L}(y,x)\mathcal{P}_{\mathcal{M}(x)}\mathrm{d}y, \tag{5.1}$$

where $\mathcal{P}_{\mathcal{M}(x)}$ is the pdf defined by $\mathcal{M}(x)$. $Err(x)$ illustrates the expected error when applying $\mathcal{M}$ on $x$ under the loss function $\mathcal{L}$. For example, $\mathcal{L}(y,x) \coloneqq |y-x|$ is the absolute error, and $\mathcal{L}(y,x) \coloneqq (y-x)^2$ is the square error. Then $Err(x)$ corresponds to the mean absolute error (MAE) and mean square error (MSE) [101, 150], respectively. Lower $Err(x)$ indicates better data utility.

**Limitations of TPM.** Existing instantiations of TPM have the following limitations.

- *Not optimal in data utility.* None of the existing instantiations provided closed forms for the optimal data utility. Meanwhile, they also assume an invariable length $r_{x,\varepsilon}-l_{x,\varepsilon}$ of the central piece for all $x$, and symmetric probability $p_\varepsilon/e^\varepsilon$ for the remaining two pieces. However, a general-form piecewise-based mechanism can have more pieces, unfixed piece lengths, and asymmetric probabilities, potentially improving data utility.

- *Limited applicability.* Existing instantiations of TPM have enlarged and unfixed output domains $\mathcal{Y}_\varepsilon \supset \mathcal{X}$. Enlarged output domain incurs applicability issues in scenarios where the collector requires the output domain to align with the input domain (i.e. $\mathcal{Y}=\mathcal{X}$),[†] such as in common sensor-based services.

## 5.2 Generalized Piecewise-based Mechanism

This section generalizes TPM to its most general form (GPM). We introduce a framework for deriving the closed-form optimal GPM for the classical domain.

**Definition 4.** *Generalized m-piecewise mechanism (m-GPM) $\mathcal{M}:\mathcal{X}\to\mathcal{Y}$ is a family of probability density functions that, given input $x\in\mathcal{X}$, outputs $y\in\mathcal{Y}$ according to*

$$\mathrm{pdf}[\mathcal{M}(x)=y] = \begin{cases} p_{1,\varepsilon} & \textit{if } y\in[l_{1,x,\varepsilon}, r_{1,x,\varepsilon}), \\ \vdots & \vdots \qquad \vdots \\ p_{m,\varepsilon} & \textit{if } y\in[l_{m,x,\varepsilon}, r_{m,x,\varepsilon}), \end{cases}$$

$$\forall i,j\in[m], \max\frac{p_{i,\varepsilon}}{p_{j,\varepsilon}} \le e^\varepsilon,$$

[†]While post-processing the output by truncating it to $\mathcal{X}$ is possible, this approach may still result in low data utility. Sections 5.6.1 provide comparisons with mechanisms that include truncation.

*where $[m] \coloneqq \{1, ..., m\}$. Each probability $p_{i,\varepsilon}$ depends solely on privacy parameter $\varepsilon$, while interval boundaries $l_{i,x,\varepsilon}$ and $r_{i,x,\varepsilon}$ depend on both $x$ and $\varepsilon$.*

An $m$-GPM partitions its output domain into $m$ pieces, assigning probability $p_{i,\varepsilon}$ to each piece $[l_{i,x,\varepsilon}, r_{i,x,\varepsilon})$. The probabilities $p_{i,\varepsilon}$ are independent of the input $x$, and their ratios must be bounded by $e^{\varepsilon}$ to satisfy $\varepsilon$-LDP. For notational clarity, we omit subscripts $x$ and $\varepsilon$ when their context is clear. Additionally, $\mathcal{M}$ must satisfy standard probability requirements: non-negativity ($p_i \geq 0$), continuity ($r_i = l_{i+1}$), and normalization. TPM is a special case of GPM with $m = 3$.

Finding the optimal GPM requires determining both the optimal number of pieces $m$ and the corresponding $p_{i,\varepsilon}, l_{i,x,\varepsilon}, r_{i,x,\varepsilon}$. Due to the infinite possibilities for $m \in \mathbb{N}^+$ and the resulting $3m$ variables, analytical solutions are computationally intractable. We therefore propose a framework that combines analytical proofs with off-the-shelf optimization solvers.

### 5.2.1 Framework for Deriving the Optimal GPM

To derive the *closed-form* optimal GPM, we (i) formulate finding the optimal $m$-GPM as an optimization problem; (ii) determine the optimal $m$ based on the solutions of the optimization problem; (iii) derive the optimal closed-form expression (among all $m$-GPM).

**Optimal $m$-GPM.** To find the optimal GPM instantiation with $m$ pieces, we need to determine the variables $p_i$, $l_i$, and $r_i$. Any feasible assignment of these variables yields a mechanism $\mathcal{M}$ whose utility can be measured by $Err(x)$ from Formula (5.1). Finding the optimal $m$-GPM requires solving a min-max optimization problem that minimizes the worst-case error over all possible inputs $x$:[‡]

$$\begin{gathered}\min_{p_i, l_{i,x}, r_{i,x}} \max_{x} \int_{\mathcal{Y}} \mathcal{L}(y, x) \mathcal{P}_{\mathcal{M}(x)} \mathrm{d}y, \\ \text{s.t. } \mathcal{M} \text{ satisfies Definition 4.}\end{gathered} \tag{5.2}$$

This formulation yields the optimal $x$-independent $p_i$ values and the corresponding $l_{i,x}, r_{i,x}$ for the worst-case input $x$. However, since these $l_{i,x}, r_{i,x}$ may not be optimal for other inputs, we need a second optimization step using the obtained optimal $p_i$:

$$\begin{gathered}\min_{l_{i,x}, r_{i,x}} \int_{\mathcal{Y}} \mathcal{L}(y, x) \mathcal{P}_{\mathcal{M}(x)} \mathrm{d}y, \\ \text{s.t. } \mathcal{M} \text{ satisfies Definition 4 with } p_i.\end{gathered} \tag{5.3}$$

[‡]Worst-case error is the most common utility metric in mechanism design [101, 150]. We can also optimize the error at other specific points, see Section 5.5.

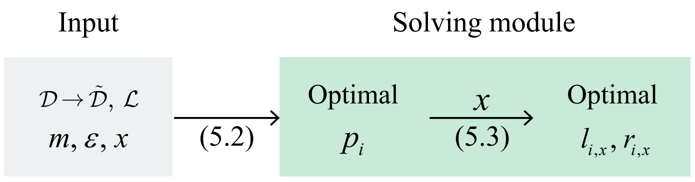


Figure 5.1: Solving flow for the optimal $m$-GPM. Two arrows indicate problems in (5.2) and (5.3).

Together, these two steps determine the optimal instantiation of $m$-GPM for any given domain mapping $\mathcal{X} \to \mathcal{Y}$, distance metric $\mathcal{L}$, piece number $m$, privacy parameter $\varepsilon$, and input $x$. Figure 5.1 illustrates this solving process.

**Challenges.** Nonetheless, solving Formulation (5.2) has practical difficulties. Even if it can be solved, there is still a gap between the solved optimal $m$-GPM and the closed-form optimal GPM (among all $m$-GPM). We detail them as follows.

- *(Solving difficulty)* Formulation (5.2) is a min-max problem, and the integrand $\mathcal{L}(y, x)\mathcal{P}_{\mathcal{M}(x)}$ is non-linear. It is a non-convex problem whose global optimal hard to solve.
- *(Optimal $m$)* The solved optimal is only for $m$-GPM given $m$. It is necessary to find the optimal piece number $m$.
- *(Closed form)* For practical usage, we need closed-form $p_i$, $l_{i,x}$ and $r_{i,x}$ (i.e. their relationships with $\varepsilon$ and $x$), rather than their specific values for every $\varepsilon$ and $x$.

**Solutions.** We address the challenges and gaps with the following solutions.

- Formulation (5.2) can be simplified to two *bilinear optimization* problems that can be solved by off-the-shelf solvers. (Section 5.2.2)
- If the optimal $(m + 1)$-GPM is identical to the $m$-GPM, then $m$ is the optimal piece number. (Section 5.2.2)
- Leveraging the above results, the optimal closed-form $p_i$, $l_{i,x}$ and $r_{i,x}$ can be obtained by analytical deduction (for TPM) or numerical regression (for any $m$-GPM). (Section 5.2.2)

Following these solutions, we can obtain the closed-form optimal GPM (among all $m$) for any $\varepsilon$ given $\mathcal{X} \to \mathcal{Y}$ and $\mathcal{L}$. Before presenting the detailed solutions, we first discuss the conditions under which the obtained GPM is optimal.

**Conditions for optimality.** When discussing optimality, the following aspects should be specified: (i) the error metric, (ii) the data domain and family of mechanisms, (iii) the strength of the optimality, and (iv) whether post-processing is allowed. In this dissertation, the optimality of GPM is defined with respect to: (i) the worst-case $\ell_p$-similar error metric, (ii) bounded numerical domains $\mathcal{X} \to \mathcal{Y}$ and mechanisms based on piecewise distributions, (iii) minimization of error value (not asymptotic or order-of-magnitude optimality), and (iv) without post-processing. These conditions are widely applicable in practice and literature. However, varying any of them may lead to different optimality results. Appendix C.2.2 provides a detailed discussion of these conditions and related optimalities.

### 5.2.2 Detailed Solutions

This subsection details the positive answers to the three challenges associated with deriving the optimal GPM.

#### Solution 1: Simplified Form

The min-max problem in Formulation (5.2) can be simplified. The key observation is that its inner maximization term, $\max_x$, has a closed form, i.e. the worst-case error is from the endpoints of $\mathcal{X}$. Lemma 1 states this observation.

**Lemma 1.** *Assume $\mathcal{X} = [a, b)$, the objective of Formulation (5.2) can be simplified to*

$$\min_{p_i, l_{i,x}, r_{i,x}} \max_{x \in \{a,b\}} \int_{\mathcal{Y}} \mathcal{L}(y, x) \mathcal{P}_{\mathcal{M}(x)} \mathrm{d}y.$$

*Proof.* (Sketch) The key of the proof is that each integral on $\mathcal{Y}$ is convex function w.r.t $x$. Thus, their non-negative weighted sum is also convex. According to the Bauer maximum principle [165], the maximum is achieved at the endpoints of $\mathcal{X}$, i.e. $x = a$ or $b$. Appendix C.1.1 provides the full proof. $\square$

**Complexity.** (i) Lemma 1 simplifies Formulation (5.2) to two *bilinear optimization* problems, i.e. when $x = a$ and $x = b$ respectively. The integrand $\mathcal{L}(y, a)\mathcal{P}_{\mathcal{M}(a)}$ includes terms such as $p_i l_i$ and $p_i r_i$, which involve multiplications of two variables. This problem can be solved by off-the-shelf solvers such as Gurobi [2], which employs stochastic *Branch and Bound* [68] method to handle the bilinear terms. (ii) The number of variables is at most $3m$, which can be efficiently solved for small $m$. For example, we can obtain the exact optimal for $m \leq 7$ and $\mathcal{L} = |y - x|$ within 2 seconds. Furthermore, solvers generally provide over- and under-approximation for bilinear problems [120]. We can obtain solutions with $\leq 1\%$ gap from the optimal for $m \leq 19$ within 1 minute. (iii) Formulation (5.3) for solving $l_i$ and $r_i$ has at most $2m$ variables and without $p_i$ terms, it is a toy-size problem when $m$ is small.

**Encoding details.** We need to encode the problem in Lemma 1 to simple mathematical expressions that can be handled by the solver. If we instantiate $\mathcal{L} = |y - x|$ and focus on the left endpoint $x = a$, this problem becomes:

$$\min_{p_i, l_i, r_i} \sum_{i=1}^{m} p_i \int_{l_i}^{r_i} (y - a) \mathrm{d}y = \min_{p_i, l_i, r_i} \sum_{i=1}^{m} \frac{p_i}{2} \left( (r_i - a)^2 - (l_i - a)^2 \right).$$

This problem is a bilinear optimization problem. The highest-degree term is $p_i \cdot r_i \cdot r_i$, which can be reformulated as $p_i \cdot t$ with $t = r_i \cdot r_i$, i.e. multiplication of bilinear terms. Such problems can be solved by off-the-shelf bilinear solvers [2]. The other problem in Formulation (5.3) can be encoded similarly but is much easier due to constant $p_i$ solved from this step.

**Solution 2: Optimal Piece Number**

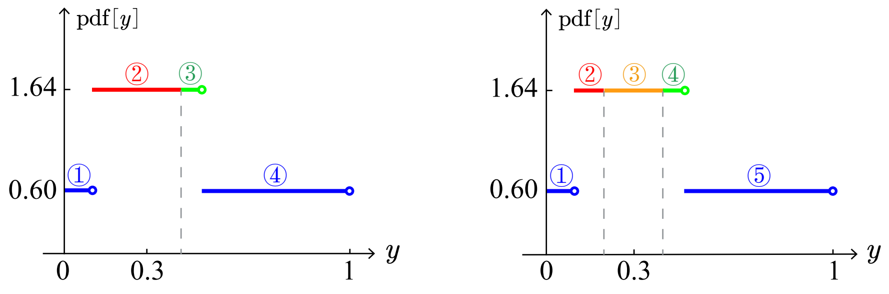
Although we can obtain the optimal $m$-GPM for any $m$ given sufficient time, solving for each $m$ is unnecessary. The following lemma provides a theoretical basis for capping the optimal number of pieces.

**Lemma 2.** *For all possible $\varepsilon$ and $x$, if the optimal $(m+1)$-GPM is the same as the $m$-GPM, then the optimal piece number is $m$.*

*Proof.* (Sketch) The key insight is that, if the optimal piece number is not $m$, i.e. an additional piece can lower the error, then this additional piece will be captured by the optimal $(m+1)$-GPM. Therefore, (i) if $m$ is not the optimal piece number, then the optimal $(m+1)$-GPM is different from the optimal $m$-GPM. (ii) if $m$ is the optimal piece number, then there is no additional piece can lower the error, making the optimal $(m+1)$-GPM is the same as the optimal $m$-GPM. Now, no additional piece can be captured, hence $m$ is the optimal piece number. Appendix C.1.2 provides the full proof. □

Lemma 2 requires checking the results for every $\varepsilon$ and $x$, which is challenging as $\varepsilon \in [0, \infty)$ and $x \in \mathcal{X}$ are infinite sets. In practice, we restrict $\varepsilon$ within its generally meaningful domain, e.g. $\varepsilon \in [0, 10)$, and employ *Monte Carlo random sampling* to generate a set of random pairs $(\mathcal{E}, \mathcal{X}) = \{(\varepsilon_1, x_1), (\varepsilon_2, x_2), \ldots, (\varepsilon_n, x_n)\}$. If Lemma 2 holds for the $n$-size set $(\mathcal{E}, \mathcal{X})$, the optimality of $m$-GPM is guaranteed with probability 1 as $n \to \infty$ [134].

**Results for Monte Carlo sampling.** Fixing $\mathcal{X} = \mathcal{Y} = [0, 1)$, for $\mathcal{L} = |y - x|$ and $\mathcal{L} = (y - x)^2$, the optimal 4-GPM are identical as the optimal 3-GPM for random $(\mathcal{E}, \mathcal{X})$ with at least $n = 10^4$. These optimal results align with TPM, but the optimal $p, l$ and $r$ values are different from existing instantiations.



(a) $m = 4$.  (b) $m = 5$.

Figure 5.2: Example of optimal 4-GPM and 5-GPM when $\varepsilon = 1$ and $x = 0.3$. They are identical as $m = 3$ after merging redundant pieces.

Given the continuity of the objective functions w.r.t $\varepsilon$ and $x$, we posit that it is, in fact, the exact optimal. Therefore, when we say the optimal piece number is $m = 3$, it implies the *statistical optimality* guaranteed by the Monte Carlo asymptotic technique with a strength of at least $n = 10^4$ random samples.

**Example 4.** *For $[0, 1) \to [0, 1)$ and $\mathcal{L} = |y - x|$, Figure 5.2 shows two examples of the optimal 4-GPM and optimal 5-GPM when $\varepsilon = 1$ and $x = 0.3$. After merging redundant pieces, i.e. connected pieces with the same probability, they are the same as the 3-GPM and fall into the TPM category.*

### Solution 3: Closed-form Instantiation

After determining the optimal $m$, we can derive the closed-form instantiation. If the optimal results exhibit $m = 3$ and coincide with TPM, it facilitates analytical deduction for the closed-form optimal $p$, $l$ and $r$. Otherwise, numerical regression can be employed to obtain the closed-form instantiation. Figure 5.3 illustrates the workflow.

**Analytical deduction.** For TPM, the optimization procedure for solving $p_i$, $l_i$ and $r_i$ can be conducted analytically. The key observation is that there are only three variables: $p, l$, and $r$ in TPM. Due to the normalization constraint of probability, the central interval length $r - l$ can be replaced by its probability $p$. This reduces the solving for the optimal $p$ to a univariate optimization problem w.r.t. $p$, which can be solved by analyzing the first-order derivative. With the solved $p$, solving $l$ and $r$ also becomes a univariate optimization problem. Appendix C.2.4 provides the formalized process.

**Numerical regression.** For any $m$-GPM, we can obtain the closed-form $p_i$, $l_i$, and $r_i$ through numerical regression on their solved optimal values.

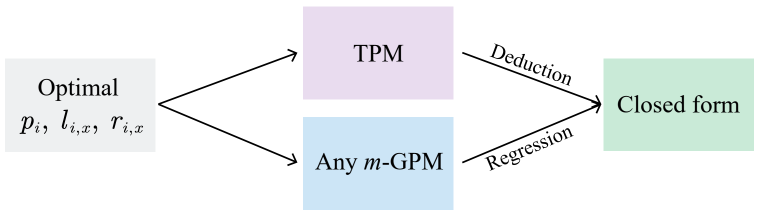


Figure 5.3: Solving flow for the optimal closed form.

Assume that we want to find the closed-form optimal $p_i$. Given $(\mathcal{E}, \mathcal{P}_i) = \{(\varepsilon_1, p_{i,1}), \dots, (\varepsilon_n, p_{i,n})\}$, which contains the solved optimal $p_i$ for random $\varepsilon$, we aim to find the relationship between $\varepsilon$ and $p_i$. This relationship can be approximated by $\hat{p}_i = f(\varepsilon, \beta)$, where $f$ is a designed feature with $\beta$ as regression parameters.

Ideally, the designed feature $f$ matches the truth form of $p_i$. In this case, the regression result of $f$ on $(\mathcal{E}, \mathcal{P}_i)$ converges to the optimal closed-form $p_i$. If not, the regression result may not converge to the optimal. In practice, we can use heuristic forms of $f$ for tighter approximation. Due to the structure of the LDP constraint, we suggest choosing $p_i$ with a form of $e^{\beta_1 \varepsilon}$, allowing us to design $f = e^{\beta_1 \varepsilon} + \beta_2$.

**Example 5.** *For $\mathcal{L} = |y - x|$, $\mathcal{X} = \mathcal{Y} = [0, 1)$ and $m = 3$, assume the probability $p$ has the form $\hat{p} = e^{\beta_1 \varepsilon} + \beta_2$. We use the* `scipy.curve_fit` *package to regress $p$ on 50 random $\varepsilon$; then its regression result is $\hat{p} = e^{\varepsilon/2} - 0.07$ with a maximal error $\leq 10^{-2}$. Note that this result almost coincides with the ground-truth $p$ in Theorem 7.*

### 5.2.3 Closed Form for Classical Domain

The above framework for deriving the closed-form optimal GPM is applicable to any $\mathcal{X} \to \mathcal{Y}$. Using this framework, this subsection provides instantiations for a common case: $\mathcal{X} = \mathcal{Y}$.

**Restricting domain.** In real-world applications, the output domain is often required to match the input domain, i.e. $\mathcal{X} = \mathcal{Y}$.[§] Furthermore, a concrete domain (e.g. $\mathcal{X} = [0, 1)$) does not limit the generality, as $\mathcal{X}$ can be transformed to other domains through scaling and shifting operations. The theorem below characterizes the privacy and utility invariants under such transformations.

**Theorem 6** (Transformation invariants)**.** *Given a GPM $\mathcal{M} : \mathcal{X} \to \mathcal{Y}$ satisfying $\varepsilon$-LDP, if domain*

[§]$\mathcal{Y}$ generally should be larger than or equal to $\mathcal{X}$ to ensure the mechanism's meaningfulness; otherwise, it means some ranges in $\mathcal{X}$ will disappear after applying the mechanism.

*$\mathcal{X}' = c\mathcal{X} + d$ and $\mathcal{Y}' = c\mathcal{Y} + d$ with $c > 0$, then transformation*

$$\mathcal{T}(\mathcal{M}) : x' = cx + d,\ p_i' = p_i/c,\ [l_i', r_i') = c[l_i, r_i) + d$$

*results in GPM $\mathcal{M}' = \mathcal{T}(\mathcal{M}) : \mathcal{X}' \to \mathcal{Y}'$ having the same privacy parameter $\varepsilon$. (privacy invariant)*

*Meanwhile, if there is another GPM $\mathcal{M}_{bad} : \mathcal{X} \to \mathcal{Y}$ with error $Err(x, \mathcal{M}) \leq Err(x, \mathcal{M}_{bad})$ for a given $x$, then*

$$Err(x', \mathcal{M}') \leq Err(x', \mathcal{M}'_{bad}),$$

*i.e. $\mathcal{T}$ maintains the data utility ordering. (utility invariant)*

*Proof.* (Sketch) The privacy invariant is the same as applying a linear *post-processing* of DP to GPM. The utility invariant is due to $\mathcal{T}$ as a linear function on $\mathcal{X}$. Appendix C.1.3 provides the full proof of these two invariants. □

Theorem 6 allows us to discuss the optimality on a fixed input domain. If a mechanism's input domain differs from $\mathcal{X}$, we can transform it to $\mathcal{X}$, and this transformation maintains the optimality.

**Hypothesis 1.** *For any domain $\mathcal{X} \to \mathcal{X}$, under absolute error and square error metrics, the optimal piecewise-based mechanism falls into 3-GPM.*

Support. We validated this hypothesis for $\mathcal{X} = [0, 1)$ by performing Monte Carlo sampling on $10^4$ random $(\varepsilon, x)$ pairs (detailed in Section 5.2.2). Theorem 6 then extends this optimality to any $\mathcal{X}$. Given the continuity of the objective functions w.r.t. $\varepsilon$ and $x$, we posit that $m = 3$ is indeed the exact optimal. To further support this hypothesis, Appendix C.2.3 outlines two directions for analytical proof and highlights the associated challenges.

**Theorem 7.** *If hypothesis 1 holds, then GPM $\mathcal{M} : [0, 1) \to [0, 1)$ with the following closed-form instantiation*

$$\text{pdf}[\mathcal{M}(x) = y] = \begin{cases} p_\varepsilon & \text{if } y \in [l_{x,\varepsilon}, r_{x,\varepsilon}), \\ p_\varepsilon / e^\varepsilon & y \in [0, 1) \setminus [l_{x,\varepsilon}, r_{x,\varepsilon}), \end{cases}$$

*where $p_\varepsilon = e^{\varepsilon/2}$,*

$$[l_{x,\varepsilon}, r_{x,\varepsilon}) = \begin{cases} [0, 2C) & \text{if } x \in [0, C), \\ x + [-C, C) & \text{if } x \in [C, 1 - C), \\ [1 - 2C, 1) & \text{otherwise}, \end{cases}$$

*with $C = (e^{\varepsilon/2} - 1)/(2e^\varepsilon - 2)$, is optimal for $[0, 1) \to [0, 1)$ under the absolute error and square error metric.*

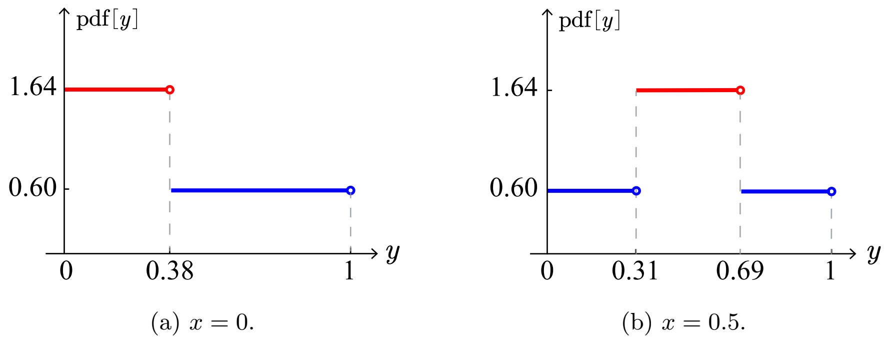


Figure 5.4: Optimal GPM (Theorem 7) when $\varepsilon = 1$, $x = 0$ and $x = 0.5$.

*Proof.* Provided by analytical deduction with $[a, b) = [\tilde{a}, \tilde{b}) = [0, 1)$, $\mathcal{L} = |y - x|$ and $\mathcal{L} = |y - x|^2$ respectively. Appendix C.1.4 provides the full proof. $\square$

This optimality on $[0, 1)$ can be transformed to $\mathcal{M}' : [a, b) \to [a, b)$ by applying $\mathcal{T} : x' = (b - a)x + a$, $p'_\varepsilon = p_\varepsilon/(b - a)$, $[l', r') = (b - a) \cdot [l, r) + a$, while maintaining the optimality.

**Example 6.** *Figure 5.4 shows two examples of Theorem 7. When $x = 0$ in the left figure, the optimal GPM has $p \approx 1.64$ and $[l, r) \approx [0, 0.38)$. When $x = 0.5$ in the right figure, the optimal GPM has $p \approx 1.64$ and $[l, r) \approx [0.31, 0.69)$.*

**MSE analysis.** We can calculate the mean squared error (MSE) of the optimal GPM in Theorem 7 as

$$\begin{aligned}\mathrm{MSE}[\mathcal{M}(x)] &= \int_{\mathcal{Y}} (y - x)^2 \cdot \mathrm{pdf}[\mathcal{M}(x) = y]\mathrm{d}y \\ &= \int_0^{l_{x,\varepsilon}} (y - x)^2 \frac{p_\varepsilon}{e^\varepsilon}\mathrm{d}y + \int_{l_{x,\varepsilon}}^{r_{x,\varepsilon}} (y - x)^2 p_\varepsilon \mathrm{d}y + \int_{r_{x,\varepsilon}}^1 (y - x)^2 \frac{p_\varepsilon}{e^\varepsilon}\mathrm{d}y.\end{aligned}$$

which leads to the results in Appendix C.2.5.

The theoretical MSE allows us to analytically compare the optimal GPM with existing mechanisms. For mechanisms defined on $\mathcal{X} = [0, 1)$, e.g. SW [95], we can compare their MSE with the above result. For mechanisms defined on other domains, e.g. PM [150] on $\mathcal{X} = [-1, 1)$, we can transform the optimal GPM to $[-1, 1)$ and compare their MSE. Detailed comparisons with them are presented in the evaluation section.

## 5.3 Optimal GPM for Circular Domain

This section presents the optimal GPM for the circular domain, another type of bounded domain. Circular domains are widely used in cyclic data such as time, angle, and compass direction. However, none of the existing piecewise-based mechanisms consider this type of domain, limiting their applicability.

**Different meanings of distance.** In the circular domain $[0, 2\pi)$, the distance between two elements differs from that in the classical domain $[0, 2\pi)$. For example, if we denote the distance between $x$ and $y$ in the circular domain as $\mathcal{L}_{\text{mod}}(y, x) = |y - x|$, then it implies $\mathcal{L}_{\text{mod}}(2\pi, 0) = 0$ and $\mathcal{L}_{\text{mod}}(3\pi/2, 0) = \pi/2$, which are different from $\mathcal{L}(y, x) = |y - x|$ in the classical domain. The biggest difference is that there are no endpoints in the circular domain. This unique property makes the mechanisms designed for the classical domain not directly suitable for the circular domain. Although we can "flatten" the circular domain to the classical domain, this conversion changes the distance between elements, leading to data utility loss.

Formally, in the circular domain $[0, 2\pi)$, the distance metric $\mathcal{L}_{\text{mod}}(y, x)$ has the following relationship with $\mathcal{L}(y, x)$ in the classical domain:

$$\mathcal{L}_{\text{mod}}(y, x) = \min\big(\mathcal{L}(y, x), \mathcal{L}(y, 2\pi - x)\big),$$

i.e. the distance between $y$ and $x$ is the smaller one between two arcs from $y$ to $x$. Under this distance metric, finding the optimal $m$-GPM for the circular domain is to solve $\mathcal{M} : [0, 2\pi) \to [0, 2\pi)$ such that

$$\begin{gathered}\min_{p_i, l_{i,x}, r_{i,x}} \max_{x} \int_0^{2\pi} \mathcal{L}_{\text{mod}}(y, x) \mathcal{P}_{\mathcal{M}(x)} \mathrm{d}y, \\ \text{s.t. } \mathcal{M} \text{ satisfies Definition 4,}\end{gathered} \tag{5.4}$$

and use the solved optimal $x$-independent $p_i$ to determine $l_{i,x}, r_{i,x}$ for any given $x$:

$$\begin{gathered}\min_{l_{i,x}, r_{i,x}} \int_0^{2\pi} \mathcal{L}_{\text{mod}}(y, x) \mathcal{P}_{\mathcal{M}(x)} \mathrm{d}y \\ \text{s.t. } \mathcal{M} \text{ satisfies Definition 4 with } p_i.\end{gathered} \tag{5.5}$$

We show that these two problems can be reduced to those in the classical domain, thereby enabling the usage of existing results.

### 5.3.1 Reduced Forms

Similar to the classical domain, the min-max objective of Formulation (5.4) also has a closed-form solution, which reduces the problem to the classical domain.

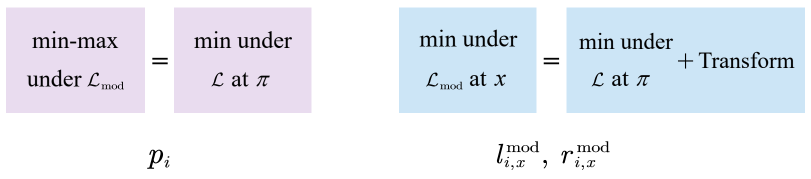

min-max under $\mathcal{L}_{\text{mod}}$
=
min under $\mathcal{L}$ at $\pi$
min under $\mathcal{L}_{\text{mod}}$ at $x$
=
min under $\mathcal{L}$ at $\pi$
+ Transform
$p_i$
$l_{i,x}^{\text{mod}}$, $r_{i,x}^{\text{mod}}$

Figure 5.5: Reduced forms of solving the optimal $p_i$, $l_{i,x}^{\text{mod}}$ and $r_{i,x}^{\text{mod}}$. Optimizations under circular distance $\mathcal{L}_{\text{mod}}$ can be reduced to those under linear distance $\mathcal{L}$.

**Lemma 3.** *The objective of Formulation (5.4) can be reduced to*

$$\min_{p_i, l_{i,x}, r_{i,x}} \int_0^{2\pi} \mathcal{L}(y,\pi)\mathcal{P}_{\mathcal{M}(\pi)} \mathrm{d}y.$$

*Proof.* (Sketch) We prove it by showing that, for any fixed $y$, $\max_x \mathcal{L}_{\text{mod}}(y,x) = \mathcal{L}_{\text{mod}}(y,\pi) = \mathcal{L}(y,\pi)$, i.e. the maximum distance between $y$ and $x$ is achieved at a unique $x = \pi$ for any $y$. Appendix C.1.5 provides the full proof. □

Following this reduction, the optimal results in the classical domain $[0, 2\pi)$ can be applied to determine the optimal $p_i$.

For Formulation (5.5) to solve $l_{i,x}$ and $r_{i,x}$, we shift the domain $[0, 2\pi)$ by $\pi - x$. This is a trick operation as $\mathcal{L}_{\text{mod}}(y,x)$ is transformed to the following form:

$$\begin{aligned}
\mathcal{L}_{\text{mod}}(y+\pi-x, x+\pi-x) =& \mathcal{L}_{\text{mod}}(y+\pi-x, \pi) \\
=& \min\left(\mathcal{L}(y+\pi-x,\pi), \mathcal{L}(y+\pi-x, 2\pi-\pi)\right) \\
=& \mathcal{L}(y+\pi-x,\pi),
\end{aligned}$$

which transforms $\mathcal{L}_{\text{mod}}$ to $\mathcal{L}$ at $x = \pi$. Then, the optimization problem in the shifted domain becomes

$$\min_{l_{i,x}, r_{i,x}} \int_{\pi-x}^{3\pi-x} \mathcal{L}(y+\pi-x,\pi)\mathcal{P}_{\mathcal{M}(\pi)} \mathrm{d}y.$$

It is a problem in the classical domain $[\pi - x, 3\pi - x)$. We can obtain the closed-form optimal $l_{i,x}$ and $r_{i,x}$ by applying the results of the classical domain. Since the obtained $l_{i,x}$ and $r_{i,x}$ depends on $x$ in the shifted domain, we shift them back to the circular domain using

$$\begin{aligned}
l_{i,x}^{\text{mod}} &= l_{i,x} - (\pi - x) \mod 2\pi, \\
r_{i,x}^{\text{mod}} &= r_{i,x} - (\pi - x) \mod 2\pi.
\end{aligned}$$

Transformation invariants ensure their optimality in the circular domain. Figure 5.5 summarizes the above two reductions to solve the optimal $p_i$, $l_{i,x}^{\text{mod}}$ and $r_{i,x}^{\text{mod}}$ in the circular domain.

### 5.3.2 Closed Form for Circular Domain

By applying the above reductions and following the same steps as in the classical domain, we can obtain the optimal GPM for the circular domain.

**Theorem 8.** *If Hypothesis 1 holds, then GPM $\mathcal{M} : [0, 2\pi) \to [0, 2\pi)$ with the following closed-form instantiation*

$$\text{pdf}[\mathcal{M}(x) = y] = \begin{cases} p_\varepsilon & \text{if } y \in [l^{\text{mod}}_{x,\varepsilon}, r^{\text{mod}}_{x,\varepsilon}), \\ p_\varepsilon / e^\varepsilon & y \in [0, 2\pi) \setminus [l^{\text{mod}}_{x,\varepsilon}, r^{\text{mod}}_{x,\varepsilon}), \end{cases}$$

*where $p_\varepsilon = \frac{1}{2\pi} e^{\varepsilon/2}$,*

$$l^{\text{mod}}_{x,\varepsilon} = \left( x - \pi \frac{e^{\varepsilon/2} - 1}{e^\varepsilon - 1} \right) \mod 2\pi,$$

$$r^{\text{mod}}_{x,\varepsilon} = \left( x + \pi \frac{e^{\varepsilon/2} - 1}{e^\varepsilon - 1} \right) \mod 2\pi,$$

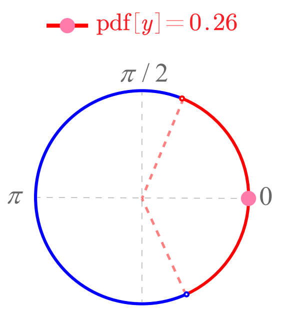
*is optimal for the circular domain under the absolute error and square error metric.*

*Proof.* Combination of the reduced forms, the results of the classical domain, Lemma 3, and transformation invariants leads to the conclusion. □

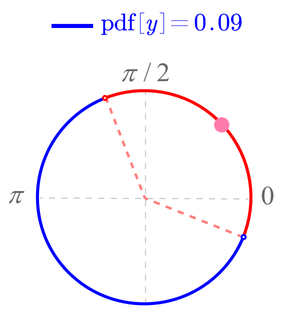
Compared to the optimal GPM for the classical domain in Theorem 7, the key difference is that the mechanism in the circular domain allows $[l^{\text{mod}}_{x,\varepsilon}, r^{\text{mod}}_{x,\varepsilon})$ to span the 0 or $2\pi$ boundary, significantly reducing the error. We also observe that their instantiations of $p_\varepsilon, l_{x,\varepsilon}, r_{x,\varepsilon}$ are connected through transformation invariants. For instance, moving from the classical domain $[0, 1)$ to the circular domain $[0, 2\pi)$, $p_\varepsilon = e^{\varepsilon/2}$ transforms to $p_\varepsilon = \frac{1}{2\pi} e^{\varepsilon/2}$, reflecting the ratio of $[0, 1)$ to $[0, 2\pi)$.

**Example 7.** *Figure 5.6 shows two examples of Theorem 8 when $\varepsilon = 1$. For $x = 0$ in the left figure, it samples the output $y$ from $[1.62\pi, 2\pi) \cup [0, 0.38\pi)$ with probability density $0.26$ and from $[0.38\pi, 1.62\pi)$ with probability density $0.09$.*

**MSE analysis.** The MSE of the optimal GPM in Theorem 8 needs a separate analysis due to the circular domain. The biggest difference from the classical domain is that there exist no endpoints, i.e. fixed farthest points, in the circular domain. Without loss of generality, assume $\mathcal{L} := |y - x|^2$ and $x > \pi$, the farthest distance from $x$ is always $\pi$, i.e. from $x$ to $x - \pi$. If we shift the data domain by $\pi - x$, then point $x - \pi$ is mapped to 0. This domain shift does not change the value of $\mathcal{L}$, but $x$ now locates at $\pi$. Therefore, the MSE of the optimal GPM in Theorem 8 has an identical



(a) $x = 0$.



(b) $x = \pi/4$.

Figure 5.6: Optimal GPM (Theorem 8) for the circular domain with $\varepsilon = 1$. Here $[0, 2\pi)$ is wrapped into a circle. Angle values in the red arc (centered at $x$) have a higher sampling pdf.

value for all $x$ in the circular domain, which is

$$\begin{aligned}\mathrm{MSE}[\mathcal{M}(x)] =& \mathrm{MSE}[\mathcal{M}(\pi)] \\ =& 2\left(\int_0^{l^{\mathrm{mod}}_{\pi,\varepsilon}} (y-\pi)^2 \frac{p_\varepsilon}{e^\varepsilon} \mathrm{d}y + \int_{l^{\mathrm{mod}}_{\pi,\varepsilon}}^{\pi} (y-\pi)^2 p_\varepsilon \mathrm{d}y\right).\end{aligned}$$

Calculating the above integral, we can obtain the MSE of the optimal GPM in the circular domain.

**Theorem 9.** *The optimal GPM in Theorem 8 has an identical MSE for all $x$ in the circular domain, which is*

$$\mathrm{MSE}[\mathcal{M}(x)] = \frac{2}{3}\left((\pi^3 - C^3)\frac{p_\varepsilon}{e^\varepsilon} + C^3 p_\varepsilon\right),$$

*where $C = \pi(e^{\varepsilon/2} - 1)/(e^\varepsilon - 1)$.*

*Proof.* We have shown that the MSE at $x$ is the same as that at $\pi$. Then the calculation is straightforward by plugging in the values of $p_\varepsilon$ and $l^{\mathrm{mod}}_{\pi,\varepsilon}$ in Theorem 8. □

If we do not consider $\mathcal{L}_{\mathrm{mod}}$ and directly apply the optimal GPM in the classical domain (or SW and PM mechanisms) to the circular domain, i.e. flatten the circular domain to the classical domain $[0, 2\pi)$ and consider $\mathcal{L}$, the MSE will varies for different $x$. In the flattened domain, the worst-case MSE is at $x = 0$ or $x = 2\pi$ and the best-case MSE is at $x = \pi$. Therefore, the optimal GPM in Theorem 8 always has a lower MSE than "flattened" mechanisms. The evaluation section will further demonstrate this.

## 5.4 Distribution and Mean Estimation

This section applies the proposed mechanisms to support the commonly used distribution and mean estimation. We will show that the proposed mechanisms also provide optimality for these estimation tasks.

Assume a set of users with sensitive data $\mathcal{X} = \{x_1, x_2, \dots, x_n\}$. They apply $\mathcal{M}$ to produce randomized outputs $\mathcal{Y} = \{y_1, y_2, \dots, y_n\}$. The data collector then estimates the distribution and mean of $\mathcal{X}$ using $\mathcal{Y}$. Specifically, the collector uses the values in $\mathcal{Y}$ as it is, i.e. without knowing and applying any post-processing based on prior knowledge of $\mathcal{X}$.

### 5.4.1 Distribution Estimation

To estimate the distribution of $\mathcal{X}$ from $\mathcal{Y}$, the collector must discretize the continuous domain $\mathcal{X}$ into $k$ bins $B_1, B_2, \dots, B_k$. Each bin $B_j$'s probability is then estimated by the proportion of $y_i$ that falls within $B_j$. Probabilities of all bins together form the estimated distribution $\hat{\mathcal{F}}_B$ using $\mathcal{Y}$. This estimation's accuracy is measured by the distance between the estimated distribution $\hat{\mathcal{F}}_B$ and the true distribution $\mathcal{F}_B$ from $\mathcal{X}$ [95].

Note that the bin size can impact the estimation accuracy due to the rounding of the bins. However, it is actually a hyperparameter that does not inherently affect the estimation accuracy if we have a sufficiently large number of bins, as the support of $\hat{\mathcal{F}}_B$ (i.e. non-zero bins) converges to $\mathcal{Y}$ and the support of $\mathcal{F}_B$ converges to $\mathcal{X}$, i.e.

$$\lim_{k\to\infty} \operatorname{supp}(\hat{\mathcal{F}}_B) = \mathcal{Y}, \quad \lim_{k\to\infty} \operatorname{supp}(\mathcal{F}_B) = \mathcal{X}.$$

The proposed mechanisms guarantee the optimal error between $\mathcal{X}$ and $\mathcal{Y}$ for each $x_i$. This ensures their optimality among GPM when applied to distribution estimation.

Some other statistical estimations are special applications of distribution estimation, such as range and quantiles [95] to estimate a specific part of the distribution. The proposed mechanisms have optimality for these estimations as well.

### 5.4.2 Mean Estimation

To estimate the mean of $\mathcal{X}$ from $\mathcal{Y}$, the collector uses the estimator $\hat{\mu} = \sum_{i=1}^{n} y_i/n$. The accuracy of this estimator is measured by $|\hat{\mu} - \mu|$, where $\mu$ is the true mean of $\mathcal{X}$. The proposed mechanisms guarantee the optimal error between each $x_i$ and $y_i$, which in turn leads to the smallest $|\hat{\mu} - \mu|$ among all GPMs under this metric.

Typically, a mean estimator may also need to be unbiased, i.e. $\mathrm{E}[\hat{\mu}] = \mu$. This constraint translates to $\mathrm{E}[y_i] = \mathrm{E}[\mathcal{M}(x_i)] = x_i$.[¶] This is unachievable for same-domain mapping $\mathcal{M} : \mathcal{X} \to \mathcal{X}$ on classical domains, as the endpoints of $\mathcal{X}$ cannot be the mean value (or center) of any distribution over $\mathcal{X}$. For example, for any LDP mechanism $\mathcal{M} : [0, 1) \to [0, 1)$, distribution of $\mathcal{M}(0)$ can not be unbiased. So the mechanism in Theorem 7 is biased.

**Unbiased mean estimation.** Note that an unbiased mean estimator can be achieved by enlarging the output domain $\mathcal{X} \to \mathcal{Y}_\varepsilon$. Mathematically, this involves incorporating the unbiasedness constraint $\mathrm{E}[\mathcal{M}(x)] = x$ into optimization problems for solving $\mathcal{M}$. Following the same optimization process as in the classical domain, we hypothesize that the 3-GPM remains optimal for domain $\mathcal{Y}_\varepsilon$.

**Hypothesis 2.** *For any domain $\mathcal{X} \to \mathcal{Y}_\varepsilon$, where $\mathcal{Y}_\varepsilon$ is a variable w.r.t $\varepsilon$, and under absolute error and square error metrics, the optimal piecewise-based mechanism falls into 3-GPM.*

Hypothesis 2 is a natural extension of Hypothesis 7, as the output domain $\mathcal{Y}_\varepsilon$ becomes explicit once $\varepsilon$ is specified. Under the 3-GPM, an unbiased mechanism $\mathcal{M}$ with a variable output domain $\mathcal{Y}_\varepsilon$ can be analytically derived by incorporating the unbiasedness constraint. As a complement to Theorem 7, we provide Theorem 10 for mean estimation in the classical domain.

**Theorem 10.** *Denote $\mathcal{Y}_\varepsilon = [-C, C+1)$ with $C = (e^{\varepsilon/2}+1)/(e^{\varepsilon/2}-1)$. If Hypothesis 2 holds, then among the unbiased GPM $\mathcal{M} : \mathcal{X} \to \mathcal{Y}_\varepsilon$ (i.e. $\mathrm{E}[\mathcal{M}(x)] = x$), closed form*

$$\mathrm{pdf}[\mathcal{M}(x) = y] = \begin{cases} p_\varepsilon & \text{if } y \in [l_{x,\varepsilon}, r_{x,\varepsilon}), \\ p_\varepsilon / e^\varepsilon & y \in \mathcal{Y}_\varepsilon \setminus [l_{x,\varepsilon}, r_{x,\varepsilon}), \end{cases}$$

*where $p = e^{\varepsilon/2}/(2C+1)$,*

$$l_{x,\varepsilon} = \frac{C+1}{2} \cdot x - \frac{(3C+1)(C-1)}{4C},$$
$$r_{x,\varepsilon} = \frac{C+1}{2} \cdot x + \frac{(C+1)(C-1)}{4C}.$$

*is optimal for $[0, 1) \to \mathcal{Y}_\varepsilon$ and the square error metric.*

*Proof.* Instantiations of $C$, $p$, $l_{x,\varepsilon}$, and $r_{x,\varepsilon}$ are derived by analytical deduction. Appendix C.1.6 proves the unbiasedness. □

In the context of the circular domain, the $\mathcal{M}$ in Theorem 8 is unbiased, as the distribution of $\mathcal{M}$ is always centered at $x$. This property is also illustrated in Figure 5.6.

[¶]Actually, unbiasedness for numerical data is not as important as for categorical data, as it is for a single data point $x_i$. When the dataset $\mathcal{X}$ is not concentrated around a single point, an unbiased mechanism may not necessarily provide better performance.

Table 5.2: Optimal distribution and mean estimation in three domain types under GPM.

| **Domain type** | **Distribution** | **Mean** |
|---|---|---|
| Classical ($\mathcal{X} \to \mathcal{X}$) | Theorem 7 | Theorem 7 (biased) |
| Circular domain | Theorem 8 | Theorem 8 (unbiased) |
| Classical ($\mathcal{X} \to \mathcal{Y}_\varepsilon$) | – | Theorem 10 (unbiased) |

Table 5.2 summarizes the optimal GPM for both distribution and mean estimation in three domain types. Theorems use a concrete domain $\mathcal{X}$ for the sake of clarity. We do not give closed-form optimal GPM for distribution estimation on $\mathcal{X} \to \mathcal{Y}_\varepsilon$ because $\mathcal{Y}_\varepsilon$ can not be easily concretized by constraints as in mean estimation.

## 5.5 Discussion and Extension

**Minimize Error at a Specific $x$**

The proposed optimal GPMs are designed to minimize the worst-case error over the whole domain. However, the framework can be used to minimize the error at any specific $x$. This is useful when the data distribution is concentrated around a specific data point.

Formally, assume the data distribution is concentrated around $x_0$, and we want to minimize the error at $x_0$. The optimization problem in Lemma 1 for solving $p_i$ can be modified to

$$\min_{p_i, l_{i,x}, r_{i,x}} \int_{\mathcal{Y}} \mathcal{L}(y, x_0)\mathcal{P}_{\mathcal{M}(x_0)}\mathrm{d}y.$$

This optimization problem generally leads to a different $p_i$ from the optimal GPM for the worst-case error.

**Example 8.** *Assume $\mathcal{X} \to \mathcal{Y} = [0, 1) \to [0, 1)$, $\mathcal{L} := |y - x|$, and the data distribution is concentrated around $x_0 = 0.2$. Solving the above optimization problem with $\varepsilon = 1$ gives probability (of the second piece) $p_2 = 1.54$ and $Err(x_0) = 0.241$. In contrast, the optimal GPM for the worst-case error uses $p_2 = 1.64$, as shown in Figure 5.4, which gives $Err(x_0) = 0.243$.*

Importantly, optimizing for a specific $x_0$ does not leak information about $x_0$, as the mechanism (i.e. $p_i$, $l_{i,x}$, and $r_{i,x}$) still contains no information about $x_0$. Moreover, observing $Err(x)$ at all $x$ does not reveal $x_0$, as the error at $x_0$ is not necessarily the smallest. Therefore, the adversary cannot infer $x_0$ from observing the mechanism.

**An Extension: 2D Polar Coordinates**

Polar coordinates are widely used in relative location representation, e.g. navigation systems that have the locations of surrounding objects relative to it. The proposed optimal GPM for the classical and circular domain can be combined and extended to polar coordinates for collecting such data under LDP.

**Privacy.** A polar coordinate data is represented by a 2D tuple $(x_1, x_2) \in [0, d) \times [0, 2\pi)$, where $x_1$ is the distance from the pole and $x_2$ is the angle from the polar axis. The first dimension is linear, while the second is naturally circular, thus we can combine the optimal GPM for both domains to provide LDP for such data.

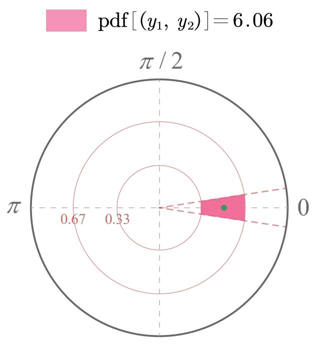
**Utility.** The proposed mechanisms guarantee the optimal error for each dimension. Therefore, if we use $\mathcal{L}_{\text{2D}} \coloneqq \mathcal{L}(y_1, x_1) + \mathcal{L}_{\text{mod}}(y_2, x_2)$ as the error metric for the polar coordinate data and optimally assign the privacy parameter $\varepsilon$ to each dimension, the optimal GPM preserves the optimal error.

**Optimal assignment of $\varepsilon$.** Formally, we want to assign $\varepsilon = \varepsilon_1 + \varepsilon_2$ to $x_1$ and $x_2$ respectively to minimize the worst-case error in 2D polar coordinates. This error is the sum of the worst-case error for $x_1$ and $x_2$ under $\mathcal{L}$ and $\mathcal{L}_{\text{mod}}$ respectively. Therefore, the optimal assignment of $\varepsilon$ can be derived by solving

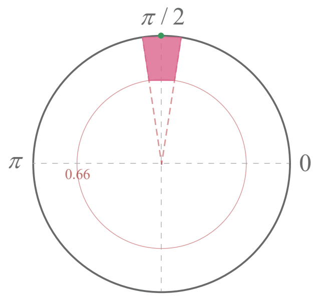
$$\min_{\varepsilon_1, \varepsilon_2} Err_{1,\text{wor}}(\varepsilon_1) + Err_{2,\text{wor}}(\varepsilon_2),$$

where $Err_{1,\text{wor}}(\varepsilon_1)$ and $Err_{2,\text{wor}}(\varepsilon_2)$ are the worst-case error for the classical domain $[0, d)$ and the circular domain $[0, 2\pi)$ respectively. Closed-form $Err_{1,\text{wor}}(\varepsilon_1)$ and $Err_{2,\text{wor}}(\varepsilon_2)$ can be derived from instantiations of the mechanism and the error metric. Then the above optimization problem gives the optimal $\varepsilon_1$ and $\varepsilon_2$. Appendix C.2.6 provides details for solving this optimization problem.

**Example 9.** *Figure 5.7 shows two examples of the optimal GPM for 2D polar coordinates in* $[0, 1) \times [0, 2\pi)$ *with* $\varepsilon = 1 + 2\pi$*. The green point represents the sensitive data, the optimal GPM samples the output from the pink area with a higher probability.*



(a) $(x_1, x_2) = (0.5, 2\pi)$.



(b) $(x_1, x_2) = (1, \pi/2)$.

Figure 5.7: Optimal GPM for 2D polar coordinates when $\varepsilon = 1 + 2\pi$ and $\mathcal{L} = |y - x|^2$ under $\mathcal{L}_{\text{2D}} \coloneqq \mathcal{L}(y_1, x_1) + \mathcal{L}_{\text{mod}}(y_2, x_2)$.

## 5.6 Evaluations

This section evaluates the theoretical and experimental data utility of the proposed methods by comparing them with existing instantiations and their variants:

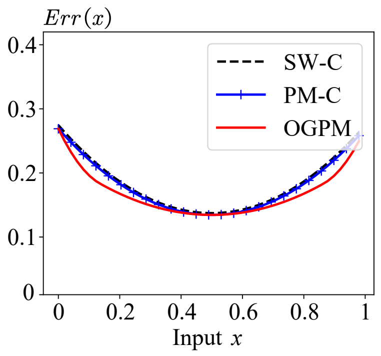
- OGPM: closed-form optimal GPM (Theorem 7 and 8 for the classical and circular domain, respectively).
- OGPM-U: unbiased closed-form optimal GPM (Theorem 10) for mean estimation in the classical domain.
- PM [150], SW [95], and their post-processed versions: PM is the first TPM designed for mean estimation, while SW is designed for distribution estimation. Both mechanisms output enlarged domains but can be post-processed by truncating outputs to the input domain. These post-processed versions are referred to as T-PM and T-SW for convenience.
- PM-C and SW-C: the compressed versions of PM and SW for $\mathcal{X} \to \mathcal{X}$. For the best potential of PM and SW, we adapt them to $\mathcal{X} \to \mathcal{X}$ as PM-C and SW-C by linearly compressing their output domain $\mathcal{Y}_\varepsilon$ to $\mathcal{X}$, i.e. by transformation invariants, which maintains the privacy parameter.

We also compare OGPM's expected error with non-piecewise-based mechanisms that can be applied to bounded numerical domains:

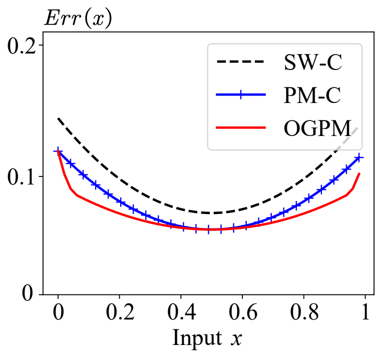
- Variants of the Laplace mechanism: including the staircase mechanism [60], Laplace mechanism with post-processing by truncation (T-Laplace), and the bounded Laplace mechanism (B-Laplace) [73], which redesigns a bounded Laplace-shape distribution.
- Purkayastha mechanism [160]: a mechanism for directional data on spheres $\mathbb{S}^{n-1}$. When $n = 2$, it is a counterpart of OGPM in the circular domain.

We omit comparison with PTT [101] because it does not provide a concrete method to find a closed-form mechanism. We use $\mathcal{X} = [0, 1)$ as the classical domain; this does not change their data utility ordering by the transformation invariant. PM and SW can only be applied to the classical domain, so when evaluating them in the circular domain, we "flatten" the circular domain to the classical domain $[0, 2\pi)$ and apply them to the flattened domain.

The first subsection presents the comparison of expected errors, followed by the distribution and mean estimations on real-world datasets in the second subsection.



(a) Privacy parameter $\varepsilon = 2$.



(b) Privacy parameter $\varepsilon = 4$.

Figure 5.8: Whole-domain error comparison in the classical domain with error metric $\mathcal{L} = |y - x|$.

### 5.6.1 Expected Errors

GPM's data utility under distance metric $\mathcal{L}$ is measured by the expected error $Err(x)$ in Formula (5.1):

$$Err(x) = \int_{\mathcal{Y}} \mathcal{L}(y, x)\mathcal{P}_{\mathcal{M}(x)}\mathrm{d}y,$$

where $x$ is the sensitive input and $y$ is the output of $\mathcal{M}$. Based on $Err(x)$, two types of error need to be considered:

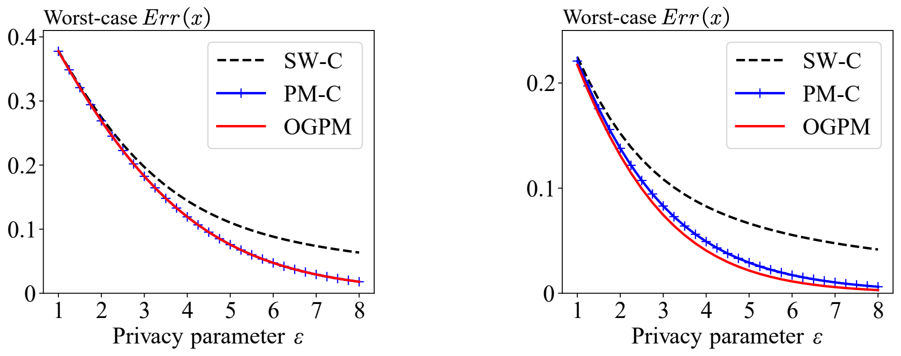
- **whole-domain error:** $Err(x)$ values for the whole domain, i.e. for all $x \in \mathcal{X}$.
- **worst-case error:** the largest $Err(x)$ value among the whole domain. PM [150] and PTT [101] also use this error to evaluate data utility.

We have proved that the worst-case error of the classical domain is from the endpoints, so this error is actually the error at $x = 0$ and $x = 1$ for the classical domain. For the circular domain, the worst-case error is at $x = \pi$.

#### Comparison with PM-C and SW-C

PM-C and SW-C exhibit the best potential of PM and SW, so we compare OGPM with them in the first place. The comparisons are conducted under the classical domain and the circular domain, respectively.

**Classical Domain.** Figure 5.8 shows the comparison of the whole-domain error in the classical domain. We use distance metric $\mathcal{L} = |y - x|$ and set $\varepsilon = 2$ and $\varepsilon = 4$ for the comparison. It can



(a) Distance metric $\mathcal{L} = |y - x|$.

(b) Distance metric $\mathcal{L} = |y - x|^2$.

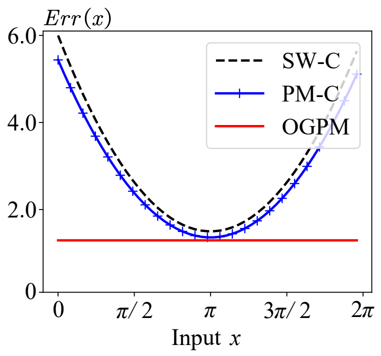
Figure 5.9: Worst-case error comparison in the classical domain.

be seen that OGPM consistently has the lowest error across all $x$ values. For all mechanisms, the expected error achieves the maximal value at the endpoints and the minimal value at the midpoint. At small $\varepsilon$ values, their errors are similar due to the strong randomness (privacy constraint). At larger $\varepsilon$ values, OGPM's error has a significant advantage over PM-C and SW-C, especially between the endpoints and the midpoint. Statistically, under $\mathcal{L} = |y - x|$ with $\varepsilon = 2$, OGPM's average error is 94.2% of PM-C and 92.3% of SW-C across the whole domain. When $\varepsilon = 4$, OGPM's average error is 90.5% of PM-C and 74.7% of SW-C. More comparisons under smaller $\varepsilon$ values are provided in Appendix C.2.7.

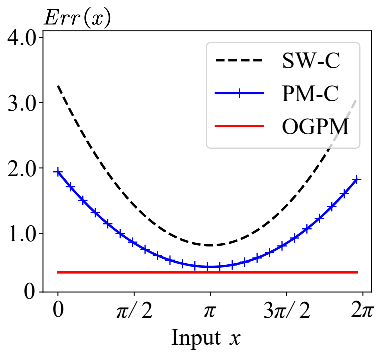
Figure 5.9 shows the comparison of the worst-case error w.r.t $\varepsilon$ in the classical domain. The error is measured by two distance metrics: $\mathcal{L} = |y - x|$ and $\mathcal{L} = |y - x|^2$. It can be seen that the error of all mechanisms decreases with $\varepsilon$, but OGPM and PM-C decreases faster than SW-C. For $\mathcal{L} = |y - x|$, OGPM has almost the same worst-case error as PM-C; for $\mathcal{L} = |y - x|^2$, OGPM's error is slightly smaller than PM-C. For both metrics, OGPM's worst-case error is the lowest across all $\varepsilon$ values. Statistically, under $\mathcal{L} = |y - x|^2$, OGPM's average error is 89.9% of PM-C and 61.7% of SW-C.

**Circular Domain.** Figure 5.10 shows the comparison of the whole-domain error in the circular domain. We use distance metric $\mathcal{L} = |y - x|^2$ and set $\varepsilon = 2$ and $\varepsilon = 4$ for the comparison. It can be seen that OGPM consistently has the lowest error across all $x$ values, and the error is stable across $x$, which is consistent with the theoretical analysis in Theorem 9. For PM-C and SW-C, which treat the circular domain as the classical domain, their errors vary with $x$ and are higher than OGPM's error, especially near the endpoints. Statistically, under $\mathcal{L} = |y - x|^2$ with $\varepsilon = 2$, OGPM's average error is 47.5% of PM-C and 43.0% of SW-C across the whole domain. When $\varepsilon = 4$, OGPM's average error is 41.3% of PM-C and 24.6% of SW-C.

Figure 5.11 shows the comparison of the worst-case error w.r.t $\varepsilon$ in the circular domain. The error is measured by two distance metrics: $\mathcal{L} = |y - x|$ and $\mathcal{L} = |y - x|^2$. Similar to the classical domain,

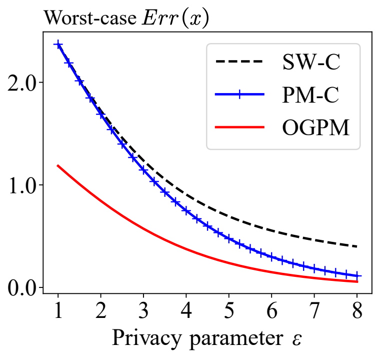


(a) Privacy parameter $\varepsilon = 2$.

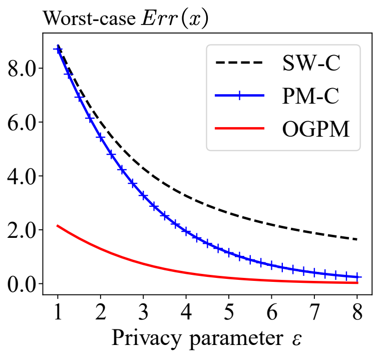


(b) Privacy parameter $\varepsilon = 4$.

Figure 5.10: Whole-domain error comparison in the circular domain with error metric $\mathcal{L} = |y - x|$.



(a) Distance metric $\mathcal{L} = |y - x|$.



(b) Distance metric $\mathcal{L} = |y - x|^2$.

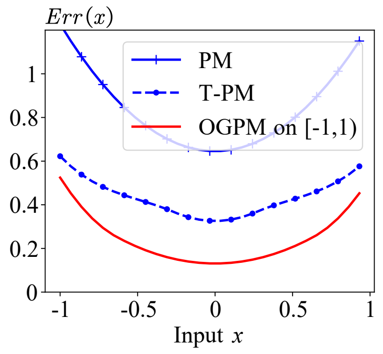
Figure 5.11: Worst-case error comparison in the circular domain.

OGPM has the lowest worst-case error across all $\varepsilon$ values, and the advantage is more significant, especially for small $\varepsilon$ values. Statistically, under $\mathcal{L} = |y - x|$, OGPM's average error is 50.0% of PM-C and 41.6% of SW-C across the range of $\varepsilon$. Under $\mathcal{L} = |y - x|^2$, OGPM's average error is 22.4% of PM-C and 15.4% of SW-C.

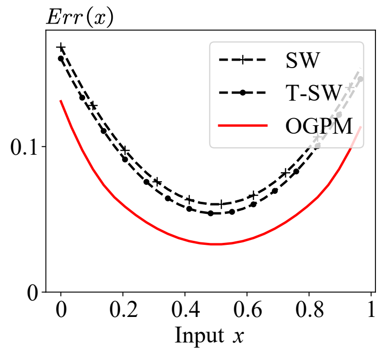
The above comparisons show that OGPM has the lowest expected error in both the classical and circular domains. The advantage of OGPM is more significant in the circular domain, where the error is stable across $x$.

### Comparison with PM and SW

Figure 5.12 presents the comparison of the whole-domain error in the classical domain for the original PM and SW mechanisms, along with their post-processed versions, T-PM and T-SW. For a fair comparison, OGPM is adapted to the domain $\mathcal{X} = [-1, 1)$ to match PM's design, while

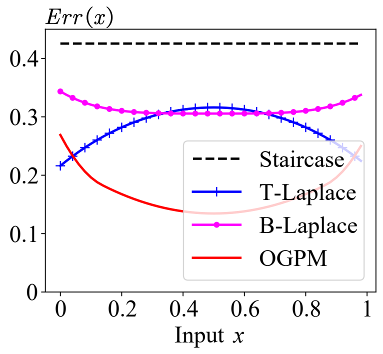


(a) Comparison with PM.

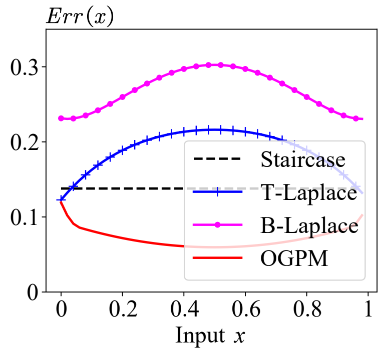


(b) Comparison with SW.

Figure 5.12: Whole-domain error comparison with PM and SW on their data domains (i.e. $\mathcal{X} = [-1, 1)$ and $\mathcal{X} = [0, 1)$, respectively) when $\varepsilon = 2$.



(a) Privacy parameter $\varepsilon = 2$.



(b) Privacy parameter $\varepsilon = 4$.

Figure 5.13: Whole-domain error comparison with the staircase mechanism [60], T-Laplace and B-Laplace mechanisms [73] in the classical domain with error metric $\mathcal{L} = |y - x|$.

SW and OGPM remain consistent with $\mathcal{X} = [0, 1)$. The post-processing of PM and SW involves truncating their outputs in the enlarged domain to the input domain, i.e. applying $\mathcal{I} \circ \mathcal{M}(x)$, where $\mathcal{I} : \mathcal{Y} \to \mathcal{X}$ is the truncation operator. We use the distance metric $\mathcal{L} = |y - x|^2$ and set $\varepsilon = 2$ for the comparison among these five mechanisms. It can be observed that OGPM consistently achieves the lowest error across all $x$ values, with a more significant advantage compared to the comparison with PM-C and SW-C. This is because the original PM and SW output larger domains, resulting in higher errors. Meanwhile, T-PM reduces the error of PM more effectively than T-SW reduces the error of SW, as the original PM has a more enlarged output domain than SW, making truncation more impactful. This comparison highlights OGPM's error advantage over the original PM, SW, and their post-processed versions when applied to their respective data domains.

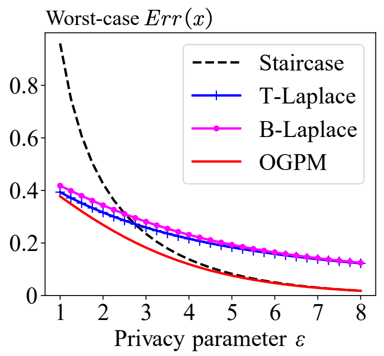


Figure 5.14: Worst-case error comparison (continued from Figure 5.13).

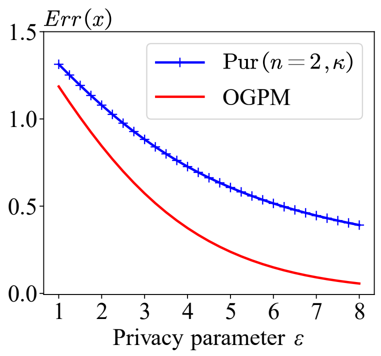


Figure 5.15: Comparison with the Purkayastha mechanism [160] for sphere $\mathbb{S}^{n-1}$.

### Comparison with the Staircase Mechanism, T-Laplace, and B-Laplace

In addition to piecewise-based mechanisms, the Laplace mechanism and its variants can also be applied to the classical domain to achieve LDP. Among these, the staircase mechanism [60] claims to be optimal under certain assumptions. For the input domain $\mathcal{X} = [0, 1)$ (i.e. sensitivity $\Delta = 1$) and error metric $\mathcal{L} = |y - x|$, its expected error is given by Theorem 3 in [60]: $e^{\varepsilon/2}/(e^{\varepsilon} - 1)$. Another approach involves using the Laplace mechanism with truncation [73], referred to here as T-Laplace for convenience. T-Laplace preserves the privacy guarantees of the Laplace mechanism while reducing the expected error, particularly for data points near the endpoints or for small $\varepsilon$ values. Additionally, the bounded Laplace mechanism (B-Laplace) [73] introduces a redesigned bounded Laplace-shaped distribution tailored for bounded domains.[‖]

Figure 5.13 compares the whole-domain error in the classical domain $\mathcal{X} = [0, 1)$ for the staircase mechanism, T-Laplace, and B-Laplace. These mechanisms exhibit distinct error patterns across the domain. For the staircase mechanism, the error remains constant, as it is determined by a fixed staircase distribution and is independent of $x$. For T-Laplace, the error reaches its maximum at the midpoint and its minimum at the endpoints, as truncation favors the endpoints. For instance, when $x = 0$, it is error-free with a probability of $1/2$, due to the symmetry of the Laplace distribution around 0. For B-Laplace, the error trend varies with $\varepsilon$. When $\varepsilon = 2$, the error decreases with $x$ and reaches its minimum at the midpoint, whereas for $\varepsilon = 4$, the error increases with $x$ and peaks at the midpoint. Despite these differing error patterns, OGPM generally achieves lower errors than the staircase mechanism, T-Laplace, and B-Laplace across the whole domain.

Figure 5.14 compares the worst-case error w.r.t. $\varepsilon$ in the classical domain. OGPM consistently

[‖]Appendix C.2.8 provides details on the expected error of B-Laplace.

achieves the lowest worst-case error across all $\varepsilon$ values. For small $\varepsilon$, T-Laplace and B-Laplace exhibit a significant advantage over the staircase mechanism; however, this advantage diminishes as $\varepsilon$ increases. At larger $\varepsilon$ values, the error of the staircase mechanism approaches that of OGPM.

#### Comparison with the Purkayastha Mechanism

The paper "Differential Privacy for Directional Data" [160] introduces two mechanisms for data on spheres $\mathbb{S}^{n-1}$: the VMF mechanism (ensuring indistinguishability of any two points with distance *through* the sphere) and the Purkayastha mechanism (ensuring indistinguishability of any two points with distance *along* the sphere). When $n = 2$, the sphere $\mathbb{S}^1$ corresponds to a circle, making the Purkayastha mechanism a counterpart of OGPM in the circular domain. Therefore, we compare them in the circular domain.[**]

Figure 5.15 presents the comparison of the expected error in the circular domain between OGPM and the Purkayastha mechanism. The expected error of the Purkayastha mechanism is derived using the closed-form expressions in Theorem 19 and 22 of [160], with $\kappa = \varepsilon/\Delta_{\measuredangle}$. Since the errors of both mechanisms are $x$-independent in the circular domain, it suffices to compare their worst-case errors. The results demonstrate that OGPM consistently outperforms the Purkayastha mechanism, achieving significantly lower errors.

### 5.6.2 Distribution and Mean Estimations

This section compares the experimental data utility of the mechanisms in distribution and mean estimations.

#### Setup

We choose the MotionSense dataset [1, 105] for the evaluation. It contains smartphone sensor data recorded during various human activities. Specifically, we use the data from the first three files, encompassing a total of 6 159 data entries. We focus on two types of sensors:

[**]We omit the comparison with the VMF mechanism also because (i) it has been shown that the Purkayastha mechanism outperforms the VMF mechanism (with the same sensitivity $\Delta_{\measuredangle} = \pi$ for sphere $\mathbb{S}^1$, e.g. Figure 5 and 10 in [160]), and (ii) the expected error of the VMF mechanism lacks a closed-form expression (Theorem 17 in [160]), making it complex to compute.

- Accelerometer (linear data): We normalize the dataset to $[0, 1)$ for the classical domain.
- Attitude sensor (angular data): We use this dataset for the circular domain.

Upon applying each LDP mechanism to these datasets, we evaluate the accuracy of distribution and mean estimations on them. For distribution estimation, we divide the domain into $k = 50$ bins and compute the distance between the estimated distribution ($\hat{\mathcal{F}}_B$) and the true distribution ($\mathcal{F}_B$) by summing the absolute difference of each bin's value. Formally,

$$\mid \hat{\mathcal{F}}_B - \mathcal{F}_B \mid := \sum_{i=1}^{50} \mid \hat{\mathcal{F}}_{B_i} - \mathcal{F}_{B_i} \mid,$$

where $\hat{\mathcal{F}}_{B_i}$ and $\mathcal{F}_{B_i}$ are the $i$-th bin's value of the estimated and true distributions, respectively. Although this approach cannot capture the property of the circular data, it remains the most viable metric for comparing circular distributions [55, 106]. Under a more relevant approach, the performance of OGPM for the circular domain could be even better.

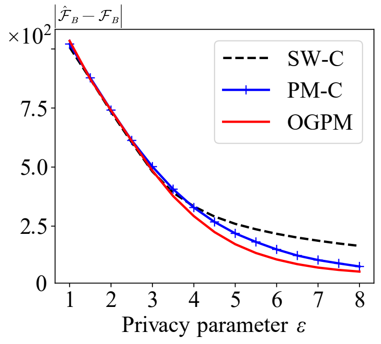
For mean estimation, we compute the absolute difference between the estimated and true mean, i.e. $|\hat{\mu} - \mu|$, where $\hat{\mu}$ is the estimated mean and $\mu$ is the true mean in the classical domain or the circular domain. In the classical domain, the true mean is $\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$. In the circular domain, the mean is computed by the circular mean formula [55, 106]:

$$\mu = \operatorname{atan2}\left(\frac{1}{n}\sum_{i=1}^{n}\sin x_i, \frac{1}{n}\sum_{i=1}^{n}\cos x_i\right).$$

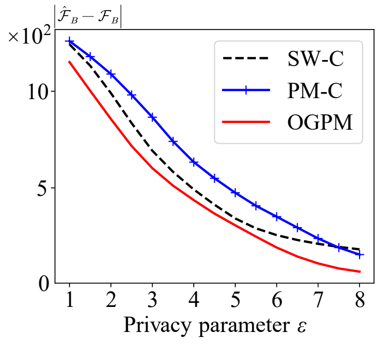
We repeat the experiments 500 times for stable results and report the average error.

**Distribution Estimation**

Figure 5.16 shows the comparison of the errors of distribution estimation in the classical and circular domains. We can see that OGPM outperforms SW and PM with smaller errors in both types of domains. In the classical domain, OGPM's error decreases faster when $\varepsilon$ increases above 3, and SW-C decreases slower than PM-C, consistent with the expected error comparison. In the circular domain, OGPM's error is significantly lower than SW-C and PM-C, despite the limitation of the distance metric used for circular distributions. We also observe that SW-C performs better than PM-C in this domain. This is because SW has higher sampling probabilities for both the central piece and other pieces, making it sample the true value more frequently when $\varepsilon$ is large in practice in a large-size domain, despite the large expected error theoretically. Statistically, OGPM's distribution error is 93.5% of PM-C and 86.7% of SW-C in the classical domain, and 72.2% of PM-C and 84.0% of SW-C in the circular domain.

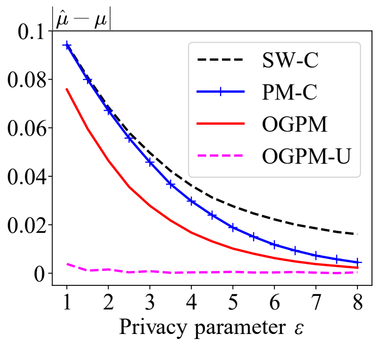


(a) Classical domain.

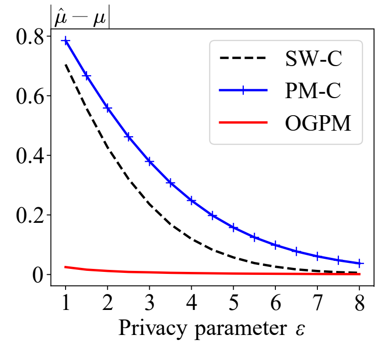


(b) Circular domain.

Figure 5.16: Comparison of distribution estimation error.



(a) Classical domain.



(b) Circular domain.

Figure 5.17: Comparison of mean estimation error.

### Mean Estimation

Figure 5.17 shows the comparison of the errors of mean estimation in the classical and circular domains. OGPM consistently outperforms other mechanisms in both types of domains, with significantly lower errors. In the classical domain, we also compare with OGPM-U, which is specifically designed for unbiased mean estimation. Since the Accelerometer dataset is concentrated around zero, it particularly favors unbiased mechanisms, as their outputs tend to average closely to the true mean. We can see that OGPM-U achieves significantly lower errors than OGPM when $\varepsilon$ is small. In the circular domain, OGPM is unbiased, having a negligible mean estimation error. We also observe that SW-C outperforms PM-C in this large-scale domain. Statistically, OGPM's mean estimation error is 66.2% of PM-C and 55.4% of SW-C in the classical domain. In the circular domain, OGPM's mean estimation error is merely 2.3% of PM-C and 3.6% of SW-C.

## 5.7 Related Work

This chapter focuses on the optimal mechanism for collecting numerical data with bounded domains under LDP, related to numerical data collection and the optimality of DP mechanisms.

### Numerical Data Collection under LDP

Classical noise-adding mechanisms such as the Laplace and Gauss mechanisms [47,48] add randomly sampled noise from a distribution to the data to achieve LDP. However, they generate outputs in an unbounded domain due to the unbounded noise distributions, rendering them unsuitable for applications requiring bounded domains [150].

For bounded data domains, the basic idea is to sample the output from a carefully designed distribution on the bounded domain. Duchi et al. randomize any data in $[-1, 1]$ to two discrete values $y \in \{-C_\varepsilon, C_\varepsilon\}$ [44], where $C_\varepsilon$ is a constant depending on the privacy parameter $\varepsilon$. This binary-output mechanism exhibits a large randomization error as the output space is too coarse. PM [150] extends the binary output to a continuous output in $[-C_\varepsilon, C_\varepsilon]$. It designs a piecewise-based mechanism to sample the output, which uses different sampling intervals for different $x$, achieving a lower randomization error. Both PM and later SW [95] have shown the data utility advantage of TPM in numerical data collection with bounded domains under LDP. PTT [101] discusses the optimality of TPM. It shows that there exist TPM instantiations that yield optimal data utility, but it does not provide the closed-form mechanism. TPM is a special case of GPM using 3-piece distributions and a specific form for each piece. Meanwhile, existing TPM instantiations are designed for specific error metrics and only classical domains.

**Applications of TPM.** A natural application of TPM is in high-dimensional numerical data. This includes scenarios where the sensitive data may be a high-dimensional vector [150, 175], an infinite data stream [130], or matrixes [157]. Another application area is federated learning. TPM ensures LDP in the $[0, 1]$ domain, which is commonly used as the normalized domain in model training. TPM avoids the data clipping required by noise-adding mechanisms [4,57,96]. Another typical application is in sensors, where the data is bounded by the sensor's physical nature. This dissertation can replace the existing TPM to achieve better utility.

A recent work aims to design other types of bounded distributions besides TPM to achieve (L)DP [178]. They tailor sin functions and quartic functions on the bounded domain to satisfy the DP constraint. From their experimental results, the piecewise-based design is the best choice among their six instantiations for bounded domains under DP. This also indicates the advantages of piecewise-based mechanisms in the bounded domain.

**Optimality of DP Mechanisms**

Achieving optimal data utility is a common concern across all DP mechanisms. The staircase mechanism showed that the Laplace mechanism does not generate optimal noise [60,141]. It samples noise from a staircase distribution, which has been shown to achieve lower error compared to the Laplace mechanism. Besides optimality in concrete error values (which is the focus of this dissertation), another widely focused concept is asymptotic optimality [19, 101], which studies optimal asymptotic bounds of a mechanism or a statistical estimation. Appendix C.2.2 discusses more detailed optimalities.

Beyond the optimality of a single DP procedure, the optimality of multifold compositions such as iterative training under DP, is studied by advanced compositions [37,38,79,117]. In high-dimensional settings, the sensitivity set across dimensions also influences the optimality [168], because it affects the choice of the privacy parameter. These works either focus on different privacy constraints or discuss optimality beyond a single DP procedure, making them orthogonal to this dissertation.

## 5.8 Conclusions

This chapter presents the optimal piecewise-based mechanism for collecting numerical data with bounded domains under LDP. To find the optimal mechanism among all possible piecewise mechanisms, this chapter generalizes the existing 3-piece mechanism to an $m$-piece mechanism with the most general form. It proposed a framework that combines analytical proofs and off-the-shelf optimization solvers to find the optimal mechanism. The results include the closed-form optimal piecewise mechanisms for both the classical and circular domain. Theoretical and experimental evaluations confirm the advantages of the proposed mechanisms over existing mechanisms.

# Chapter 6

# Trajectory Collection in Continuous Space under LDP

## 6.1 Preliminaries

This section formulates the problem and reviews three trajectory collection methods under LDP, along with their shortcomings in continuous spaces, which motivates the necessity of designing new LDP methods for trajectory collection in continuous spaces.

### 6.1.1 Problem Formulation

A typical trajectory collection schema consists of a set of trajectories from users and one collector. Each trajectory $\mathcal{T}$ is a sequence of locations $\mathcal{T} = \{\tau_1, \tau_2, \ldots, \tau_n\}$, where $\tau_i \in \mathcal{S}$ and $\mathcal{S} \subset \mathbb{R}^2$ is a continuous and bounded domain. The collector needs to collect these trajectories to provide analysis or location-based services.

However, the collector is untrusted and may act as an adversary attempting to infer users' sensitive data. Therefore, releasing the sensitive trajectories to the collector poses a privacy risk. To protect privacy, users perturb their trajectories using a privacy mechanism $\mathcal{M} : \mathcal{S} \rightarrow Range(\mathcal{M})$, and then send the perturbed trajectories $\mathcal{T}' = \{\tau_1', \tau_2', \ldots, \tau_n'\}$ to the collector.

This chapter aim to design a mechanism $\mathcal{M}$ that satisfies $\varepsilon$-LDP, providing provable privacy guarantee for sensitive trajectories. Additionally, $\mathcal{M}$ should preserve utility, ensuring perturbed trajectories remain comparable to the originals, while incurring only modest computational overhead suitable for real-time applications.

### 6.1.2 Local Differential Privacy

**Definition 5** ($\varepsilon$-LDP [42])**.** *A perturbation mechanism $\mathcal{M} : \mathcal{X} \to Range(\mathcal{M})$ satisfies $\varepsilon$-LDP, if for two arbitrary input $x_1$ and $x_2$, the probability ratio of outputting the same $y$ is bounded:*

$$\forall x_1, x_2 \in \mathcal{X}, \forall y \in Range(\mathcal{M}) : \frac{\Pr[\mathcal{M}(x_1) = y]}{\Pr[\mathcal{M}(x_2) = y]} \leq \exp(\varepsilon).$$

If $\mathcal{M}(x)$ is continuous, the probability $\Pr[\cdot]$ is replaced by probability density function (*pdf*). Intuitively, Definition 5 represents the difficulty of distinguishing $x_1$ and $x_2$ given $y$. Lower values of the privacy parameter $\varepsilon \in [0, +\infty)$ mean higher privacy. For example, $\varepsilon = 0$ requires $\mathcal{M}$ to map two arbitrary inputs to any output $y$ with the same probability, thus the perturbed output contains no distribution information of the sensitive input, making any hypothesis-testing method to infer the sensitive $x$ powerless.

**Theorem 11** (Sequential Composition of LDP [48, 181])**.** *Let $\mathcal{M}_1$ and $\mathcal{M}_2$ be two mechanisms that satisfy $\varepsilon_1$ and $\varepsilon_2$-LDP, respectively. Their composition, defined as $\mathcal{M}_{1,2} \coloneqq (\mathcal{M}_1, \mathcal{M}_2) : (\mathcal{X}_1, \mathcal{X}_2) \to (Range(\mathcal{M}_1), Range(\mathcal{M}_2))$, satisfies $(\varepsilon_1 + \varepsilon_2)$-LDP.*

Sequential composition enables (i) designing LDP mechanisms for 2D data by composing two 1D mechanisms, and (ii) perturbing multiple locations in a trajectory (i.e. repeated composition) while still providing an overall LDP guarantee.

#### Piecewise-based Mechanism

When the input domain $\mathcal{X}$ is continuous and bounded, state-of-the-art LDP mechanisms are often *piecewise-based* [95, 150, 184]. They generate an output by sampling from a probability distribution over $\mathcal{X}$ whose density is defined in a piecewise manner. Below, we present a unified formulation that captures this family of mechanisms.

**Definition 6** (Piecewise-based mechanism)**.** *A piecewise-based mechanism $\mathcal{M} : \mathcal{X} \to Range(\mathcal{M})$ is a family of probability distributions that, given input $x \in \mathcal{X}$, outputs $y \in Range(\mathcal{M})$ according to*

$$pdf[\mathcal{M}(x) = y] = \begin{cases} p_\varepsilon & \text{if } y \in [l_{x,\varepsilon}, r_{x,\varepsilon}), \\ p_\varepsilon / \exp(\varepsilon) & \text{otherwise,} \end{cases}$$

*where $p_\varepsilon$ is the sampling probability w.r.t. $\varepsilon$, $[l_{x,\varepsilon}, r_{x,\varepsilon})$ is the sampling interval w.r.t. $x$ and $\varepsilon$. $Range(\mathcal{M}) \supseteq \mathcal{X}$ is also a continuous and bounded domain.*

Piecewise-based mechanisms sample an output $y$ with a high probability $p_\varepsilon$ from the interval $[l_{x,\varepsilon}, r_{x,\varepsilon})$, and with a lower probability from the remaining two pieces, while still satisfying $\varepsilon$-LDP. Representative instantiations include OGPM [184] and PM [150] for mean estimation, and SW [95] for distribution estimation. This chapter's perturbation mechanisms build on the 1D piecewise-based mechanism in OGPM to design new 2D mechanisms tailored to trajectory collection.

### *k*-RR and Exponential Mechanism

If $\mathcal{X}$ is discrete, especially when $|\mathcal{X}|$ is not large [153], a classical LDP mechanism is $k$-RR (or GRR in some literature) [78].

**Definition 7.** *$k$-RR is a sampling mechanism $\mathcal{M} : \mathcal{X} \to \mathcal{X}$ that, given $|\mathcal{X}| = k$ and input $x \in \mathcal{X}$, outputs $y \in \mathcal{X}$ according to*

$$\Pr[\mathcal{M}(x) = y] = \begin{cases} \dfrac{\exp(\varepsilon)}{|\mathcal{X}| - 1 + \exp(\varepsilon)} & \textit{if } y = x, \\ \dfrac{1}{|\mathcal{X}| - 1 + \exp(\varepsilon)} & \textit{otherwise.} \end{cases}$$

$k$-RR outputs the truth $x$ with a higher probability or outputs other values with a lower probability, while satisfying $\varepsilon$-LDP.

When there is a semantic distance (score) function between elements in $\mathcal{X}$, the Exponential mechanism [121] is more widely used. Unlike $k$-RR, which treats other values (except $x$) equally, the Exponential mechanism leverages a score function to assign different probabilities. $k$-RR is a special case of the Exponential mechanism when the score function is a binary indicator function.

**Definition 8** (Exponential Mechanism [121])**.** *Given a score function $d : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$, a privacy parameter $\varepsilon > 0$, and a set of possible outputs $\mathcal{Y}$, the Exponential mechanism $\mathcal{M} : \mathcal{X} \to \mathcal{Y}$ is defined by:*

$$\Pr[\mathcal{M}(x) = y] = \frac{\exp\left(\frac{\varepsilon d(x,y)}{2\Delta d}\right)}{\sum_{y' \in \mathcal{Y}} \exp\left(\frac{\varepsilon d(x,y')}{2\Delta d}\right)},$$

*where $\Delta d = \max_{x,y,y' \in \mathcal{Y}} |d(x,y) - d(x,y')|$ is the sensitivity of the score function $d$.*

In the context of trajectory collection, common score functions include the negative Euclidean distance $d(x,y) := -||x-y||_2$, great circle distance [181], etc. Under such score functions (e.g. $d(x,y) = -\|x-y\|_2$), locations closer to the true location $x$ have larger scores $d(x,y)$ and thus receive higher sampling probabilities, making them more likely to be selected.

Table 6.1: Comparison with existing methods.

| **Method** | **Main technique** | **LDP guarantee** |
|---|---|---|
| NGram [34] | Hierarchical decomposition | |
| L-SRR [148] | Staircase RR mechanism | Discrete space |
| ATP [181] | Direction perturbation | |
| TraCS | Direction[a] & coordinate perturbation | Continuous space |

[a] Designed for continuous direction space (different from ATP)

**Privacy Parameter for Trajectory Data**

When applying LDP to trajectory data, there are two strategies for setting the privacy parameter. (i) Treating each location $\tau_i$ in a trajectory $\mathcal{T}$ as a data item and applying an $\varepsilon$-LDP mechanism to perturb each location; (ii) Treating the entire trajectory $\mathcal{T}$ as a data item and applying an $\varepsilon$-LDP mechanism to perturb the whole trajectory.* The first strategy is cleaner and convertible to the second strategy via the Sequential Composition Theorem 11, whereas the second depends on the trajectory length, which complicates the analysis and the design of mechanisms under a fixed $\varepsilon$. Therefore, we primarily adopt the first strategy in this dissertation, and include experiments for the second in Appendix D.2.10 for completeness.

### 6.1.3 Existing Methods

Table 6.1 summarizes three state-of-the-art trajectory collection methods under pure LDP, along with our approach. Other works that emphasize external knowledge rather than new perturbation mechanisms are discussed in Appendix 6.4.

NGram [34] hierarchically decomposes the physical space $\mathcal{S}$ into fine-grained discrete regions with semantic labels. Concretely, $\mathcal{S}$ is first partitioned into small spatial regions $R_s$, then further refined into category-specific regions $R_c$, and finally augmented with temporal information $R_t$. A resulting "gram" can be written as $r_{sct} = \{\text{mountain, church, 3am}\}$. This decomposition represents a trajectory $\mathcal{T}$ at the knowledge level, enabling the use of public knowledge to define *reachability*, i.e. to constrain the set of feasible perturbed trajectories. For instance, visiting a church at 3am is

*This concerns how to set the privacy parameter for trajectory data, rather than defining the "trajectory space" as the input space in LDP mechanisms. We focus on LDP mechanisms for location spaces in this dissertation, as the trajectory space in $\mathbb{R}^2$ is infinite-dimensional.

unlikely because it is typically closed, which reduces the reachable set of $\mathcal{T}'$. Finally, NGram applies the Exponential mechanism to sample a perturbed trajectory from this reachable set.

L-SRR [148] introduces the staircase randomized response (SRR) mechanism for location perturbation. Unlike $k$-RR, which assigns only two probabilities—one for the true location and another for all other locations—the SRR mechanism employs a staircase-shaped probability distribution. L-SRR recursively partitions the physical space $\mathcal{S}$ and forms location groups $G_1, G_2, \dots, G_g$ based on their distances to the true location. By integrating the SRR mechanism, L-SRR assigns higher probabilities to locations in groups that are closer to the true location, and lower probabilities to those farther away. This approach is similar to the Exponential mechanism, but uses group-level distance instead of location-level distance.

ATP [181] perturbs directions to restrict the set of feasible perturbed locations. For a trajectory $\mathcal{T} = \{\tau_1, \tau_2, \dots, \tau_n\}$, when perturbing $\tau_{i+1}$, ATP first perturbs the direction from $\tau_i$ to $\tau_{i+1}$. Specifically, it partitions the direction space into $k$ sectors and applies $k$-RR to sample a sector. The perturbed location $\tau'_{i+1}$ is then constrained to lie within the sampled sector, which reduces the candidate set of $\tau'_{i+1}$; ATP finally samples $\tau'_{i+1}$ from this constrained set using the Exponential mechanism.

We summarize their limitations in the context of continuous spaces as follows:

- Existing methods are designed for discrete location spaces. Discretizing a continuous domain is possible, but choosing an appropriate granularity is non-trivial.
- Beyond discretization, methods that rely on the Exponential mechanism (e.g. NGram and ATP) typically incur high computational cost for evaluating scores and sampling, and their utility can be sensitive to the spatial distribution of candidate locations. L-SRR requires constructing distance-based groups for every location, leading to similar scalability issues. Moreover, approaches that depend on public knowledge (e.g. NGram) are hard to design in continuous spaces (where the location set is infinite and semantics are difficult to define), are often dataset-specific, and the public knowledge is typically *known* to the adversary, which can further weaken privacy protection.

Appendix D.2.2 details the limitations of the Exponential mechanism. In brief, perturbing a location requires evaluating the score function $d(x, y)$ for all candidate outputs $y$ associated with $x$, which takes $\Theta(m)$ time where $m$ is the number of locations in the discrete space. Moreover, generating each perturbed location requires sampling from an $m$-piece cumulative distribution function (CDF), which also incurs $\Theta(m)$ time. Such a per-location cost is prohibitive for large-scale location spaces. Finally, because the sampling probabilities are induced by the score function, they can be sensitive to the spatial distribution of candidate locations and their relative distances, which may further degrade utility.

To address these limitations, we propose TraCS, a novel trajectory collection method that ensures rigorous LDP guarantees for trajectories in continuous spaces. TraCS is carefully designed to tackle the privacy, utility, and efficiency challenges inherent in handling continuous domains without relying on discretization.

## 6.2 The Proposed Method: TraCS

This section presents the proposed methods, TraCS-D and TraCS-C, along with their theoretical analyses and extensions.

### 6.2.1 Location Space

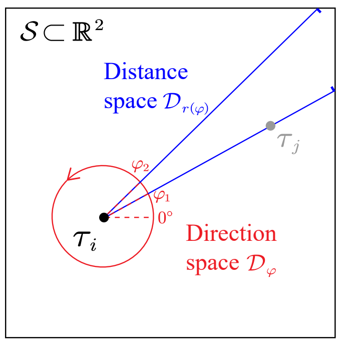
We consider the location space constrained by a pair of longitude and latitude lines, i.e. $\mathcal{S} \subset \mathbb{R}^2$ is a rectangular area $[a_{\text{sta}}, a_{\text{end}}] \times [b_{\text{sta}}, b_{\text{end}}]$, where $a_{\text{sta}}, a_{\text{end}}$ denote the longitudes and $b_{\text{sta}}, b_{\text{end}}$ denote the latitudes. Consequently, each location $\tau_i \in \mathcal{S}$ has a natural representation as a pair of coordinates $(a_i, b_i)$. This representation aligns with real-world location data (e.g. GPS data) and existing trajectory collection methods for discrete spaces [34, 40, 76, 148, 181].

The core idea behind having various LDP methods for the location space $\mathcal{S}$ stems from its multiple decompositions. One decomposition means $\mathcal{S}$ can be represented by several subspaces. For example, in addition to the aforementioned longitude-latitude coordinate representation, each location $\tau_i \in \mathcal{S}$ can also be represented by a direction component and a distance component. Based on these decompositions, we propose two LDP methods: TraCS-D, which perturbs the direction and distance subspaces, and TraCS-C, which perturbs subspaces of the Cartesian coordinates.

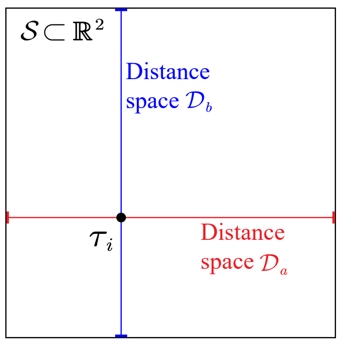
### 6.2.2 TraCS-D

Continuous space $\mathcal{S}$ can be represented as the composition of a direction space $\mathcal{D}_\varphi$ and a distance space $\mathcal{D}_{r(\varphi)}$ at each (reference) location $\tau_i$. Figure 6.1a illustrates this decomposition.

**Direction space $\mathcal{D}_\varphi$.** In the space $\mathcal{S} \subset \mathbb{R}^2$, the direction is represented by the angle $\varphi$ relative to a reference direction. We define the reference direction as the latitude lines, i.e. the $0^\circ$ direction. Thus, the direction space $\mathcal{D}_\varphi$ is a circular domain $[0, 2\pi)$ for any location. Fixing a location $\tau_i$, any


$\mathcal{S} \subset \mathbb{R}^2$
Distance space $\mathcal{D}_{r(\varphi)}$
$\tau_j$
$\varphi_2$
$\varphi_1$
$0°$
$\tau_i$
Direction space $\mathcal{D}_\varphi$

(a) TraCS-D: $\mathcal{S} = \mathcal{D}_\varphi \otimes \mathcal{D}_{r(\varphi)}$


$\mathcal{S} \subset \mathbb{R}^2$
Distance space $\mathcal{D}_b$
$\tau_i$
Distance space $\mathcal{D}_a$

(b) TraCS-C: $\mathcal{S} = \mathcal{D}_a \times \mathcal{D}_b$

Figure 6.1: Two decompositions of location space $\mathcal{S}$ at $\tau_i$. Any other location $\tau_j \in \mathcal{S}$ can be represented by direction-distance coordinates $(\varphi, r(\varphi))$ or Cartesian coordinates $(a, b)$.

other location $\tau_j \neq \tau_i$ has a unique direction with respect to $\tau_i$.

**Distance space $\mathcal{D}_{r(\varphi)}$.** The other subspace, i.e. the distance space $\mathcal{D}_{r(\varphi)}$, is determined by the direction $\varphi$. Its size is the distance from $\tau_i$ to the boundary of $\mathcal{S}$ in the direction $\varphi$, which is a function of $\varphi$. Since $\mathcal{S}$ is rectangular, $r(\varphi)$ has a closed-form expression, which is detailed later. Thus, fixing $\tau_i$ and a direction $\varphi$, any other location $\tau_j$ has a unique distance $r(\varphi)$ with respect to $\tau_i$.

By this decomposition, we can represent any location $\tau_j \in S$ as coordinates $(\varphi, r(\varphi)) \in (\mathcal{D}_\varphi, \mathcal{D}_{r(\varphi)})$ relative to $\tau_i$. As an example, $\tau_j$ in Figure 6.1a is represented by $(\varphi_1, r(\varphi_1))$. This representation is unique and reversible: any pair of coordinates $(\varphi, r(\varphi))$ can be mapped back to a unique location $\tau_j \in \mathcal{S}$. Thus, if $\tau_j$ is a sensitive location, an LDP mechanism applied to $\tau_j = (\varphi, r(\varphi))$ is essentially applied to $\varphi$ and $r(\varphi)$. By Composition Theorem 11, we can design perturbation mechanisms for $\varphi$ and $r(\varphi)$, respectively, to achieve LDP for $\tau_j$.

### Direction Perturbation

The technical challenge in direction perturbation is designing an LDP mechanism for the circular domain $\mathcal{D}_\varphi = [0, 2\pi)$. This is non-trivial due to the following properties of the circular domain:

- The circular domain $[0, 2\pi)$ is bounded but not finite, making the unbounded mechanisms and discrete mechanisms inapplicable.
- The circular domain has a unique distance metric, e.g. the distance between 0 and $2\pi$ is 0, not $2\pi$, making the Euclidean distance-based mechanisms inapplicable.

Specifically, classical mechanisms like Laplace and Gaussian [48] add unbounded noise, resulting in an output domain of $(-\infty, +\infty)$. This output domain makes them inapplicable to the circular domain $[0, 2\pi)$. Additionally, these mechanisms are designed for the distance metric $d(y, x) = |y - x|$, which is inconsistent with the distance in circular domain. Discrete mechanisms, such as $k$-RR [78], assume a finite domain of size $k$ and therefore cannot be directly applied to the continuous circular domain $[0, 2\pi)$. Recently, OGPM [184] proposed a piecewise-based mechanism tailored to circular domains, which inspires our design.

To highlight the necessity and superiority of TraCS-D's piecewise-based design compared to simpler alternatives, we start by presenting a strawman approach that extends the $k$-RR mechanism to the circular domain $[0, 2\pi)$.

**Strawman approach: $k$-RR + uniform sampling.** A former work ATP [181] divides the direction space $[0, 2\pi)$ into $k$ sectors and applies the $k$-RR mechanism. This approach ensures LDP for the $k$ sectors but not for the entire $[0, 2\pi)$ domain. Even so, we can extend this approach to $[0, 2\pi)$ by further uniformly sampling a direction $\varphi'$ from the output sector of $k$-RR.

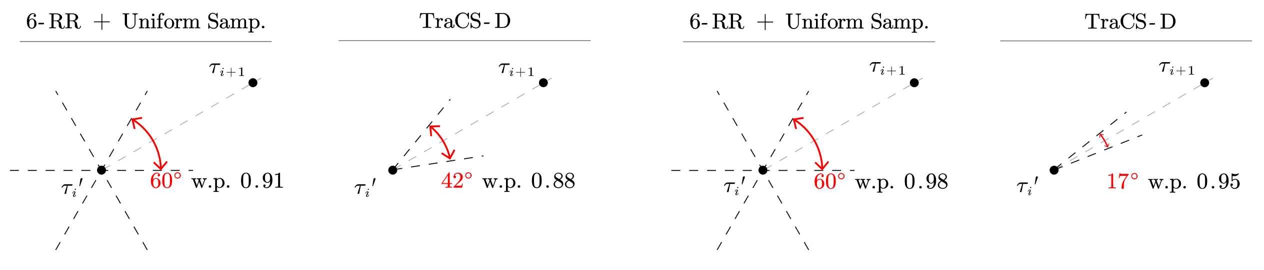
Specifically, the circular domain $[0, 2\pi)$ is divided into $k$ sectors $[i - 1, i) \cdot 2\pi/k$ for $i = 1, 2, \ldots, k$. Assume the sensitive direction $\varphi$ falls into the $j$-th sector. Applying the $k$-RR mechanism outputs sector $i = j$ with probability $p = \exp(\varepsilon)/(k - 1 + \exp(\varepsilon))$, or outputs sector $i \neq j$ with probability $p/\exp(\varepsilon)$. Then, we uniformly sample a direction $\varphi'$ from the $i$-th sector, i.e. $\varphi' \sim U(i - 1, i) \cdot 2\pi/k$.

**Limitation.** This strawman approach introduces inherent *inner-sector* errors due to uniform sampling within a specific sector. That is, even though the perturbed direction $\varphi'$ falls into the same sector as the sensitive direction $\varphi$, the distance between $\varphi$ and $\varphi'$ may still be large due to the large sector size. This issue is particularly prominent when $k$ is small, as each sector then spans $2\pi/k$ radians. Uniform sampling within such wide sectors introduces substantial inner-sector errors, which persist regardless of how large the privacy parameter $\varepsilon$ is set. On the other hand, increasing $k$ to reduce sector width impairs the utility of the $k$-RR mechanism [153].

**Design rationale.** To address this limitation, we design a direction perturbation mechanism in which the perturbed direction is independent of hyperparameters like $k$, thereby avoiding inherent errors beyond those introduced by the privacy parameter $\varepsilon$.

We adapt the design of piecewise-based mechanisms for circular domains from OGPM [184]. Specifically, (i) we instantiate a piecewise-based mechanism over the circular domain $[0, 2\pi)$, which guarantees LDP for the entire direction space; (ii) the perturbed direction is sampled from a piecewise probability distribution, which is centered around the sensitive direction with high probability and depends solely on the privacy parameter $\varepsilon$, thus eliminating the need for pre-defined fixed sectors.

**Definition 9** (Direction Perturbation Mechanism)**.** *Given a sensitive direction $\varphi$ and a privacy*



(a) $\varepsilon = 4$ (b) $\varepsilon = 6$

Figure 6.2: Comparison of the dominant sectors of the strawman approach and TraCS-D with $\varepsilon = 4$ and $\varepsilon = 6$. The red angular arcs indicate the dominant sectors, with their probabilities shown on the right. The dominant sector of TraCS-D narrows as $\varepsilon$ increases, leading to a smaller inner-sector (sampling) error. In contrast, the strawman approach has a fixed dominant sector, which is independent of $\varepsilon$ and leads to a large inner-sector error.

*parameter $\varepsilon$, TraCS-D's direction perturbation mechanism $\mathcal{M}_\circ : [0, 2\pi) \to [0, 2\pi)$ is defined by:*

$$pdf[\mathcal{M}_\circ(\varphi) = \varphi'] = \begin{cases} p_\varepsilon & \text{if } \varphi' \in [l_{\varphi,\varepsilon}, r_{\varphi,\varepsilon}), \\ p_\varepsilon / \exp(\varepsilon) & \text{otherwise,} \end{cases}$$

*where $p_\varepsilon = \frac{1}{2\pi}\exp(\varepsilon/2)$ is the sampling probability, and $[l_{\varphi,\varepsilon}, r_{\varphi,\varepsilon})$ is the sampling interval that*

$$l_{\varphi,\varepsilon} = \left(\varphi - \pi\frac{\exp(\varepsilon/2) - 1}{\exp(\varepsilon) - 1}\right) \mod 2\pi,$$
$$r_{\varphi,\varepsilon} = \left(\varphi + \pi\frac{\exp(\varepsilon/2) - 1}{\exp(\varepsilon) - 1}\right) \mod 2\pi.$$

The above mechanism $\mathcal{M}_\circ$ is defined by a piecewise probability distribution with three pieces, where the central piece $[l_{\varphi,\varepsilon}, r_{\varphi,\varepsilon})$ has a higher probability density $p_\varepsilon$. As a piecewise-based mechanism, it evidently satisfies LDP for the circular domain $[0, 2\pi)$; see Appendix D.1.1 for the proof.

A key advantage of TraCS-D's direction perturbation mechanism is the "dominant" sector. We refer to the central piece $[l_{\varphi,\varepsilon}, r_{\varphi,\varepsilon})$ as the dominant sector because it centers around the sensitive direction $\varphi$ with high probability. The dominant sector adapts dynamically to the privacy parameter $\varepsilon$: as $\varepsilon$ increases, the dominant sector becomes narrower and more tightly centered around the sensitive direction $\varphi$. This adaptivity effectively mitigates the inner-sector errors inherent in the strawman approach, where the sector width is fixed and independent of $\varepsilon$.

**Example 10.** *Figure 6.2 compares the strawman approach and TraCS-D. Assume the next location $\tau_{i+1}$ is a sensitive location needing direction perturbation, thus the sensitive direction is $\varphi : \tau_i' \to \tau_{i+1}$. The dominant sector of TraCS-D is much smaller as $\varepsilon$ increases. For instance, when $\varepsilon = 6$ and $\varphi = \pi/6$, it can be calculated that $p_\varepsilon \approx 3.20$ and $[l_{\varphi,\varepsilon}, r_{\varphi,\varepsilon}) \approx [0.119\pi, 0.214\pi)$. Thus, the dominant* *sector has a size of* $0.095\pi \approx 17^\circ$ *with probability* $0.095\pi \times 3.20 \approx 0.95$ *of being chosen. In comparison, the strawman approach has a dominant sector of* $2\pi/6 = 60^\circ$ *with probability* $0.98$ *of being chosen. With almost the same probability of being chosen, TraCS-D's dominant sector is significantly narrower than that of the strawman approach, resulting in a much smaller inner-sector (sampling) error.*

Detailed comparison between the strawman approach and Definition 9 is provided in Appendix D.2.4. This Appendix will show that TraCS-D achieves a better trade-off between the size and probability of the dominant sector compared to the strawman approach with any $k$ value.

### Distance Perturbation

The other subspace of TraCS-D is the distance space $\mathcal{D}_{r(\varphi)}$. Besides the boundedness, the size of this space also varies with the reference location $\tau_i$ and the direction $\varphi$. The technical challenge in designing an LDP mechanism for this space lies in calculating the size $|\mathcal{D}_{r(\varphi)}|$ for any $\varphi$ at any $\tau_i$.

Given location $\tau_i$, any other location $\tau_j \in \mathcal{S}$ can be represented by a pair of coordinates $(\varphi, r(\varphi))$ relative to $\tau_i$. Since we focus on $\mathcal{S}$ as a rectangular area $[a_{\text{sta}}, a_{\text{end}}] \times [b_{\text{sta}}, b_{\text{end}}]$, the distance from $\tau_i$ to the boundary of $\mathcal{S}$ has a closed-form expression depending on the direction $\varphi$. Specifically, this distance falls into one of the following four cases:

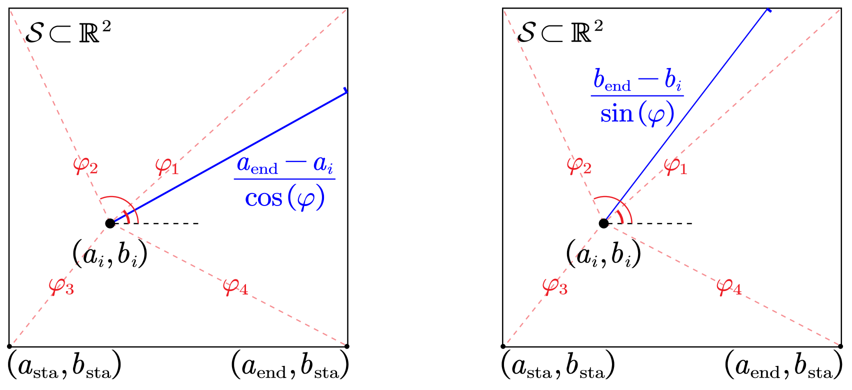
$$|\mathcal{D}_{r(\varphi)}| \in \left\{ \frac{a_{\text{end}} - a_i}{\cos(\varphi)}, \frac{b_{\text{end}} - b_i}{\sin(\varphi)}, \frac{a_i - a_{\text{sta}}}{-\cos(\varphi)}, \frac{b_i - b_{\text{sta}}}{-\sin(\varphi)} \right\}, \tag{6.1}$$

depending on the value of $\varphi$. Appendix D.2.5 provides the detailed equation form of Equation (6.1). Figure 6.3 illustrates the first two cases of $|\mathcal{D}_{r(\varphi)}|$: when $\varphi$ falls into $[0, \varphi_1)$, i.e. Figure 6.3a, $|\mathcal{D}_{r(\varphi)}| = (a_{\text{end}} - a_i)/\cos(\varphi)$; Figure 6.3b shows the second case, i.e. when $\varphi \in [\varphi_1, \varphi_2)$, then $|\mathcal{D}_{r(\varphi)}| = (b_{\text{end}} - b_i)/\sin(\varphi)$.

With the known distance space $[0, |\mathcal{D}_{r(\varphi)}|]$ and sensitive distance $r(\varphi)$, we can design a distance perturbation mechanism to guarantee LDP for $\mathcal{D}_{r(\varphi)}$. For the convenience of presentation, we normalize the distance $r(\varphi)$ by $|\mathcal{D}_{r(\varphi)}|$, resulting in a normalized distance $\overline{r}(\varphi) \in [0, 1)$.[†] We then employ the piecewise-based mechanism in OGPM [184] for the normalized distance $\overline{r}(\varphi)$ to guarantee LDP on $[0, 1)$.

**Definition 10** (Distance Perturbation Mechanism)**.** *Given a sensitive distance* $\overline{r}(\varphi)$ *and a privacy*

[†]We use $[0, 1)$ instead of $[0, 1]$ to ease the presentation of Definition 10 and to align with the circular domain. These two domains are equivalent in implementation.


$\mathcal{S} \subset \mathbb{R}^2$
$\frac{a_{\text{end}} - a_i}{\cos(\varphi)}$
$\varphi_1$
$\varphi_2$
$\varphi_3$
$\varphi_4$
$(a_i, b_i)$
$(a_{\text{sta}}, b_{\text{sta}})$
$(a_{\text{end}}, b_{\text{sta}})$
$\frac{b_{\text{end}} - b_i}{\sin(\varphi)}$

(a) $\varphi \in [0, \varphi_1)$ (b) $\varphi \in [\varphi_1, \varphi_2)$

Figure 6.3: Two cases of $|\mathcal{D}_{r(\varphi)}|$ at $\tau_i$ (blue lines).

*parameter $\varepsilon$, TraCS-D's distance perturbation mechanism $\mathcal{M}_{-} : [0,1) \to [0,1)$ is defined by:*

$$pdf[\mathcal{M}_{-}(\overline{r}(\varphi)) = \overline{r}'(\varphi)] = \begin{cases} p_\varepsilon & \textit{if } \overline{r}'(\varphi) \in [u,v), \\ p_\varepsilon / \exp(\varepsilon) & \textit{otherwise}, \end{cases}$$

*where $p_\varepsilon = \exp(\varepsilon/2)$ is the sampling probability, and $[u,v)$ is the sampling interval that*

$$[u,v) = \begin{cases} \overline{r}(\varphi) + [-C, C) & \textit{if } \overline{r}(\varphi) \in [C, 1-C), \\ [0, 2C) & \textit{if } \overline{r}(\varphi) \in [0, C), \\ [1-2C, 1) & \textit{otherwise}, \end{cases}$$

*with $C = (\exp(\varepsilon/2) - 1)/(2\exp(\varepsilon) - 2)$.*

Similar to $\mathcal{M}_{\circ}$, the above mechanism $\mathcal{M}_{-}$ has a higher probability density $p_\varepsilon$ in the central piece $[u,v)$, and a larger $\varepsilon$ results in a higher $p_\varepsilon$ and a narrower $[u,v)$, boosting the utility. It ensures LDP for $[0,1)$ and outputs a perturbed normalized distance $\overline{r}'(\varphi)$. To cooperate with the direction perturbation, $\overline{r}'(\varphi)$ can be mapped to another distance space $\mathcal{D}_{r(\varphi')}$ at the perturbed direction $\varphi'$. This is a linear mapping without randomness, so the post-processing property [48] ensures it preserves the same privacy level.

### Workflow of TraCS-D

Combining the direction perturbation and distance perturbation, TraCS-D perturbs each sensitive location in the trajectory.

Algorithm 1 presents the workflow of TraCS-D. It takes the location space $\mathcal{S}$, the sensitive trajectory $\mathcal{T}$, and the privacy parameter $\varepsilon$ as input. Since TraCS-D relies on a non-sensitive reference location,

```
Algorithm 1: TraCS-D
Input: Rectangular location space $\mathcal{S}$, sensitive trajectory $\mathcal{T} = \{\tau_1, \tau_2, \ldots, \tau_n\}$, privacy parameter $\varepsilon$
Output: Perturbed trajectory $\mathcal{T}' = \{\tau'_1, \tau'_2, \ldots, \tau'_n\}$
1  $\mathcal{T}' \leftarrow \emptyset, \mathcal{T} \leftarrow \tau'_0 \cup \mathcal{T}$;                 ▷ Add a dummy location $\tau'_0$
2  for $i \leftarrow 0$ to $n-1$ do
3      $\tau'_i = (a_i, b_i), \tau_{i+1} = (a_{i+1}, b_{i+1})$;      ▷ $\tau'_i$ is the ref location for this iteration
4      $\varphi \leftarrow \text{atan2}(b_{i+1} - b_i, a_{i+1} - a_i)$;         ▷ Sensitive direction
5      $\varphi' \leftarrow \mathcal{M}_\circ(\varphi; \varepsilon_d)$;                        ▷ Perturb direction
6      $R \leftarrow |\mathcal{D}_{r(\varphi)}|$ in Equation (6.1);
7      $\overline{r}(\varphi) \leftarrow ||\tau_{i+1} - \tau'_i||_2 / R$;           ▷ Sensitive (norm.)  distance
8      $\overline{r}'(\varphi) \leftarrow \mathcal{M}_-(\overline{r}(\varphi); \varepsilon - \varepsilon_d)$;          ▷ Perturb distance
9      $R' \leftarrow |\mathcal{D}_{r(\varphi')}|$, $r'(\varphi') \leftarrow \overline{r}'(\varphi) \times R'$;        ▷ De-normalize
       ▷ Transform back to (longitude, latitude)
10     $\tau'_{i+1} = (a_i + r'(\varphi')\cos(\varphi'), b_i + r'(\varphi')\sin(\varphi'))$;
11     $\mathcal{T}' \leftarrow \mathcal{T}' \cup \tau'_{i+1}$;                 ▷ $\tau'_{i+1}$ is the next ref location
12 return $\mathcal{T}'$;
```

we add a dummy location $\tau'_0$ to $\mathcal{T}$, which can be the starting coordinate of $\mathcal{S}$ or a randomly drawn location. Then, TraCS-D iterates over each pair of consecutive locations $\tau'_i$ and $\tau_{i+1}$ (line 3). For each sensitive location $\tau_{i+1}$, it perturbs the direction (red block) and normalized distance (blue block) with privacy parameters $\varepsilon_d$ and $\varepsilon - \varepsilon_d$, respectively. The perturbed distance is then mapped along the perturbed direction (line 9). Line 11 updates the reference location to the perturbed location $\tau'_{i+1}$, which is then used as the reference for the next iteration $(i+1)$. Finally, the algorithm outputs the perturbed trajectory $\mathcal{T}'$.

Algorithm 1 uses each $\tau'_i$ as the reference location for the next perturbation. In fact, it can be any non-sensitive location in $\mathcal{S}$. We choose different $\tau'_i$ for alleviating the impact of a specific reference location.

### Analysis of TraCS-D

This subsection analyzes the privacy, computational complexity, and utility of TraCS-D.

**Theorem 12.** *TraCS-D (Algorithm 1) satisfies $n\varepsilon$-LDP for the rectangular location space $\mathcal{S}$.*

*Proof.* (Sketch) The direction perturbation mechanism $\mathcal{M}_\circ$ satisfies $\varepsilon_d$-LDP by its definition. For the distance perturbation, the randomness comes entirely from mechanism $\mathcal{M}_-$, which satisfies $(\varepsilon - \varepsilon_d)$-LDP. The post-processing property preserves the same privacy level after linearly mapping the perturbed distance. Then each location satisfies $\varepsilon$-LDP by Composition Theorem 11 (or by computing the 2D *pdf* ratio in the LDP definition). Hence, the entire perturbed trajectory satisfies $n\varepsilon$-LDP. Appendix D.1.2 provides details. $\square$

**Complexity.** TraCS-D (Algorithm 1) has $\Theta(1)$ time complexity for each location, as each line is $\Theta(1)$. Thus, the entire algorithm has $\Theta(n)$ time complexity, where $n$ is the length of the trajectory. The space complexity is also $\Theta(n)$ because it needs to store the perturbed trajectory $\mathcal{T}'$.

The $\Theta(n)$ time complexity is optimal for per-location perturbation mechanisms. Among existing LDP mechanisms for discrete location spaces, even if we ignore the complexity of the score function and sampling in the Exponential mechanism [121], NGram [34] has $\Theta(tn)$ time complexity, where $t$ is the number of predefined "gram". It also includes a large constant factor due to its search to satisfy reachability constraints. L-SRR [148] has $\Theta(mn)$ time complexity, where $m$ is the number of locations, due to the grouping of locations. ATP [181] also has $\Theta(mn)$ time complexity, because of its trajectory merging step.

Note that the $\Theta(n)$ space complexity is for the convenience of presenting the algorithm. It can be reduced to $\Theta(1)$ by only updating each perturbed location $\tau'_{i+1}$ in place within $\mathcal{T}$. This results in a lightweight memory footprint, which is crucial for edge computing.

**Theorem 13.** *In TraCS-D, both the worst-case mean square error (MSE) of $\mathcal{M}_\circ$ and $\mathcal{M}_-$ converge to zero with a rate of $\Theta(e^{-\varepsilon/2})$.*

*Proof.* (Sketch) The MSE can be calculated by the closed-form expression of $\mathcal{M}_\circ$ and $\mathcal{M}_-$, allowing us to prove the convergence rate. Appendix D.1.3 provides the details. $\square$

This theorem indicates that both the error of the perturbed direction and distance are reduced exponentially with the privacy parameter $\varepsilon$. Note that the size of the distance space also affects the variance. Specifically, denote the output of $\mathcal{M}_-$ as a random variable $Y$ and the sensitive input as $x$. After the linear mapping, the perturbed distance is $Y \cdot |\mathcal{D}_r|$ and the sensitive distance is $x \cdot |\mathcal{D}_r|$, where $|\mathcal{D}_r|$ is the size of the distance space. Thus, the MSE of the perturbed distance is $\mathrm{MSE}[Y \cdot |\mathcal{D}_r|] = |\mathcal{D}_r|^2 \cdot \mathrm{MSE}[Y]$. Therefore, a smaller $|\mathcal{D}_r|$ leads to a smaller MSE of the perturbed distance.

### 6.2.3 TraCS-C

TraCS-C decomposes the rectangular location space $\mathcal{S}$ into two independent distance spaces $\mathcal{D}_a$ and $\mathcal{D}_b$, as shown in Figure 6.1b. Specifically, we can fix $(a_{\text{sta}}, b_{\text{sta}})$ as the reference location,

```
Algorithm 2: TraCS-C
  Input: Rectangular location space $\mathcal{S}$, sensitive trajectory $\mathcal{T} = \{\tau_1, \tau_2, \ldots, \tau_n\}$, privacy parameter $\varepsilon$
  Output: Perturbed trajectory $\mathcal{T}' = \{\tau'_1, \tau'_2, \ldots, \tau'_n\}$
1 $\mathcal{T}' \leftarrow \emptyset$;
2 for $i \leftarrow 1$ to $n$ do
3     $\tau_i = (a_i, b_i)$;
4     $(\overline{d}_a, \overline{d}_b) \leftarrow (\frac{a_i - a_{\text{sta}}}{a_{\text{end}} - a_{\text{sta}}}, \frac{b_i - b_{\text{sta}}}{b_{\text{end}} - b_{\text{sta}}})$;        ▷ Normalize coordinates
5     $\overline{d}'_a \leftarrow \mathcal{M}_{-}(\overline{d}_a; \varepsilon/2)$;        ▷ Perturb coordinates
6     $\overline{d}'_b \leftarrow \mathcal{M}_{-}(\overline{d}_b; \varepsilon/2)$;
7     $(d'_a, d'_b) \leftarrow (\overline{d}'_a |\mathcal{D}_a|, \overline{d}'_b |\mathcal{D}_b|)$, $\tau'_i = (a_{\text{sta}}, b_{\text{sta}}) + (d'_a, d'_b)$;        ▷ De-normalize to $(\mathcal{D}_a, \mathcal{D}_b)$
8     $\mathcal{T}' \leftarrow \mathcal{T}' \cup \{\tau'_i\}$;
9 return $\mathcal{T}'$;
```

with longitude $a_{\text{sta}}$ as the $a$-axis and latitude $b_{\text{sta}}$ as the $b$-axis. Then any location $\tau_i \in \mathcal{S}$ can be represented as a pair of Cartesian coordinates $(d_a, d_b)$, where $d_a \in \mathcal{D}_a$ and $d_b \in \mathcal{D}_b$ are the distances from $\tau_i$ to the $a$-axis and $b$-axis, respectively. This representation is unique and reversible: any pair of distances $(d_a, d_b)$ can be mapped back to a unique location $\tau_i \in \mathcal{S}$. Thus, if $\tau_i$ is a sensitive location, an LDP mechanism applied to $\tau_i$ is essentially applied to $d_a$ and $d_b$. By Composition Theorem 11, we can achieve LDP for $\tau_j$ by applying perturbation mechanisms for $d_a$ and $d_b$, respectively.

### Coordinates Perturbation

In TraCS-C's representation of locations, $\mathcal{D}_a = [0, a_{\text{end}} - a_{\text{sta}})$ and $\mathcal{D}_b = [0, b_{\text{end}} - b_{\text{sta}})$ are independent of any specific location. Therefore, the location perturbation of any $\tau_i$ can be performed independently in $\mathcal{D}_a$ and $\mathcal{D}_b$.

Following the same idea of distance perturbation in TraCS-D, we normalize the distance $d_a$ and $d_b$ to $[0, 1)$ by dividing $|\mathcal{D}_a|$ and $|\mathcal{D}_b|$, respectively. Then the mechanism $\mathcal{M}_{-}$ in Definition 10 provides $\varepsilon$-LDP for the normalized distances, and the linear mappings back to $\mathcal{D}_a$ and $\mathcal{D}_b$ preserve the privacy guarantee.

Algorithm 2 shows the workflow of TraCS-C. It takes the rectangular location space $\mathcal{S}$, the sensitive trajectory $\mathcal{T}$, and the privacy parameter $\varepsilon$ as input, and outputs the perturbed trajectory $\mathcal{T}'$. For each location $\tau_i = (a_i, b_i)$ in $\mathcal{T}$, Line 4 computes its coordinates $(d_a, d_b)$ and normalizes them to $(\overline{d}_a, \overline{d}_b) \in [0, 1) \times [0, 1)$. Lines 5-6 perturb the normalized coordinates $\overline{d}_a$ and $\overline{d}_b$ using $\mathcal{M}_{-}$ with privacy parameter $\varepsilon/2$. Then Line 8 maps the perturbed normalized coordinates back to $\mathcal{D}_a$ and $\mathcal{D}_b$, and converts them to the longitude-latitude representation $\tau'_i$.

**Analysis of TraCS-C**

Privacy and complexity of TraCS-C can be analyzed similarly to TraCS-D.

**Theorem 14.** *TraCS-C (Algorithm 2) satisfies $n\varepsilon$-LDP for the rectangular location space $\mathcal{S}$.*

*Proof.* (Sketch) Algorithm 2 uses mechanism $\mathcal{M}_{-}$ twice, each with a privacy parameter $\varepsilon/2$. By Composition Theorem 11 (or by computing the 2D *pdf* ratio in the LDP definition), their composition satisfies $\varepsilon$-LDP. The subsequent linear mappings to $\mathcal{D}_a$ and $\mathcal{D}_b$ are post-processing steps that preserve the same privacy. Hence, each perturbed location satisfies $\varepsilon$-LDP, and the whole trajectory satisfies $n\varepsilon$-LDP. Appendix D.1.4 provides details. $\square$

**Complexity.** TraCS-C (Algorithm 2) has $\Theta(n)$ time complexity and $\Theta(n)$ space complexity, where $n$ is the length of the trajectory. The time complexity is $\Theta(n)$ because each line of Algorithm 2 is $\Theta(1)$. The space complexity is $\Theta(n)$ because it needs to store the perturbed trajectory $\mathcal{T}'$. Note that the $\Theta(n)$ space complexity is for the convenience of presenting the algorithm. It can also be reduced to $\Theta(1)$ by updating each perturbed location $\tau_i'$ in place within $\mathcal{T}$, which benefits edge devices with limited computation resources.

### 6.2.4 Rounding to Discrete Space

Although TraCS is designed for continuous location spaces, it can be applied to any fine-grained discrete location space by rounding the perturbed locations to the nearest discrete locations. The rounding is a post-processing step that does not affect the privacy guarantee for the continuous space.

Specifically, if the discrete location space is a set of cells, we can round the perturbed location to the cell that contains it. Formally, assume $\mathcal{C} = \{c_1, \dots, c_m\} \subseteq \mathcal{S}$ discretize $\mathcal{S}$ into $m$ cells. If $\tau_i'$ is the perturbed location of $\tau_i$ in TraCS, then the discretized TraCS outputs $c_j$ such that $\tau_i' \in c_j$. Commonly, the cells are evenly divided, so the rounding costs $\Theta(1)$ time, as $c_j$ can be determined by the coordinates of $\tau_i'$.

If the location space is a set of location points, we can round the perturbed location to the nearest location point. Formally, assume $\mathcal{P} = \{p_1, \dots, p_m\} \subseteq \mathcal{S}$ is a set of location points, and $\tau_i'$ is the perturbed location of $\tau_i$ in TraCS. Then, the discretized TraCS outputs $\arg\min_{p_j \in \mathcal{P}} ||\tau_i' - p_j||_2$. The time complexity of rounding to the nearest location point is $\mathcal{O}(m)$ in the worst case. In practice, this time complexity can be significantly reduced by using heuristic or approximation algorithms.

The candidate locations in a discrete space are typically public knowledge, which enables a heuristic selection of a better reference location for TraCS-D. The key observation is that a smaller distance space $\mathcal{D}_{r(\varphi)}$ yields a smaller distance-perturbation error. Therefore, it is preferable to choose a reference location that minimizes the size of $\mathcal{D}_{r(\varphi)}$ with respect to the candidate locations.

Specifically, the direction from the reference location to other locations is often concentrated in a dominant sector, as illustrated in Figure 6.2. If such a sector covers most candidate locations (thus reducing rounding error) and is close to the boundary of the location space (thus shrinking the distance domain), then the corresponding reference location is a good choice.

### 6.2.5 Discussions and Extensions

This subsection discusses the comparison between TraCS-D and TraCS-C, Euclidean versus spherical geometry, technical considerations in TraCS, and other discussions and extensions.

#### Comparison of Privacy Between TraCS-D and TraCS-C

Although TraCS-D and TraCS-C provide the same level of privacy quantified by $\varepsilon$-LDP—more specifically, location-level (or event-level) LDP as described in surveys [17, 113]—they have different privacy interpretations.

In TraCS-D, the LDP guarantee is provided for the direction information and the subsequent distance information. This means that when the use of TraCS-D and the perturbed trajectory $\mathcal{T}'$ is publicly known, any observer can hardly infer the sensitive direction $\varphi : \tau_i' \to \tau_{i+1}$ and the sensitive distance $r(\varphi) : |\tau_i' \to \tau_{i+1}|$ from the known $\tau_i'$ and $\tau_{i+1}'$. Although the distance space $\mathcal{D}_{r(\varphi)}$ relies on a specific direction, it does not leak the direction information, as the direction space $\mathcal{D}_\varphi = [0, 2\pi)$ is independent of any location and TraCS-D uses the perturbed direction.

In TraCS-C, the LDP guarantee is provided for the two independent distance spaces $\mathcal{D}_a$ and $\mathcal{D}_b$. This means that when the use of TraCS-C and the perturbed trajectory $\mathcal{T}'$ is publicly known, any observer can hardly infer the sensitive distances $d_a$ and $d_b$.

#### Euclidean Geometry vs Spherical Geometry

TraCS is designed for a rectangular location space $\mathcal{S} \subset \mathbb{R}^2$ under Euclidean geometry, where distances are measured by the Euclidean metric. In particular, TraCS-D requires computing the distance spaces $\mathcal{D}_{r(\varphi)}$, which are defined with respect to the Euclidean distance. This choice is made for generality: (i) for many continuous trajectories (e.g. from wearable sensors or indoor devices), Euclidean geometry is a natural choice; and (ii) for city-scale GPS trajectories (e.g. in Chicago and Tokyo), the distortion induced by approximating geographic coordinates as Cartesian coordinates is typically negligible.

Under spherical geometry, distances are measured by the great-circle distance. (i) For country-scale GPS trajectories, standard map projections such as UTM [145] can convert GPS coordinates into local Cartesian coordinates, after which Euclidean distance can be used. (ii) For general spherical domains (e.g. the Earth's surface), we can treat longitude and latitude as two independent coordinates and redesign TraCS-C via independent composition of $(\mathcal{M}_\circ, \mathcal{M}_-)$. Specifically, for a location with GPS coordinates $\tau_i = (a_i, b_i) \in ([-\pi, \pi), [-\pi/2, \pi/2])$, i.e. longitude and latitude, we perturb longitude and latitude independently using $\mathcal{M}_\circ$ and $\mathcal{M}_-$, respectively. This design does not involve Euclidean distance and aligns with the semantics of longitude and latitude as separate coordinates.

### Why Perturb Direction First in TraCS-D

In TraCS-D, we perturb the direction first and then the distance. This order is motivated by two considerations. First, the direction space $\mathcal{D}_\varphi = [0, 2\pi)$ is location-independent, making it well suited for direct perturbation. Second, the distance space $\mathcal{D}_{r(\varphi)}$ is always defined relative to a specific direction $\varphi$. If we were to perturb the distance before the direction, then after perturbing the direction we would need to redefine the corresponding distance domain using the perturbed direction, which complicates the procedure. Perturbing the direction first avoids this issue and yields a cleaner mechanism design.

### Impact of the Size of $\mathcal{S}$

Intuitively, a larger location space $\mathcal{S}$ requires larger privacy parameters to achieve the same level of utility. For TraCS-D and TraCS-C, this impact is reflected in the size of the direction space and distance spaces.

In TraCS-D, the direction space $\mathcal{D}_\varphi = [0, 2\pi)$ is independent of the size of $\mathcal{S}$, meaning that the MSE of the perturbed direction, MSE$[\varphi']$, remains constant regardless of the size of $\mathcal{S}$. The distance space $\mathcal{D}_{r(\varphi)}$ depends on the specific direction $\varphi$ and the size of $\mathcal{S}$. In the worst case, $\mathcal{D}_{r(\varphi)}$ is the diagonal of $\mathcal{S}$. We have shown that the MSE of $\mathcal{M}_-$ is quadratic to the size of the distance space after linear mapping in Section 12. Therefore, when $|\mathcal{D}_{r(\varphi)}|$ increases linearly, the expected error of the perturbed distance, i.e. $\mathrm{E}[|r'(\varphi) - r(\varphi)|]$, also increases linearly. Nonetheless, the error in the direction space $\mathcal{D}_\varphi$ affects the error in $\mathcal{D}_{r(\varphi)}$ in a non-linear way. Specifically, under the Euclidean distance metric, this effect is $|\mathcal{D}_{r(\varphi)}| \cdot 2\sin(|\varphi - \varphi'|/2)$, which is the chord length of the arc between $\varphi$ and $\varphi'$.

In TraCS-C, both distance spaces $\mathcal{D}_a$ and $\mathcal{D}_b$ are determined by the size of $\mathcal{S}$. Since they use mechanism $\mathcal{M}_-$ independently, their expected error increases linearly with the increase of $|\mathcal{D}_a|$ and $|\mathcal{D}_b|$. Meanwhile, the error in $\mathcal{D}_a$ affects the error in $\mathcal{D}_b$ linearly.

### Comparison with Geo-indistinguishability

Geo-indistinguishability [7] (Geo-Ind for brevity, also called metric privacy [129] in recent works) is a more general (or weaker) version of LDP. From the perspective of Geo-ind, LDP force the proximity-aware function $d(x, y)$ in Geo-ind to be $d(x, y) = 1$ for all $x, y \in S$, ignoring that different $x$ and $y$ have different $d(x, y)$ in the context of their locations. Conversely, if the upper bound $\max_{x,y \in S} d(x, y) \leq d^*$ holds in $S$, then TraCS also satisfies $d^*\varepsilon$-Geo-ind. However, this trivial satisfaction of $d^*\varepsilon$-Geo-ind is highly suboptimal, since it uses the global upper bound of $d(x, y)$ rather than the fine-grained, actual values of $d(x, y)$ for different location pairs of $x$ and $y$.

We omit trajectory utility evaluations against Geo-ind as it is a different privacy model from LDP. Generally, a well-designed Geo-ind mechanism can achieve better data utility than LDP mechanisms. For piecewise-shaped mechanisms, it is possible to tailor different pieces to account for varying $d(x, y)$ for different $x, y$, thereby satisfying Geo-ind while achieving better data utility than the current LDP mechanisms.

### Other LDP Mechanisms for Bounded and Continuous Space

In this chapter, we employ utility-optimized piecewise-based mechanisms (OGPM) [184] for bounded numerical domains to design the direction and distance perturbations in TraCS. Other LDP mechanisms for bounded numerical domains can also be incorporated into TraCS, but most of them may require redesign to suit the direction perturbation, as the circular domain $\mathcal{D}_\varphi = [0, 2\pi)$ has a different distance metric than linear domains. One exception is the Purkayastha mechanism [160], which is designed for sphere $\mathbb{S}^{n-1}$, e.g. circular lines when $n = 2$. It can be an alternative to $\mathcal{M}_\circ$, but not be directly applicable to $S \subset \mathbb{R}^2$. We don't leverage it for direction perturbation because of its sampling complexity (Section 3.4.2 in [160]) and higher MSE [184].

Another category of LDP mechanisms for bounded numerical domains is truncated mechanisms, including truncated Laplace [73, 97] and truncated Gaussian mechanisms [21]. These mechanisms modify Laplace and Gaussian distributions by redesigning distributions on a bounded domain, e.g. $[0, 1)$. However, truncated mechanisms cannot preserve the same privacy level as their original counterparts [21, 73, 97]. Determining their new privacy level is non-trivial and typically relies on numerical testing algorithms [21, 73] rather than closed-form solutions. Compared with truncated mechanisms, piecewise-based mechanisms are explicitly designed for LDP in bounded numerical domains, making them more effective and easier to analyze.

### Other Piecewise-based Mechanisms

In Definition 9 and Definition 10, we use specific instantiations for the parameters $p$ and $[l, r)$ in the piecewise-based mechanism. Besides these instantiations, TraCS can be integrated with any other piecewise-based mechanisms [95, 98, 101, 150, 183] by redesigning them. For example, SW [95, 184] is a famous mechanism proposed for distribution estimation; it is defined on $[0, 1) \to [-b, 1 + b)$, with $b$ as a parameter. It uses different $p$ and $[l, r)$, and employs a strategy of maximizing the mutual information between the perturbed and sensitive data to determine $b$. SW can be redesigned [184] for the direction perturbation and distance perturbation in TraCS-D and TraCS-C. Appendix D.2.6 provides the detailed redesign of SW for them and shows intuitive comparisons with Definition 9 and Definition 10, and Appendix D.2.7 provides a comparison.

### Other Shapes of Location Space

The design of LDP mechanisms is space-specific, implying that extensions to other shapes of location space are not straightforward.

**By post-processing.** A practical workaround for non-rectangular continuous domains is to post-process perturbed locations so they lie inside the target region. If a perturbed location falls outside the shape, project it to the nearest point inside the shape (for polygons this is the closest point on the boundary). Because the shape is public and projection is a data-independent post-processing step, it does not weaken the LDP guarantee. This approach is especially useful for irregular shapes or unions of multiple cities.

**Direct extension.** Directly extending TraCS to other shapes requires decomposing the domain into independent subspaces that are compatible with our perturbation primitives. For TraCS, the challenge lies in determining the size of each subspace, i.e. $|\mathcal{D}_\varphi|$, $|\mathcal{D}_{r(\varphi)}|$, $|\mathcal{D}_a|$, and $|\mathcal{D}_b|$. Among them, $|\mathcal{D}_{r(\varphi)}|$ is not apparent but can be calculated for rectangular location spaces. Although calculating $|\mathcal{D}_{r(\varphi)}|$ is generally non-trivial for irregular shapes, TraCS can be extended to parallelogram location spaces by a linear transformation to a rectangular space. Meanwhile, TraCS-D can be extended to circular *regions*, which are widely used in relative-location representations (e.g. radar and sonar systems). Extending TraCS-C to circular regions is more challenging, since such regions do not admit a natural Cartesian decomposition into independent coordinates. Appendix D.2.7 reports experimental results of TraCS-D on circular location spaces.

## 6.3 Evaluation

This section evaluates the performance of TraCS on both continuous and discrete spaces.

### 6.3.1 Results on Continuous Space

The strawman method—our extension of $k$-RR for direction perturbation—can be combined with the distance perturbation mechanism in TraCS-D to ensure LDP in continuous spaces $\mathcal{S} \subset \mathbb{R}^2$. For simplicity, we continue to refer to this extended approach as the strawman method, and compare it with TraCS in continuous space. We also include the planar Laplace mechanism [7] + clipping (truncation) as a baseline. This baseline perturbs each location by adding noise drawn from a 2D planar Laplace distribution, and then truncates the output to the location space $\mathcal{S}$ if it falls outside.[‡]

#### Setup

TraCS requires the following parameters: the location space $\mathcal{S}$, the sensitive trajectory $\mathcal{T}$, the privacy parameter $\varepsilon$, and $\varepsilon_d$ for direction perturbation in TraCS-D. The strawman method also requires an instance of $k$ in $k$-RR. We use configurations as follows for the experiments.

- Location space $\mathcal{S}$: We consider synthetic and real-world settings. (i) The default location space is $[0, 1) \times [0, 1)$. Without loss of generality, we consider a larger space $\mathcal{S} = [0, 2) \times [0, 10)$ for comparison. We generate 100 random trajectories for each location space as synthetic datasets. Specifically, each trajectory is generated by randomly sampling a sequence of 100 points in $\mathcal{S}$. This setup excludes dataset-specific effects to emphasize algorithmic performance. (ii) Areas of TKY and CHI: These two trajectory datasets were collected in Tokyo and Chicago, extracted from the Foursquare dataset [171] and Gowalla dataset [27], respectively. Locations in TKY are within area $[139.47, 139.90) \times [35.51, 35.86)$ and CHI within area $[-87.9, -87.5) \times [41.6, 42.0)$. These two areas are visually shown in Figure D.8. We use the first 100 trajectories in each dataset.
- Privacy parameter $\varepsilon_d$: We set $\varepsilon_d = \varepsilon\pi/(\pi + 1)$ to heuristically balance the perturbation in $\mathcal{M}_\circ$ and $\mathcal{M}_-$ according to their domain sizes.[§]

[‡]Appendix D.2.8 provides more details on the planar Laplace mechanism + clipping.

[§]Appendix D.2.9 provides more discussion on setting $\varepsilon_d$.

- $k$-RR: We set $k = 6$ as the strawman method's default value, and consider 3-RR and 12-RR for comparison.

We evaluate $\varepsilon$ in the range $[2, 10]$. For 2D discrete spaces with large location sets, this range represents a moderate privacy level.[¶] For example, with $k$-RR mechanism and $|\mathcal{X}| = 3600$ locations, $\varepsilon = 10$ yields the true location output with probability $\approx 0.85$, while $\varepsilon = 2$ results in a much lower probability of $\approx 0.002$. The Exponential mechanism exhibits similar behavior, though the exact probabilities depend on the score function. In such cases where $|\mathcal{X}|$ is large, even though the probability ratio $\exp(\varepsilon)$ in the LDP definition is high, it remains difficult for an adversary to infer the true location with high probability due to the vast number of possible locations.

**Trajectory utility metrics.** We follow the Euclidean distance to measure the utility of the perturbed trajectory [34, 148, 181]. The distance between two locations $\tau_i$ and $\tau'_i$ is defined as $\|\tau_i - \tau'_i\|_2$. Therefore, given a sensitive trajectory $\mathcal{T}$ and a perturbed trajectory $\mathcal{T}'$, the average error (AE) among all locations in the trajectory is:

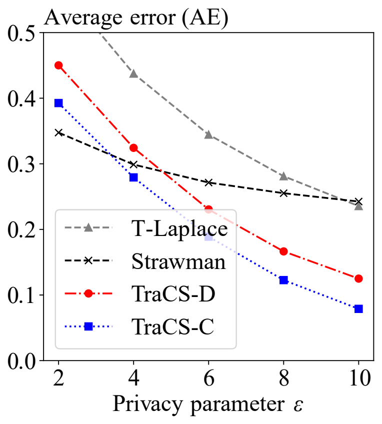
$$AE(\mathcal{T}, \mathcal{T}') = \frac{1}{|\mathcal{T}|} \sum_{i=1}^{|\mathcal{T}|} \|\tau_i - \tau'_i\|_2,$$

where $|\mathcal{T}|$ is the number of locations in the trajectory. A smaller AE indicates better trajectory utility. We compute the average AE across all trajectories in the dataset for comparison. Although AE is not a perfect utility measure, e.g. it may not fully reflect the requirements of specific downstream tasks, it remains the commonly used metric for evaluating the utility of perturbed trajectories [34, 113, 148, 181, 182]. Moreover, many other trajectory utility metrics are directly or indirectly related to AE, such as the preservation of range queries and hotspots [34, 181]. For brevity, we defer these additional metrics to Section 6.3.2.

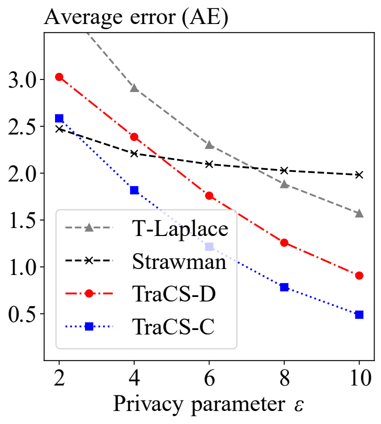
### Error Comparison on Synthetic Datasets

Figure 6.4 presents the error comparison between TraCS and the strawman method. We can observe the error advantage of TraCS over the truncated Laplace mechanism (T-Laplace) across all $\varepsilon$ values, and over the strawman method as the privacy parameter $\varepsilon$ increases. Specifically, in Figure 6.4a, TraCS-C exhibits smaller AE when $\varepsilon \gtrsim 3$, and TraCS-D exhibits smaller AE when $\varepsilon \gtrsim 5$. When the location space expands to $[0, 2) \times [0, 10)$, i.e. Figure 6.4b, the advantage of TraCS becomes more significant. Among TraCS, TraCS-C consistently has a smaller AE than TraCS-D. This is

[¶]In Definition 5, the privacy parameter $\varepsilon$ is defined with respect to a specific domain $\mathcal{X}$; its practical strength therefore depends on the size of $\mathcal{X}$.



(a) $\mathcal{S} = [0,1) \times [0,1)$



(b) $\mathcal{S} = [0,2) \times [0,10)$

Figure 6.4: Comparison on synthetic datasets.

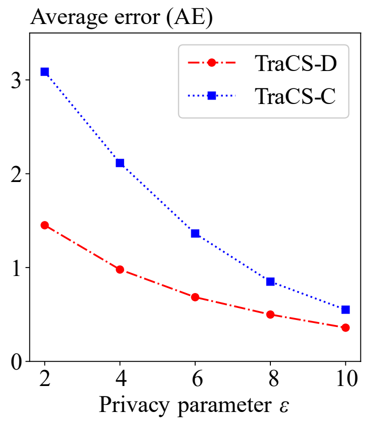


Figure 6.5: Condition TraCS-D outperforms TraCS-C.

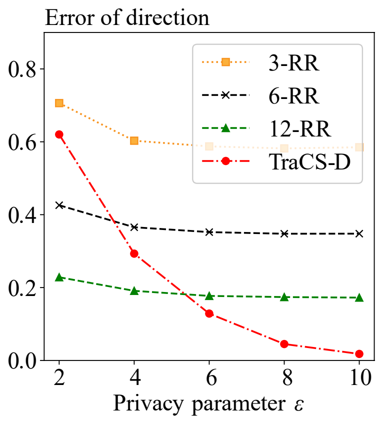
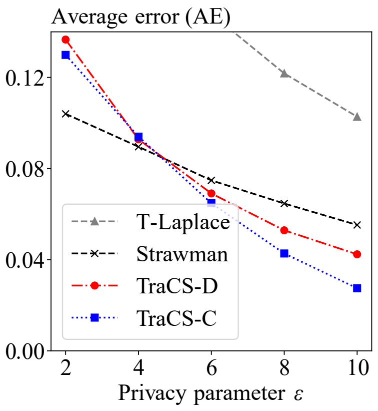


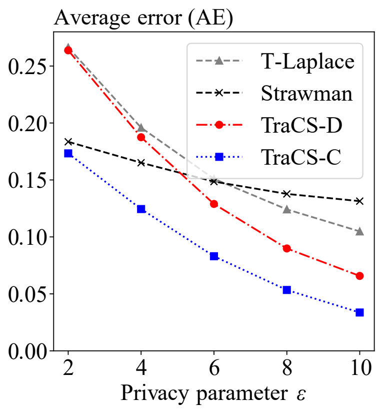
Figure 6.6: TraCS-D vs $k$-RR + uniform sampling.

because TraCS-D often deals with larger distance spaces $\mathcal{D}_{r(\varphi)}$, which also magnifies the error of direction perturbation. Statistically, the mean AE of TraCS-D is 91.1% of the strawman method in Figure 6.4a and 86.6% in Figure 6.4b across all the $\varepsilon$ values. For TraCS-C the corresponding ratios are 75.5% in Figure 6.4a and 64.0% in Figure 6.4b.

**Conditions TraCS-D outperforms TraCS-C.** The error of TraCS-C depends on the size of the distance spaces $\mathcal{D}_a$ and $\mathcal{D}_b$, while the error of TraCS-D depends only on the size of the distance space $\mathcal{D}_{r(\varphi)}$. Therefore, if $\mathcal{D}_{r(\varphi)}$ is majorly smaller than $\mathcal{D}_a$ or $\mathcal{D}_b$, TraCS-D will outperform TraCS-C. We conduct experiments to verify this condition. Specifically, we set $\mathcal{S} = [0,2) \times [0,10)$, and choose a sensitive trajectory along the $x$-axis, i.e. the short side of $\mathcal{S}$. Then we perturb the trajectory 1000 times using TraCS-C and TraCS-D and calculate the average AE. In this case, $|\mathcal{D}_{r(\varphi)}| \approx 2$, much smaller than $|\mathcal{D}_b| = 10$. Consequently, TraCS-D exhibits smaller AE than TraCS-C across all the $\varepsilon$



(a) TKY dataset



(b) CHI dataset

Figure 6.7: Comparison on real-world datasets.

values, as shown in Figure 6.5. In this experiment, the mean AE of TraCS-D is 49.8% of TraCS-C across all the $\varepsilon$ values.

**Error of direction perturbation.** Figure 6.6 shows the average error of direction perturbation in TraCS-D compared with the strawman method. We collect the sensitive directions of the trajectories in the location space $\mathcal{S} = [0, 1) \times [0, 1)$, and perform direction perturbation using TraCS-D and the strawman method. For the strawman method, we use $k$-RR with $k = 3, 6, 12$. This experiment reflects empirical error results of mechanism $\mathcal{M}_\circ$ in Definition 9 and the $k$-RR + uniform sampling. We can observe an exponentially decreasing error of TraCS-D as $\varepsilon$ increases, as stated in Theorem 13. In contrast, the strawman method exhibits an eventually stable error as $\varepsilon$ increases, due to the inherent inner-sector error of $k$-RR.

### Error Comparison on Real-world Datasets

Figure 6.7 shows the error comparison on the TKY and CHI areas. The trends largely mirror those observed on the synthetic datasets. T-Laplace incurs substantially larger AE than TraCS and the strawman method on the TKY dataset. Across all $\varepsilon$ values, the mean AE of TraCS-D is 96.8% of the strawman method on TKY and 94.5% on CHI, for TraCS-C the corresponding ratios are 89.7% and 61.2%. The performance gap is smaller on TKY than on CHI, because TKY's location space is more compact, which lowers the error values for all perturbation methods.

**Choose TraCS-D or TraCS-C**

From the above experiments, we can conclude criteria for choosing TraCS-D or TraCS-C: (i) TraCS-C generally has better trajectory utility than TraCS-D for random trajectories; (ii) TraCS-D is better for specific trajectories where their distance spaces are smaller than in TraCS-C, and suitable for circular areas.

### 6.3.2 Results on Discrete Space

Our approach can be applied to discrete spaces by rounding each perturbed location to its nearest discrete point. Importantly, the privacy guarantee is established in the underlying continuous domain and therefore still holds after rounding, regardless of the chosen discretization. In the following, we evaluate the performance of TraCS on discrete spaces and compare it with existing methods.

**Setup**

Our evaluation includes the following LDP methods for collecting trajectories in discrete spaces:

- NGram [34]: It is often impractical to acquire reachability knowledge in practice [181]. We instead consider a strong reachability constraint, i.e. from each location, the next location can only be within a distance of $0.75 \cdot \hat{a}$,[‖] where $\hat{a}$ is the maximal length of the location space. This constraint significantly reduces the location space for perturbation, especially for the locations near the boundary.
- L-SRR [148]: We use group number $g = 3$, the same as the empirical optimal value in their paper and code. Locations in the first group are within $0.3 \cdot \hat{a}$ distance from the sensitive location, where $\hat{a}$ is the maximal length of the location space. The second group is within $0.6 \cdot \hat{a}$ distance from the sensitive location, and the remaining locations are in the third group.
- ATP [181]: We set the $k$-RR parameter $k = 6$ in ATP. The privacy parameter $\varepsilon$ in original ATP is for the entire trajectory, we distribute it to each location's perturbation according to their algorithms.

[‖]While the original NGram employs much stricter time-reachability constraints (e.g. the next location must be within 0.1h · 4km/h), this constraint is also *known* to the adversary, which undermines the privacy guarantee. TraCS are LDP mechanisms guaranteeing privacy for the entire domain and do not incorporate such constraints.

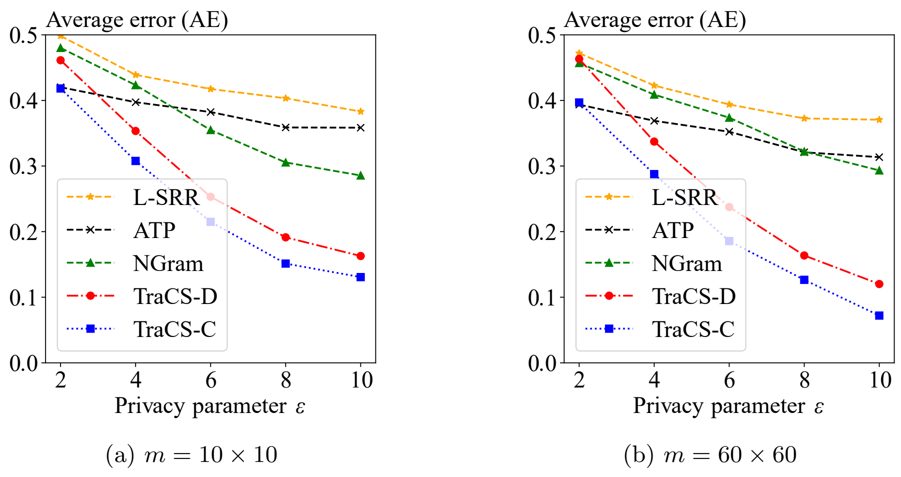


Figure 6.8: Comparison on synthetic datasets. $m$ is the number of discrete points in the location space.

- TraCS: We use the same privacy parameter $\varepsilon_d$ as in the continuous space experiments. For each perturbed location from TraCS, we round it to the nearest discrete location.

For a fair comparison, we use $\varepsilon$ as the privacy parameter for each location in all methods. The comparison is conducted on the following datasets:

- Synthetic datasets: We uniformly discretize $[0, 1) \times [0, 1)$ into $m$ discrete points and treat the $m$ points as the discrete location space. Specifically, we set $m = 10 \times 10$ and $m = 60 \times 60$ for comparison. Along with them, we generate 100 random trajectories with 100 locations each as synthetic datasets, i.e. each location is randomly sampled from the $m$ discrete points and connected sequentially.

- TKY and CHI (visualized in Figure D.8): We treat every location in the TKY and CHI datasets as a discrete point, which together constitute the discrete location space. TKY contains $7,798$ discrete locations; CHI contains $1,000$ discrete locations. We use the first 100 trajectories for evaluation.

For the TKY and CHI datasets, we perform the perturbation 5 times for each trajectory and compute the average AE, which is the same setup as in ATP [181].

### Error Comparison on Synthetic Datasets

Figure 6.8 presents the error comparison on the synthetic dataset. Figure 6.8a shows results for $m = 10 \times 10$, and Figure 6.8b shows results for $m = 60 \times 60$. We observe that TraCS-C consistently

Table 6.2: Time cost comparison (in milliseconds).

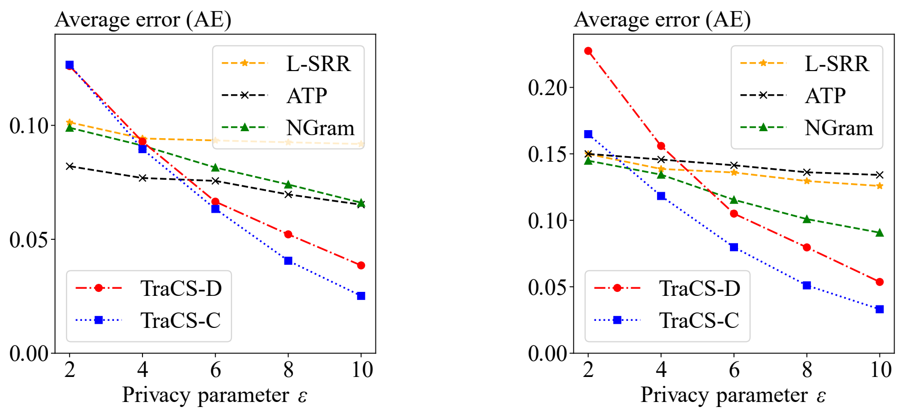
| | ATP | NGram | L-SRR | TraCS-D | TraCS-C |
|---|---|---|---|---|---|
| **Total** | 145.7 | 100.9 | 6.2 | 0.06 | 0.05 |
| **Perturb** | 125.8 | 92.8 | 0.003 | 0.018 | 0.003 |

outperforms NGram, L-SRR, and ATP across all the $\varepsilon$ values, and TraCS-D also outperforms them when $\varepsilon \gtrsim 3$. Moreover, the performance advantage of TraCS becomes more significant as $m$ increases to $60 \times 60$. This is because finer-grained discretization allows TraCS's rounding to align with the sensitive locations more accurately. Statistically, the mean AE of TraCS-C is 66.1% of NGram, 57.1% of L-SRR, and 63.8% of ATP in Figure 6.8a, and the proportions are 57.6%, 52.6%, and 61.1% in Figure 6.8b. The mean AE of TraCS-D is 76.9% of NGram, 66.4% of L-SRR, and 74.2% of ATP in Figure 6.8a, and the proportions are 71.2%, 65.0%, and 75.5% in Figure 6.8b. These results indicate the advantage of TraCS for synthetic trajectories in discrete spaces.

### Error Comparison on Real-world Datasets

Figure 6.9 presents the error comparison on the TKY and CHI datasets. Compared to the synthetic dataset, NGram, L-SRR, and ATP achieve lower errors, particularly when $\varepsilon$ is small. However, as $\varepsilon$ increases, TraCS still outperforms them.

**Reasons for the results.** (i) ATP performs exceptionally well when $\varepsilon$ is small, particularly on the TKY dataset. This is because this dataset contains many "condensed" locations, i.e. no other locations near the trajectory. In such cases, if the bi-direction perturbation of ATP is accurate, the perturbed location is error-free. (ii) The AE of ATP does not decrease significantly as $\varepsilon$ increases, which is consistent with their paper. NGram shows a similar trend, especially on the TKY dataset. This is due to the low performance of the Exponential mechanism for large location spaces, particularly when the locations are concentrated in a small area. In such cases, the score function struggles to differentiate the scores among the locations. L-SRR exhibits a similar trend. (iii) TraCS exhibits larger AE than them when $\varepsilon$ is small because it focuses on the rectangular area formed by the longitude and latitude, instead of a set of discrete locations in the cities (visualized in Figure D.8). Therefore, the perturbed locations may be far from the city center (where benefits discrete mechanisms) when $\varepsilon$ is small.



(a) TKY dataset (b) CHI dataset

Figure 6.9: Comparison on real-world datasets. TraCS operates on rectangular areas encompassing the city, which may result in larger AE when $\varepsilon$ is small.

### Time Cost Comparison

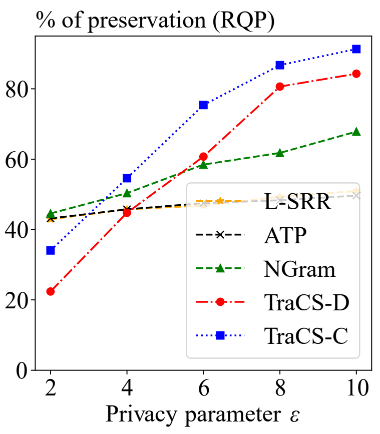
Table 6.2 presents the time cost comparison on the TKY dataset. The time cost is measured in milliseconds and averaged per location. It shows that TraCS-D and TraCS-C are significantly faster than NGram, ATP, and L-SRR. Compared to NGram and ATP, this efficiency is primarily due to the piecewise-based mechanisms in TraCS, which require negligible time for perturbation compared to the Exponential mechanism. Compared to L-SRR, TraCS-D and TraCS-C are also faster because they do not require grouping locations.

ATP has the highest time cost because it needs to process two copied trajectories and merges them. NGram has a lower time cost than ATP but is still significantly higher than TraCS. The perturbation procedure accounts for 86.3% and 92.0% of the total time cost in ATP and NGram, respectively. L-SRR has the lowest time cost among the three state-of-the-art methods due to its efficient sampling mechanism, however, its grouping procedure still incurs a non-negligible time cost.

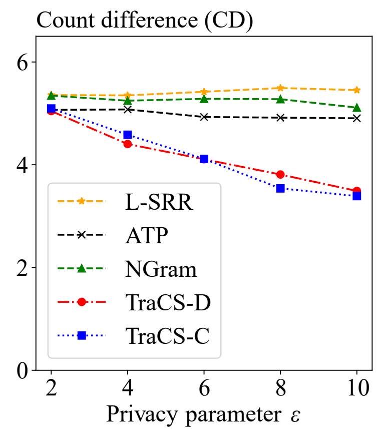
The total time cost of TraCS is less than 0.05% of ATP and NGram, and less than 1% of L-SRR. The perturbation procedure of TraCS-D and TraCS-C only takes 30% and 6% time, respectively. TraCS-D has a higher time cost than TraCS-C because it needs to calculate the distance space $\mathcal{D}_{r(\varphi)}$ for each location in the trajectory, while TraCS-C can use the same $\mathcal{D}_a$ and $\mathcal{D}_b$ for all locations.

### Experimental Results for Other Metrics

This subsection evaluates the performance of TraCS and existing methods under two additional metrics: range query preservation (Formula 17 in [34]) and hotspot preservation (Section 6.2.4



(a) Range query preservation



(b) Hotspot preservation

Figure 6.10: Comparison of other trajectory metrics on the CHI dataset. Higher range query preservation and lower error in hotspot preservation indicate better performance.

in [181]). Given a threshold $\delta$, a perturbed location is considered correct if it lies within a $\delta$ distance of the corresponding sensitive location. Formally, for a trajectory $\mathcal{T}$ and its perturbed trajectory $\mathcal{T}'$, the range query preservation (RQP) is defined as:

$$RQP(\mathcal{T}, \mathcal{T}') = \frac{1}{|\mathcal{T}|} \sum_{i=1}^{|\mathcal{T}|} \mathbf{1}\{\|\tau_i - \tau_i'\|_2 \leq \delta\} \cdot 100\%,$$

where $\mathbf{1}\{\cdot\}$ is the indicator function. A higher RQP indicates better trajectory utility. Hotspot preservation measures how many hotspots from a given set remain after perturbation. Given a set of hotspots $\mathcal{H}$, the locations in $\mathcal{H}$ are considered preserved if their perturbed locations after rounding are also in $\mathcal{H}$. Formally, the count difference (CD) of a trajectory in hotspot preservation is defined as:

$$CD(\mathcal{T}, \mathcal{T}') = \sum_{i=1}^{|\mathcal{T}|} \mathbf{1}\{\tau_i \in \mathcal{H}\} - \sum_{i=1}^{|\mathcal{T}|} \mathbf{1}\{\tau_i \in \mathcal{H} \wedge \tau_i' \in \mathcal{H}\},$$

where $\mathbf{1}\{\cdot\}$ is the indicator function. A smaller CD indicates better trajectory utility. We can find the relationship between AE and these two metrics. For range query preservation with a given threshold $\delta$, when $AE \leq \delta$, the perturbed location is expected to satisfy the range query. For hotspot preservation a smaller AE leads to better hotspot retention after rounding in TraCS.

We further evaluate the performance of TraCS-D and existing methods under varying privacy parameters $\varepsilon$ using these two metrics. Specifically, we use the CHI dataset and adopt the following experimental setups:

- Range query preservation: We set the threshold $\delta = 0.1$, and compute the RQP averaged over all trajectories.

- Hotspot preservation: The hotspot set $\mathcal{H}$ is defined as the first 20% frequent locations in the CHI dataset and we compute the CD averaged over all trajectories.

The results are shown in Figure 6.10. It can be seen that TraCS generally outperforms existing methods across different privacy levels, demonstrating better effectiveness in preserving both range queries and hotspots. Statistically, (i) the mean RQP across all $\varepsilon$ values for L-SRR, ATP, and NGram are 47.1%, 46.8%, and 56.5%, respectively, while the mean RQP of TraCS-D and TraCS-C are 58.6% and 68.4%, respectively. (ii) The mean CD across all $\varepsilon$ values for L-SRR, ATP, and NGram are 5.4, 4.9, and 5.2, respectively, while the mean CD of TraCS-D and TraCS-C are both 4.1. Particularly, TraCS shows improvement in hotspot preservation compared to existing methods across all privacy levels.

#### Assigning $\varepsilon$ for a Whole Trajectory

We also evaluate TraCS when the privacy parameter $\varepsilon$ is specified for an entire trajectory rather than per location. Specifically, for TraCS, L-SRR, NGram, and ATP, we set $\varepsilon$ at the trajectory level and allocate it uniformly across locations by assigning each location a parameter of $\varepsilon/|\mathcal{T}|$, where $|\mathcal{T}|$ is the trajectory length. This uniform allocation is simple and requires no additional information, and we consider it the most robust choice. In contrast, non-uniform allocations (e.g. assigning a larger parameter to utility-critical locations) may improve utility, but they typically rely on public notions of "importance" and can weaken privacy protection for those locations. Appendix D.2.10 provides results and discussion on this evaluation, showing that TraCS outperforms existing methods in trajectory utility under this setting when $\varepsilon$ is large.

### 6.3.3 Summary of Evaluation

From the evaluation results on both continuous and discrete spaces, we can summarize the following findings:

- In continuous spaces, TraCS generally exhibits better trajectory utility than the strawman method, particularly when the privacy parameter $\varepsilon$ is large. Meanwhile, TraCS-C has better utility than TraCS-D for common-shape location space and trajectories.
- In discrete spaces, TraCS generally outperforms NGram, L-SRR, and ATP, particularly when the privacy parameter $\varepsilon$ is large. Furthermore, TraCS has significantly lower time cost, requiring less than 1% of their time cost.

These findings validate the effectiveness and efficiency of TraCS in collecting trajectory data under LDP in both continuous and discrete spaces.

## 6.4 Related Work

This section reviews related trajectory collection methods under LDP and other quantifiable privacy notions. More privacy-preserving trajectory collection methods can be found in recent surveys [17,113].

### Trajectory Collection under LDP

We have disscussed three state-of-the-art trajectory collection methods under pure LDP: NGram [34], L-SRR [148], and ATP [181]. Among them, L-SRR was originally applied for origin-destination location pairs, while NGram, ATP, and this dissertation focus on trajectory perturbation. NGram attempts to ensure pure LDP for the "trajectory space", also named as trajectory-level or instance-level LDP in surveys [17,113], which is a stronger privacy than the location space. However, this is proved extremely challenging as the trajectory space increases exponentially with the number of possible locations in the location space. L-SRR, ATP and this dissertation focus on ensuring pure LDP for the location space. Unlike L-SRR, ATP and other existing methods based on discrete mechanisms, this dissertation focuses on continuous location spaces, which are more expressive than any fine-grained discrete location space.

Beyond the above three methods that design LDP mechanisms for trajectory collection, several works leverage external knowledge to improve the utility of LDP-based trajectory collection. WF-LDPSR [94] adaptively determines each user's privacy protection level based on the sensitivity of the user's information, and applies a water-filling strategy to allocate privacy budgets during perturbation. Regional popularity (i.e. the popularity of different regions in the location space) can also be exploited to constrain trajectory boundaries and improve utility [176]. In addition, recent work studies poisoning attacks against NGram and ATP [74], where adversaries induce malicious users to submit poisoned trajectories to the data collector with the goal of promoting target patterns.

Another line of research is trajectory *synthesis* for privacy-preserving trajectory publication. A typical work is LDPTrace [40], which synthesizes trajectories using a generative model. LDPTrace discretizes the location space into cells and represents each trajectory as a sequence of cells. Transition patterns between neighboring cells are perturbed using LDP mechanisms and collected by an untrusted curator. The curator then trains a Markov chain model on the perturbed transition patterns to capture movement behaviors. Following LDPTrace, ADGTrace [18] also employs Markov chain models for trajectory synthesis; it does not provide LDP guarantees and leverages personal features to improve utility. Compared to methods that perturb each trajectory, such as NGram, L-SRR, and ATP, synthetic trajectories may be overly too random and not reflect the original trajectories if not personalized enough.

### LDP Mechanisms for Bounded Numerical Domains

The perturbation mechanisms used in TraCS for direction and distance perturbation are utility-optimized piecewise-based mechanisms in OGPM [184]. Other LDP mechanisms for bounded numerical domains [73, 97, 160] can also be incorporated into TraCS; refer to Section 6.2.5 for discussions and comparisons. Compared with OGPM, which focuses on optimizing the 1D mechanisms' data utility, the design of 2D mechanisms in TraCS needs to consider the unique characteristics of trajectory data and 2D spaces. In particular, the direction-distance perturbation in TraCS-D to guarantee LDP for a rectangular region requires careful design. Meanwhile, this dissertation emphasizes the overlooked fundamental issues inherent in discrete-space LDP mechanisms for trajectory collection [34, 148, 181], which make privacy, utility, and efficiency discretization-dependent and prevent guarantees for continuous spaces. TraCS demonstrates that designing mechanisms directly in continuous space (via building blocks like OGPM, etc.) avoids these issues and subsumes discrete cases via rounding.

### Trajectory Collection under DP

As a weaker privacy notion than LDP, (central) DP [45, 48] assumes a trusted curator who processes the raw data and releases the perturbed data. It requires that only neighboring locations in the location space be indistinguishable, rather than any two arbitrary locations. Compared to LDP, the concept of "neighboring" in DP has many interpretations. A comprehensive survey on DP-based trajectory collection and publication is available in [113].

The use of direction information in trajectory collection dates back to an early work SDD [77], which attempts to add unbounded Laplace noise to the direction information. It provides a DP guarantee for unbounded continuous location spaces, but results in poor utility. The notion of "n-gram" in trajectory collection dates back to [25], where it abstracts a trajectory pattern as a gram. The idea of hierarchical decomposition can also be found in [177], where a spatial decomposition method is used for trajectory synthesis.

External knowledge is also widely used in DP-based trajectory collection methods. NGram's prior work [33] provides a DP-based solution for trajectory synthesis by exploiting the public knowledge of the road network. Focusing on particular privacy-concerned locations, or sensitive zones [149], can also improve the trajectory utility while maintaining partial privacy guarantees.

#### Trajectory Collection under Other Privacy Notions

A widely used relaxed privacy notion than LDP for trajectory collection is Geo-indistinguishability [7, 111], where the indistinguishability between two locations is a function of their distance, i.e. privacy can decrease with distance. Input-discriminative LDP [67] goes one step further, assigning each location its own privacy level. Threshold-integrated LDP [172] defines a threshold to constrain the sensitive region near a location. This is similar to NGram's reachability constraint but with a formal privacy definition. More relaxed LDP notions for trajectory collection can be found in paper survey [113]. Although these relaxed LDP notions generally provide better trajectory utility than pure LDP, they sacrifice the strong privacy guarantee of pure LDP, and often need hyperparameters to define the privacy level.

## 6.5 Conclusions

This chapter presents TraCS-D and TraCS-C, providing local differential privacy guarantees for trajectory collection in continuous location spaces. TraCS-D uses a novel direction-distance perturbation procedure, while TraCS-C perturbs the Cartesian coordinates of each location. These two methods can also be applied to discrete location spaces by rounding each perturbed location to the nearest discrete point, achieving better efficiency than existing discrete methods. Trajectory utility of TraCS is analyzed theoretically and evaluated empirically. Evaluation results on discrete location spaces show that TraCS outperforms state-of-the-art discrete methods in trajectory utility, particularly when the privacy parameter is large.

In summary, TraCS demonstrates that collecting trajectory data in continuous spaces generally better discrete approaches, avoiding privacy and efficiency issues caused by discretization, while can also be rounded to discrete spaces when necessary.

# Chapter 7

# Quantification of Classifier Utility under LDP

## 7.1 Preliminaries

This section formulates the problem, reviews LDP and the concept of classifier robustness.

### 7.1.1 Problem Formulation

Given a trained classifier $h$ that provides public services, users must submit their data $x$ as input to utilize the classifier. However, users' data $x$ may contain sensitive information, and the third-party classifier $h$ may not be fully trusted to protect data privacy. To address this concern, users perturb their data $x$ using an LDP mechanism $\mathcal{M}_\varepsilon$ before sending it to the classifier $h$. The perturbation mechanism $\mathcal{M}_\varepsilon$ acts as an interface between users and the classifier, forming a trust boundary by presenting the perturbed data $\mathcal{M}_\varepsilon(x)$ to the classifier $h$.

While the perturbed data $\mathcal{M}_\varepsilon(x)$ protects user privacy, it may degrade the utility of the classifier $h$, potentially leading to incorrect outputs where $h(\mathcal{M}_\varepsilon(x)) \neq h(x)$. This degradation is undesirable for both the classifier provider and the users. Thus, a critical question arises: *How can classifier designers or users quantify the utility of the classifier $h$ under the perturbation of $\mathcal{M}_\varepsilon$?* This chapter aims to answer this question by introducing a theoretical framework to quantify the utility of $h$ under the perturbation of $\mathcal{M}_\varepsilon$ w.r.t. the privacy parameter $\varepsilon$.

### 7.1.2 Sensitivity in LDP Mechanisms

Some LDP mechanisms, such as the Laplace and Gaussian mechanisms [48], were originally designed for query functions on raw data. These mechanisms require the *sensitivity* of the query function to determine the scale of perturbation. The (global) sensitivity of a function $f : \mathcal{X} \to \mathsf{Range}(f)$ is defined as the maximum change in $f$ for any two data:

$$\Delta f = \max_{x_1, x_2 \in \mathcal{X}} \left| f(x_1) - f(x_2) \right|.$$

In the context of LDP, the privacy of the data $x$ itself must be ensured, as the perturbation is applied directly to the data. Here, the query function $f$ is the identity function $f(x) = x$, whose sensitivity is the maximum length of $\mathcal{X}$. For classifiers, the input space $\mathcal{X}$ is typically normalized to $\mathcal{X} = [0, 1]$, resulting in a sensitivity of 1.

### 7.1.3 Classifier and Robustness

**Definition 11** (Classifier)**.** *A classifier $h : \mathbb{R}^d \to \{1, 2, \ldots, K\}$ is a function that maps input data $x \in \mathbb{R}^d$ to a specific label $h(x) \in \{1, 2, \ldots, K\}$.*

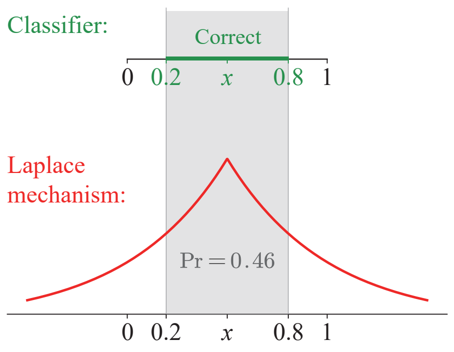
Based on representation complexity, classifiers can be categorized as follows: a *closed-form classifier* is represented by a closed-form function, such as a linear classifier; a *non-closed-form classifier* lacks a closed-form representation, such as a neural network. In both cases, if we know the classifier's structure and parameters, we call it a *white-box classifier*; otherwise, it is a *black-box classifier.*

**Definition 12** (Local robustness)**.** *Denote $B_\theta(x)$ as the $\ell_\infty$-norm ball centered at $x$ with radius $\theta$.*[*] *A classifier $h$ is $\theta$-locally-robust at $x$ if:*

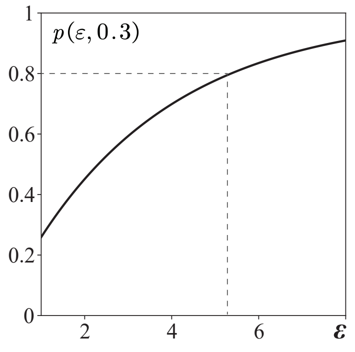
$$\forall \tilde{x} \in B_\theta(x),\ h(\tilde{x}) = h(x).$$

Intuitively, local robustness means that the classifier $h$ is insensitive to small perturbations around the input $x$. The largest feasible $\theta$ is referred to as the *robustness radius* of the classifier $h$ at $x$. Robustness radius serves as a key metric for evaluating the classifier's stability to unseen data.

[*]The $\ell_\infty$-norm is a common choice for measuring data perturbations and can be converted to other norms using norm inequalities. Here, $B_\theta(x)$ denotes the set of points $\tilde{x} \in \mathbb{R}^d$ where each dimension of $\tilde{x}$ is within a distance $\theta$ of the corresponding dimension of $x$, i.e. $B_\theta(x) \coloneqq \{\tilde{x} : |\tilde{x}^{(i)} - x^{(i)}| \leq \theta\}$.



(a) Example illustration.



(b) Classifier utility w.r.t. $\varepsilon$.

Figure 7.1: Illustration of the example and utility quantification of the classifier under the Laplace mechanism for $x = 0.5$.

## 7.2 Illustrating the Framework: An Example

This section illustrates the utility quantification framework with an intuitive example. Specifically, we consider a classifier under the Laplace mechanism and present a utility quantification statement.

**Definition 13** (The Laplace mechanism [48])**.** *For any $x \in [0, 1]$, the Laplace mechanism $\mathcal{M}_\varepsilon$ is defined as: $\mathcal{M}_\varepsilon(x) = x + \xi$, where $\xi \sim Lap(1/\varepsilon)$ are random variables drawn from Laplace distribution with mean 0 and scale $1/\varepsilon$.*

In the Laplace mechanism, $\mathcal{M}_\varepsilon(x)$ is unbounded due to the Laplace distribution. Since the classifier is typically defined on a bounded domain, such as $[0, 1]$, a common practice is to truncate the perturbed data to this range, i.e. $\mathcal{M}_\varepsilon(x)$ is truncated to $[0, 1]$.[†]

### 7.2.1 Example

Consider a classifier $h : [0, 1] \to \{1, 2\}$ with one-dimensional input and two classes, defined as:

$$h(x) = \begin{cases} 1, & \text{if } x \in [0.2, 0.8], \\ 2, & \text{otherwise.} \end{cases}$$

[†]This truncation can be treated as a post-processing step by the users that does not leak privacy [48, 64, 156]. We will also examine mechanisms with bounded perturbations in Section 7.3.1, where no truncation is needed. There are also truncated Laplace mechanisms [31, 59] that directly sample from the truncated Laplace distribution. It is also applicable to our framework; we use the original Laplace mechanism on $[0, 1]$ as an example for simplicity.

Consider a specific sensitive data $x = 0.5$, which will be perturbed using the Laplace mechanism $\mathcal{M}_\varepsilon$ before submitting it to the classifier. As a result, the classifier's output, $h(\mathcal{M}_\varepsilon(0.5))$, becomes a random variable due to the randomness of $\mathcal{M}_\varepsilon$.

To quantify the utility of $h(\mathcal{M}_\varepsilon(0.5))$ with respect to $\varepsilon$, we analyze two key properties: the concentration of $\mathcal{M}_\varepsilon$ and the robustness of $h$.

**Concentration analysis.** Denote region $B_\theta(x)$ as the $\theta$-neighborhood ball of $x$. For instance, $B_{0.3}(0.5) \coloneqq \{\tilde{x} : |\tilde{x} - 0.5| \leq 0.3\}$. Then the perturbed data $\mathcal{M}_\varepsilon(0.5)$ falls into $B_{0.3}(0.5)$ with a probability $p(\varepsilon, \theta = 0.3)$. Specifically, denote $F_{\text{Lap}}(\theta)$ as the cumulative distribution function (CDF) of the Laplace distribution used in $\mathcal{M}_\varepsilon$. We have:

$$\begin{aligned} p(\varepsilon, \theta = 0.3) &= \Pr[\mathcal{M}_\varepsilon(x) \in B_{0.3}(x)] = F_{\text{Lap}}(0.3) - F_{\text{Lap}}(-0.3) \\ &= 2F_{\text{Lap}}(0.3) - 1 = 1 - e^{-0.3\varepsilon}. \end{aligned}$$

Therefore, we can state that: $\mathcal{M}_\varepsilon(0.5)$ outputs a value within $B_{0.3}(0.5)$ with probability $1 - e^{-0.3\varepsilon}$. For instance, if we set $\varepsilon = 2$, we have $p(2, 0.3) = 0.46$. Figure 7.1a shows the illustration of this instance.

**Robustness analysis.** Analyzing the utility of $h(\mathcal{M}_\varepsilon(0.5))$ requires analyzing the robustness of $h$ under the input range $B_\theta(0.5)$. We can see that $h$ is robust under the input range $B_{0.3}(0.5) = [0.2, 0.8]$ according to its definition. Then, we can state that: $h$ preserves the correct classification result under the input range $B_{0.3}(0.5)$.

**Utility quantification.** Combining the above, we can quantify the utility of $h(\mathcal{M}_2(0.5))$: *with probability at least* $p(2, 0.3) = 0.46$*, $h$ preserves the correct classification result under* $\mathcal{M}_2(0.5)$. Importantly, this utility quantification can be extended to any $\varepsilon$ using the closed-form expression of $p(\varepsilon, \theta)$.

**Application.** For this classifier, the theoretical utility quantification $p(\varepsilon, 0.3)$ can guide the choice of the privacy parameter $\varepsilon$ in the Laplace mechanism to achieve a desired utility level. Figure 7.1b shows $p(\varepsilon, 0.3)$ as a function of $\varepsilon$. For instance, to ensure the classifier preserves the correct result with probability $p(\varepsilon, 0.3) \geq 0.8$, privacy parameter $\varepsilon$ should be set to at least 5.2.

### 7.2.2 Further Steps

The above example has demonstrated an analytical approach to quantify the utility of a classifier under the Laplace mechanism. However, this example is limited in scope as it focuses on a specific noise-adding mechanism and assumes a white-box classifier with a known robustness radius. In practice, several challenges arise: (i) Diverse LDP mechanisms. Not all LDP mechanisms behave as noise-adding mechanisms, which means the concentration analysis may differ significantly. (ii) Variety of classifiers. Classifiers come in various forms, and determining their robustness radius often requires different techniques. (iii) Conservative utility quantification. The utility provided in this example is a lower bound under strong pure LDP constraints and a conservative robustness radius. It can be refined, especially for higher-dimensional classifiers.

The subsequent sections will address these challenges by introducing a general framework for utility quantification and refinement techniques. Specifically, we will cover the following aspects:

- *General LDP mechanisms.* We model an LDP mechanism as a general distribution function and derive its concentration analysis $p(\varepsilon, \theta)$. As examples, we explore commonly used continuous and discrete mechanisms.
- *Unified method for finding robustness radius.* We treat classifiers as black-box functions and adapt a hypothesis testing framework to determine their robustness radius. This approach provides a unified method applicable to classifiers with different structures and access types (white-box or black-box).
- *Refinement techniques.* We refine the utility quantification by incorporating the concepts of robustness hyperrectangles and PAC privacy.

## 7.3 Formal Framework

This section presents the formal framework for quantifying utility of classifiers under various types of LDP mechanisms.

### 7.3.1 Concentration Analysis of LDP

In Definition 5 of LDP, the mechanism $\mathcal{M}(x)$ represents a family of probability distributions determined by the input $x$. Thus, the perturbed output $\tilde{x} = \mathcal{M}(x)$ is a random variable. The concentration property characterizes how the probability mass of the perturbed output $\tilde{x}$ concentrates around the original input $x$, and can be stated as follows.

**Proposition 2** (Concentration property of LDP)**.** *Let $\mathcal{M}(x)$ be an $\varepsilon$-LDP mechanism applied to an input $x \in \mathbb{R}$, and let $F_{\mathcal{M}}$ denote the CDF of the perturbed output $\mathcal{M}(x)$. The concentration property of $\mathcal{M}(x)$ is given by:*

$$\forall a < b : \Pr[a \leq \mathcal{M}(x) \leq b] = F_{\mathcal{M}}(b) - F_{\mathcal{M}}(a).$$

This proposition follows directly from the definition of the CDF and provides a probabilistic characterization of the "boundedness" of the LDP mechanism $\mathcal{M}(x)$. As a special case, if $a = x - \theta$ and $b = x + \theta$, it becomes $\Pr[\mathcal{M}(x) \in B_\theta(x)]$. For practical mechanisms designed for better utility, $\mathcal{M}(x)$ is typically concentrated around the input $x$. This implies that $F_{\mathcal{M}}(b) - F_{\mathcal{M}}(a)$ captures most of the probability mass when $x \in [a, b]$. We analyze the concentration properties of typical continuous and discrete LDP mechanisms in the following subsections.

### Continuous Mechanisms

When the data domain is continuous, the LDP mechanism corresponds to a continuous distribution. Based on whether the perturbation depends on the input $x$, continuous mechanisms are categorized into: (i) Noise-adding mechanisms, which add noise independent of the input $x$. (ii) Piecewise-based mechanisms, which sample outputs from input-dependent piecewise distributions.

**Noise-adding Mechanisms.** These mechanisms achieve LDP by adding carefully designed noise to the input data.

**Definition 14.** *A noise-adding LDP mechanism* $\mathcal{M} : \mathcal{X} \to \mathsf{Range}(\mathcal{M})$ *is defined as:* $\mathcal{M}(x) = x + \xi$*, where* $\xi$ *is a random variable (noise) drawn from a symmetric distribution with mean* $0$*.*

Noise-adding mechanisms are input-independent, using the same noise distribution for all inputs. Examples include the Laplace and Gaussian mechanisms [48].

**Concentration Analysis.** For a noise-adding mechanism $\mathcal{M}(x)$, the probability of outputting a perturbed value $\mathcal{M}(x) \in B_\theta(x)$ is:

$$F_{\mathcal{M}}(x + \theta) - F_{\mathcal{M}}(x - \theta) = 2 \cdot F_{\mathcal{M}}(\theta) - 1,$$

where $F_{\mathcal{M}}$ is the CDF of the noise distribution. The last equality holds due to the symmetry of the noise distribution, making the result independent of $x$. For instance, the Laplace mechanism's concentration analysis is given by:

$$2F_{\text{Lap}}(\theta) - 1 = 1 - e^{-\theta\varepsilon}.$$

In the previous section, we have seen that for $\varepsilon = 2$ and $\theta = 0.3$, the concentration probability is 0.46. This indicates that while the Laplace mechanism is defined on the whole real line, it is concentrated in a small region around $x$.

**Piecewise-based Mechanisms.** These mechanisms achieve LDP by sampling outputs from carefully designed piecewise distributions that vary based on the input $x$.

**Definition 15.** *A piecewise-based LDP mechanism* $\mathcal{M}(x) : \mathcal{X} \to \mathsf{Range}(\mathcal{M})$ *is defined as:*

$$\text{pdf}[\mathcal{M}(x) = \tilde{x}] = \begin{cases} p_\varepsilon & \text{if } \tilde{x} \in [l_{x,\varepsilon}, r_{x,\varepsilon}], \\ p_\varepsilon / e^\varepsilon & \text{otherwise,} \end{cases}$$

*where* $p_\varepsilon$ *is the sampling probability, and* $[l_{x,\varepsilon}, r_{x,\varepsilon}]$ *define the sampling interval based on* $x$ *and* $\varepsilon$*.*

Piecewise-based mechanisms are input-dependent, meaning the sampling distribution changes for different inputs $x$. Examples include PM [150] and SW [95], which use different $p_\varepsilon$ and intervals $[l_{x,\varepsilon}, r_{x,\varepsilon}]$. New mechanisms can be designed by modifying these parameters.

**Concentration Analysis.** For a piecewise-based mechanism $\mathcal{M}(x)$, the probability of outputting a perturbed value $\mathcal{M}(x) \in B_\theta(x)$ depends on $\theta$. The concentration analysis is given by:

$$\begin{cases} 2\theta \cdot p_\varepsilon & \text{if } B_\theta(x) \subseteq [l_{x,\varepsilon}, r_{x,\varepsilon}], \\ (r_{x,\varepsilon} - l_{x,\varepsilon}) p_\varepsilon + (2\theta - (r_{x,\varepsilon} - l_{x,\varepsilon})) \frac{p_\varepsilon}{e^\varepsilon} & \text{otherwise.} \end{cases}$$

This analysis is $x$-dependent due to the terms $l_{x,\varepsilon}$ and $r_{x,\varepsilon}$. For example, in the PM mechanism with $p_\varepsilon = e^{\varepsilon/2}$, if $B_\theta(x) \subseteq [l_{x,\varepsilon}, r_{x,\varepsilon}]$, the probability is $2\theta \cdot e^{\varepsilon/2}$. For $\varepsilon = 2$ and $\theta = 0.3$, the concentration probability $\mathcal{M}(x) \in B_\theta(x)$ is 0.86, indicating that the PM mechanism is more concentrated than the Laplace mechanism under these parameters.

### Discrete Mechanisms

When the data domain is discrete and finite, the LDP mechanism corresponds to a discrete distribution. Many real-world datasets are discrete, such as image pixel values (e.g. 256 levels) or integer-valued attributes like age.

**Definition 16.** *If* $|\mathcal{X}| < \infty$*, a discrete LDP mechanism* $\mathcal{M}(x) : \mathcal{X} \to \mathcal{X}$ *is defined as:*

$$\Pr[\mathcal{M}(x) = \tilde{x}_i] = p_\varepsilon(\tilde{x}_i, x), \quad \tilde{x}_i \in \mathcal{X},$$

*where* $p_\varepsilon(\tilde{x}_i, x)$ *is the probability of perturbing* $x$ *to* $\tilde{x}_i$*, determined by* $\varepsilon$*,* $x$*, and* $\tilde{x}_i$*.*

Examples of such mechanisms include the $k$-RR mechanism [78], where $p_\varepsilon(\tilde{x}_i, x)$ is uniform except for $\tilde{x}_i = x$, and the Exponential mechanism [48], where $p_\varepsilon(\tilde{x}_i, x)$ depends on the distance between $\tilde{x}_i$ and $x$.

**Concentration Analysis.** For discrete mechanisms, the support is a finite set $\mathcal{X}$, and the CDF is defined as: $F_\mathcal{M}(\tilde{x}) = \sum_{\tilde{x}_i \le \tilde{x}} p_\varepsilon(\tilde{x}_i, x)$. The probability of $\mathcal{M}(x)$ outputting a value in $B_\theta(x)$ is:

$$F_\mathcal{M}(x + \theta) - F_\mathcal{M}(x - \theta) = \sum_{\tilde{x}_i \in [x-\theta, x+\theta]} p_\varepsilon(\tilde{x}_i, x).$$

For instance, in the $k$-RR mechanism:

$$p_\varepsilon(\tilde{x}_i, x) = \frac{e^\varepsilon}{|\mathcal{X}| - 1 + e^\varepsilon} \text{ if } \tilde{x}_i = x \text{ otherwise } \frac{1}{|\mathcal{X}| - 1 + e^\varepsilon}.$$

If $\mathcal{X}$ discretizes $[0, 1]$ evenly, i.e. $\mathcal{X} = \{0, 1/|\mathcal{X}|, 2/|\mathcal{X}|, \ldots, 1\}$, the concentration analysis becomes:

$$\sum_{\tilde{x}_i \in [x-\theta, x+\theta]} p_\varepsilon(\tilde{x}_i, x) = \frac{e^\varepsilon}{|\mathcal{X}| - 1 + e^\varepsilon} + \frac{2\theta|\mathcal{X}| - 1}{|\mathcal{X}| - 1 + e^\varepsilon}.$$

When $|\mathcal{X}| = 100$, $\varepsilon = 2$, and $\theta = 0.3$, the concentration probability of the $k$-RR mechanism is 0.63.

#### Comparison of Concentration Properties

Fixing $\varepsilon = 2$ and $\theta = 0.3$, the concentration probabilities of the mentioned three mechanisms are:

- Laplace mechanism: $p(\varepsilon = 2, \theta = 0.3) = 0.46$;
- PM mechanism: $p(\varepsilon = 2, \theta = 0.3) = 0.86$;
- $k$-RR mechanism: $p(\varepsilon = 2, \theta = 0.3) = 0.63$.

These results highlight that different LDP mechanisms are differently concentrated under the same privacy parameter $\varepsilon$ and region $B_\theta(x)$. This variation is critical for analyzing classifier utility and selecting appropriate LDP mechanisms. For instance, if a classifier's robustness radius $\theta$ is 0.3, and we set privacy parameter $\varepsilon = 2$, the PM mechanism achieves the highest probability of $\mathcal{M}(x) \in B_{0.3}(x)$, making it the best choice in this scenario.

### 7.3.2 Robustness Analysis of Classifier

We analyze the robustness of classifiers under the concept of *probabilistic robustness*, a probabilistic approximation of deterministic robustness.[‡] Probabilistic robustness suffices for utility analysis under LDP mechanisms, as both the robustness and concentration properties of LDP mechanisms are probabilistic. Compared to deterministic robustness, it is computationally efficient and applicable to any classifier, including black-box classifiers, thus suitable as a general framework.

[‡]If the classifier is white-box, deterministic robustness can be derived analytically or through verification methods. See Appendix E.2.1 for details. In the most ideal case, the deterministic robustness can be expressed as a closed-form function of arbitrary $x$. We omit such easier cases for generality.

**Probabilistic Robustness**

Hypothesis testing provides an intuitive perspective for probabilistic robustness. We can state two counter hypotheses:

- $H_0$: $h$ is robust under $B_\theta(x)$.
- $H_1$: $h$ is not robust under $B_\theta(x)$.

These hypotheses can be statistically tested by querying the classifier on random samples in $B_\theta(x)$.[§] If robustness holds for almost all samples, $H_0$ is accepted with high confidence. Formally, probabilistic robustness is defined as follows:

**Definition 17** (Probabilistic robustness [144, 180])**.** *Given a classifier $h$, an input $x$, a radius $\theta$, and a tolerance $\tau > 0$, $h$ is $\theta$-locally-robust at $x$ with probability at least $1-\omega$ if*

$$\Pr_{\tilde{x}\sim B_\theta(x)}\left[1-\Pr[h(\tilde{x})=h(x)]\leq\tau\right]\geq 1-\omega,$$ [¶]

*where $\tilde{x}$ is a random variable drawn from $B_\theta(x)$.*

This definition implies that the event $h(\tilde{x}) = h(x)$ is almost always true (within a tolerance $\tau$) with probability at least $1-\omega$. Here, $1-\omega$ also corresponds to the confidence level (type I error) in the hypothesis testing. For minimal $\tau$ and $\omega$, e.g. $\tau = 0.01$ and $\omega = 0.05$, $h$ is nearly $\theta$-locally robust at $x$. Deterministic robustness is a special case with $\tau = 0$ and $\omega = 0$.

Probabilistic robustness is typically computed using the Hoeffding bound [144, 180], which provides a probabilistic guarantee for the mean of a random variable. Let $Z$ be the indicator function of $h(\tilde{x}) = h(x)$, i.e. $Z = 1$ if robust and $Z = 0$ otherwise. Then, the empirical mean $\hat{\mu}_Z$ (robust frequency) is calculated by sampling $n$ points, and the true robustness probability $\mu_Z$ can be derived using the Hoeffding bound.

**Theorem 15** (Hoeffding bound [72])**.** *For any $\tau \geq 0$ and $\omega > 0$, the probability $\omega$ that the empirical mean ($\hat{\mu}_Z$) is within $\tau$ of the true mean ($\mu_Z$) is bounded by:*

$$\Pr_{Z_i\sim P_Z}\left[|\hat{\mu}_Z-\mu_Z|\leq\tau\right]\geq 1-\omega,$$

[§] Robustness analysis is a different scenario from *using* the classifier, and does not leak users' privacy. Section 7.5 provides further discussions.

[¶] The inner probability $\Pr[h(\tilde{x}) = h(x)]$ is taken over the uncertainty of robustness, refer to Appendix E.2.2 for explanation.

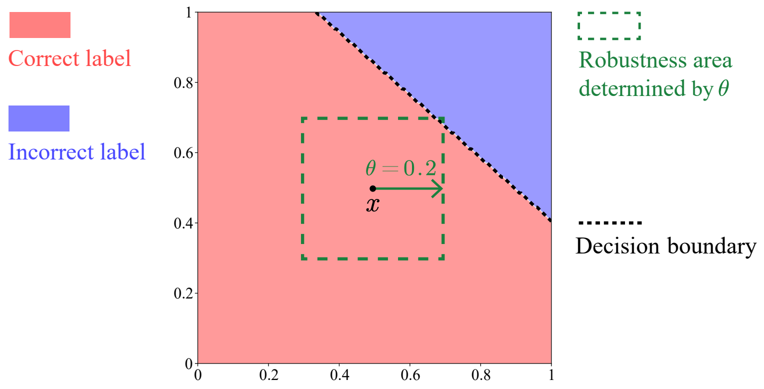


Figure 7.2: Robustness radius $\theta$ of a 2D classifier at $x$ and the tested decision boundary by brute force.

*where* $\omega = 2e^{-2n\tau^2}$ *and* $n$ *is the number of samples. Equivalently, at least* $n(\omega, \tau) = (\ln \frac{2}{\omega})/(2\tau^2)$ *samples are required to ensure the bound.*

The number of samples $n$ balances utility guarantees and computational cost. A smaller $\omega$ (more deterministic guarantee) requires more samples $n$. Given $\tau$ and $\omega$, we check if the empirical mean $\hat{\mu}_Z$ on $n(\omega, \tau/2)$ samples satisfies $\hat{\mu}_Z \in [1 - \tau/2, 1]$. If this is true, by the Hoeffding bound, $|\mu_Z - \hat{\mu}_Z| \leq \tau/2$, which implies $\mu_Z \in [1 - \tau, 1]$. Thus, Definition 17 holds, and we can claim that $h$ is robust under $B_\theta(x)$ with probability at least $1 - \tau$, and our confidence level is at least $1 - \omega$.

**Independent of the classifier.** The above method treats the classifier as a black-box function, making it applicable to any classifier $h$ without requiring knowledge of its internal structure or parameters.

### Procedure for Finding $\theta$

After setting tolerance $\tau$ and confidence level $\omega$, we can find a probabilistic guaranteed robustness radius by the following procedure. The key idea is to increase $\theta$ from 0 until the misclassification rate exceeds $\tau/2$. Specifically,

1. Compute the required number of samples $n(\omega, \tau/2)$ using the Hoeffding bound in Theorem 15.
2. For the current $\theta$, uniformly sample $n$ points from $B_\theta(x)$, then query the classifier and calculate the misclassification rate.
3. If the misclassification rate is below $\tau/2$, increase $\theta$ and repeat the second step.

As $\theta$ increases, the misclassification rate will eventually exceed $\tau/2$. The maximum $\theta$ before this point is the robustness radius. In practice, binary search is often used in the third step to reduce the complexity to a logarithmic scale.

**Example 11.** *Given a 2D neural network classifier $h : [0,1]^2 \to \{1,2\}$, treated as a black-box, and parameters $\tau = 0.02$ and $\omega = 0.05$, we determine the robustness radius $\theta$ at $x = (0.5, 0.5)$ using the described procedure. Specifically: (i) the required number of samples for testing is $n(\omega, \tau/2) \approx 1.8 \times 10^4$; (ii) by testing $\theta \in [0,1]$, we find the robustness radius to be $\theta = 0.20$.*

*Figure 7.2 illustrates this example. The true decision boundary of $h$ at $x$ is shown by the black dashed line, computed via brute force. The green dashed box represents the robustness area $B_\theta(x)$ determined by the found $\theta = 0.20$. This result closely touches the true decision boundary.*

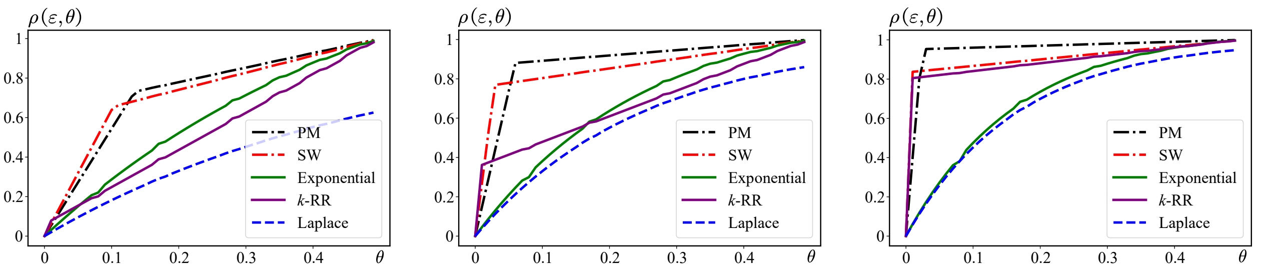
In the above example, any perturbation within the robustness region $B_\theta(x)$ ensures that the classifier's output remains unchanged, thereby preserving utility. When the perturbation is performed by an LDP mechanism, the perturbed value $\tilde{x}$ is drawn from $B_\theta(x)$ with a specific probability. The next subsection establishes the connection between $B_\theta(x)$ and this probability to quantify the classifier's utility under LDP mechanisms.

### 7.3.3 Utility Quantification

By combining the concentration analysis of LDP mechanisms and the robustness analysis of classifiers, we derive a utility quantification for a given classifier under an LDP mechanism w.r.t. any privacy parameter $\varepsilon$. This quantification framework provides a systematic evaluation of classifier performance under LDP mechanisms. In contrast, the empirical approach requires re-evaluation for different $\varepsilon$ and fails to establish analytical relationships between utility and $\varepsilon$.

Formally, by Definition 17, the classifier is robust under $B_\theta(x)$ with tolerance $\tau$ and confidence level $1-\omega$. Therefore, under the same confidence level $(1-\omega)$, a probabilistic estimate of the event $h(\tilde{x}) = h(x)$, i.e. exact robustness without tolerance, is at least $1-\tau$. Meanwhile, the LDP mechanism $\mathcal{M}$ perturbs $x$ into $B_\theta(x)$ with probability $F_{\mathcal{M}}(x+\theta) - F_{\mathcal{M}}(x-\theta)$. Thus, the utility guarantee of the classifier under the LDP mechanism is quantified by the product of these two probabilities. Specifically, given a $d$-dimensional classifier $h$ and raw data $x$, denote $\mathcal{M}^d(x)$ as the perturbed $d$-dimensional input by $d$ independent $\mathcal{M}$.[‖] We claim: *Under confidence level at least $1-\omega$, with probability at least $\rho(\varepsilon, \theta)$, the classifier $h$ preserves the correct classification result under*

[‖]For simplicity, we assume $\mathcal{M}$ is applied to all dimensions of $h$. In practice, it may only be applied to a subset of sensitive features.



(a) $\varepsilon = 2$ (b) $\varepsilon = 4$ (c) $\varepsilon = 6$

Figure 7.3: Comparison of the robustness probability $\rho(\varepsilon, \theta)$ for different LDP mechanisms with $\varepsilon = 2, 4, 6$. Details of the mechanisms and discussions on the curves of $\rho(\varepsilon, \theta)$ are provided in Appendix E.2.3.

*the input perturbed by $\mathcal{M}^d$, where*

$$\begin{aligned}\rho(\varepsilon, \theta) &= \Pr[\mathcal{M}^d(x) \in B_\theta(x)] \cdot \Pr_{\tilde{x}\sim B_\theta(x)}[h(\tilde{x}) = h(x)] \\ &\geq \left(F_{\mathcal{M}}(x+\theta) - F_{\mathcal{M}}(x-\theta)\right)^d \cdot (1-\tau),\end{aligned}$$

*with $F_{\mathcal{M}}$ the CDF of the LDP mechanism $\mathcal{M}$.*

For a given mechanism $\mathcal{M}$, the term $F_{\mathcal{M}}(x+\theta) - F_{\mathcal{M}}(x-\theta)$ is a closed form w.r.t. $\varepsilon$ and $\theta$, directly computable from the mechanism's parameters. $\omega$ and $\tau$ are predetermined based on Definition 17 of robustness. We fix $\omega = 0.05$ and $\tau = 0.01$ for all subsequent analyses. Thus, with the known robustness radius $\theta$ of the classifier $h$ at $x$, $\rho(\varepsilon, \theta)$ provides an immediate utility quantification for any $\varepsilon$.

Furthermore, if the distribution of $x$ as $P_x$ is known, we can extend the utility quantification to provide average-case (expected) and worst-case guarantees:[**]

$$\rho_{\text{avg}}(\varepsilon) = \mathbb{E}_{x\sim P_x}[\rho(\varepsilon, \theta)], \text{and } \rho_{\text{wor}}(\varepsilon) = \min_{x\sim P_x}[\rho(\varepsilon, \theta)].$$

Such guarantees provide input-distribution-aware utility quantification for the classifier. For example, the average-case utility guarantee can be stated as follows: *On average, under confidence level at least $1-\omega$,*[††] *with probability at least $\rho_{\text{avg}}(\varepsilon)$, the classifier $h$ preserves the correct classification result for an input $x \sim P_x$ perturbed by $\mathcal{M}^d$.*

**Application 1: Select the best LDP mechanism.** A direct application of this utility quantification is comparing the classifier's utility under different LDP mechanisms to select the best

[**]Evaluating average-case and worst-case utility requires knowledge of the input distribution $P_x$. In most papers, analyses of worst-case utility for their proposed mechanisms (defined on a specific domain) implicitly assume $P_x$ is defined on that domain with non-zero probability. In practice, $P_x$ can be estimated from historical data or domain knowledge.

[††]For clarity, we omit the confidence level in subsequent statements, as it is a predefined constant with $1-\omega \approx 1$.

one. Mechanisms with higher $\rho(\varepsilon, \theta)$ yield higher utility guarantees for classifiers. However, this varies with $\theta$ for different classifiers and $\varepsilon$ for different mechanisms. In fact, there is not a single best mechanism for all classifiers under all $\varepsilon$.

**Example 12.** *Figure 7.3 shows the utility guarantee $\rho(\varepsilon, \theta)$ for different LDP mechanisms, with $\theta$ ranging from* 0.1 *to* 0.5 *and at $x = 0.5$. Detailed instantiations of these mechanisms are provided in Appendix E.2.3. A higher $\rho(\varepsilon, \theta)$ indicates a higher utility for classifiers with robustness radius $\theta$. It is evident that no single mechanism is universally optimal for all $\varepsilon$ and $\theta$. For instance, when $\varepsilon$ and $\theta$ are small (e.g. $\varepsilon = 2$ and $\theta \leq 0.1$), the SW mechanism performs best. As $\varepsilon$ and $\theta$ increase, the PM mechanism becomes the superior choice.*

**Takeaway results.** Although no mechanism is universally optimal for all classifiers (with a given robustness radius $\theta$) under all $\varepsilon$, heuristic guidelines can assist in mechanism selection. From the above example and Figure 7.3, we observe that if the privacy parameter $\varepsilon$ is not too small ($\geq 2$), and the classifier's robustness radius $\theta$ is not too small ($\geq 0.1$), then the PM mechanism is generally the best choice. Experiments in Section 7.6 further support this conclusion for higher-dimensional classifiers.

**Application 2: Select $\varepsilon$ for a utility requirement.** With a chosen mechanism, the utility quantification can guide the selection of the privacy parameter $\varepsilon$ to meet a desired utility guarantee. For example, if the classifier's robustness radius $\theta$ is 0.3, and we require the classifier to preserve the correct classification result with probability $\rho(\varepsilon, \theta) \geq 0.8$ under the PM mechanism, we can set $\varepsilon = 1.8$.

**Impact of dimensionality.** For high-dimensional classifiers, the primary challenge lies in the (sensitive) input dimension $d$. The utility guarantee $\rho(\varepsilon, \theta)$ decreases exponentially with $d$, and capturing the robustness of high-dimensional classifiers becomes more difficult, making the utility quantification less meaningful.

The second part of the contribution addresses this dimensionality issue. By introducing more precise robustness analysis methods and relaxing the privacy constraint, we refine the utility guarantee $\rho(\varepsilon, \theta)$, making the utility quantification more practical for high-dimensional classifiers.

## 7.4 Refinement Techniques

This section introduces two techniques to refine the utility quantification presented in Section 7.3.3. The key observation is that the utility guarantee $\rho(\varepsilon, \theta)$ is a conservative lower bound. It is an at-least guarantee based on the robustness radius under pure $\varepsilon$-LDP. To refine it, we propose the following techniques.

- *Robustness refinement.* We generalize the robustness radius to a robustness hyperrectangle, which assigns distinct radii to different dimensions, enabling a more precise computation of $\rho(\varepsilon, \theta)$.
- *Privacy relaxation.* We relax the pure $\varepsilon$-LDP constraint to $(\varepsilon, \delta)$-PAC LDP, which tolerates a small probability of failure. Under this relaxation, we introduce a $\delta$-probabilistic indicator for applying the $\varepsilon$-LDP mechanism and propose an extended Gaussian mechanism tailored for PAC LDP.

### 7.4.1 Robustness Hyperrectangle

To refine the utility quantification, we introduce the notion of a robustness hyperrectangle, which provides a more precise description of the robustness area.

The commonly used robustness radius for classifiers [52] is a conservative measure of robustness. It defines the robustness area as an $\ell_\infty$-ball,[‡‡] where all dimensions share the same radius. However, in practice, different dimensions often have varied robustness radii, making the true robustness area larger than that defined by an $\ell_\infty$-ball. In our context, accurately describing the robustness area is crucial for refining utility quantification. To this end, we define the robustness hyperrectangle, which accounts for the robustness of each dimension individually. The formal definition is as follows.

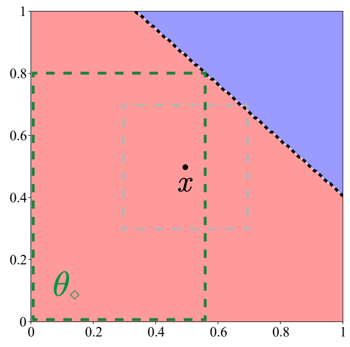
**Definition 18** (Robustness hyperrectangle)**.** *Given a $d$-dimensional classifier $h$ and a data point $x$, the robustness hyperrectangle $\theta_\diamond := [a_1, b_1] \times [a_2, b_2] \times \cdots \times [a_d, b_d]$ is a $d$-dimensional hyperrectangle that includes $x$ and satisfies*

$$\forall \tilde{x} \in \theta_\diamond,\ h(\tilde{x}) = h(x).$$

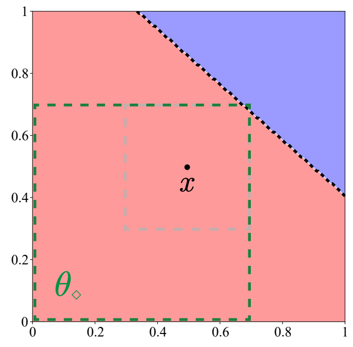
The robustness radius is a special case of the robustness hyperrectangle, where each dimension has the same size $\theta$ around $x$. More generally, we say that the classifier $h$ is $\theta_\diamond$-locally robust at $x$.

**Finding robustness hyperrectangle.** The robustness hyperrectangle $\theta_\diamond$ is not unique, unlike the robustness radius. Different methods for finding $\theta_\diamond$ may yield different results. Heuristically, we aim to include the robustness radius to ensure an improved utility quantification.

[‡‡]Other $\ell_p$-balls, such as $\ell_1$-ball and $\ell_2$-ball, are also used, but they are also symmetric across all dimensions.



(a) $\theta_\diamond = [0, 0.55] \times [0, 0.8]$.



(b) $\theta_\diamond = [0, 0.7] \times [0, 0.7]$.

Figure 7.4: Two robustness hyperrectangles (green dashed boxes) for the classifier in Figure 7.2. The original robustness radius $\theta = 0.2$ corresponds to the gray dashed box.

After determining the robustness radius $\theta$ using the procedure in Section 7.3.2, we initialize the robustness area of each dimension as $[a_i, b_i]$ with $\theta$. We then try to expand $[a_i, b_i]$ along the $a_i$ or $b_i$ directions. This process outputs a maximal robustness hyperrectangle $\theta_\diamond$ that satisfies the definition in Definition 18.

**Example 13.** *Figure 7.4 illustrates two examples of robustness hyperrectangles for the classifier in Example 11. The original robustness radius is $\theta = 0.2$, corresponding to the hyperrectangle* $[0.4, 0.6] \times [0.4, 0.6]$ *(gray dashed boxes). Two different hyperrectangles (green dashed boxes) are shown, both larger than the original robustness radius. Figure 7.4a partially includes the robustness radius, while Figure 7.4b fully encompasses it. The proposed procedure will obtain the hyperrectangle depicted in Figure 7.4b.*

Each dimension of $\theta_\diamond$, represented as $[a_i, b_i]$, influences the concentration analysis of LDP mechanisms. Larger $[a_i, b_i]$ allows for greater perturbation, leading to a stronger utility guarantee. Specifically, the concentration level $\theta$ in the utility guarantee $\rho(\varepsilon, \theta)$ is replaced with the refined $\theta_\diamond$, resulting in the improved utility guarantee $\rho(\varepsilon, \theta_\diamond)$.

**Refined utility quantification.** Given a $d$-dimensional classifier $h$, LDP mechanism $\mathcal{M}_\varepsilon$ for each dimension, and private data $x$. Denote the utility quantification $\rho(\varepsilon, \theta)$ as defined in Section 7.3.3. It can be refined to: *With probability at least $\rho(\varepsilon, \theta_\diamond)$, the classifier $h$ preserves the correct classification result under the input perturbed by $d$ independent $\mathcal{M}_\varepsilon$ (i.e. pure $d\varepsilon$-LDP), where*[§§]

$$\rho(\varepsilon, \theta_\diamond) = \prod_{i=1}^{d} F_{\mathcal{M}}(b_i) - F_{\mathcal{M}}(a_i),$$

[§§]We omit the terms $(1 - \omega)$ and $(1 - \tau)$ related to $\rho(\varepsilon, \theta_\diamond)$ for simplicity in presentation. They are almost 1 in practice (e.g. $\omega = 0.05, \tau = 0.01$) and thus also negligible.

*where $[a_i, b_i]$ represents the i-th dimension of the hyperrectangle $\theta_\diamond$.*

### 7.4.2 PAC Privacy

Probably approximately correct (PAC) privacy [167] provides a formal framework for relaxing the worst-case privacy guarantee. While pure LDP requires the privacy constraint to hold in all cases, PAC privacy allows a small probability of failure, i.e. probabilistic LDP. This relaxation allows the use of a wider range of mechanisms, notably the Gaussian mechanism, which is widely used in privacy-preserving machine learning.

To simplify notation, we denote the privacy loss [48] between $x_1$ and $x_2$ at $\tilde{x}$ as

$$\mathcal{L}_{\mathcal{M},x_1,x_2}(\tilde{x}) \coloneqq \ln\left(\frac{\Pr[\mathcal{M}(x_1) = \tilde{x}]}{\Pr[\mathcal{M}(x_2) = \tilde{x}]}\right).$$

Pure $\varepsilon$-LDP can then be expressed as: $\mathcal{L}_{\mathcal{M},x_1,x_2}(\tilde{x}) \leq \varepsilon$ for all $x_1, x_2, \tilde{x}$. In contrast, $(\varepsilon, \delta)$-PAC LDP is defined by a relaxed $\delta$-probabilistic condition.¶¶

**Definition 19** (PAC LDP, adopted from [167]). *A randomized mechanism $\mathcal{M} : \mathcal{X} \to \mathsf{Range}(\mathcal{M})$ satisfies $(\varepsilon, \delta)$-PAC LDP if*

$$\forall x_1, x_2, \tilde{x} : \Pr[\mathcal{L}_{\mathcal{M},x_1,x_2}(\tilde{x}) \leq \varepsilon] \geq 1 - \delta,$$

*i.e. $\mathcal{M}$ satisfies $\varepsilon$-LDP with probability at least $1 - \delta$.*

Pure LDP is a special case of $(\varepsilon, \delta)$-PAC LDP with $\delta = 0$. Although PAC LDP allows a $\delta$-probabilistic failure, the adversary can only infer the original data with probability at most $\delta$ when $\varepsilon$-LDP fails.

**Combination of PAC LDP mechanisms.** When applying $d$ independent PAC LDP mechanisms to $d$ dimensions, if we follow the combination theorem from Dwork [48], it gives $(d\varepsilon, d\delta)$-PAC LDP. However, this result is not tight, as it follows the failure probability $d\delta$ can exceed 1, which is impossible. We provide a tighter result for combining $d$ independent PAC LDP mechanisms.

**Theorem 16** (Combination of $(\varepsilon, \delta)$-PAC LDP). *Given $d$ independent mechanisms $\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_d$ that satisfy $(\varepsilon, \delta)$-PAC LDP, their combination satisfies $(d\varepsilon, 1 - (1 - \delta)^d)$-PAC LDP.*

*Proof.* (Sketch) The result of $\delta$ comes from computing the total failure probability of $d$ independent mechanisms. Appendix E.1.3 provides the details. □

¶¶Readers may wonder about the relationship between $(\varepsilon, \delta)$-PAC LDP and $(\varepsilon, \delta)$-LDP. In fact, the traditional $(\varepsilon, \delta)$-LDP from Dwork [48] is ambiguous and has many interpretations. Concentrated privacy [44] from Dwork, Renyi privacy [114], and PAC privacy [167] can address this ambiguity. See Appendix E.2.4 for a detailed discussion.

With the notion of PAC LDP, we present two techniques to improve the utility quantification. The first is a privacy indicator, which extends any pure LDP mechanism to PAC LDP. The second is an extended Gaussian mechanism, which cannot satisfy pure LDP but satisfy PAC LDP.

### Privacy Indicator

The failure probability in $(\varepsilon, \delta)$-PAC LDP allows us to design a $\delta$-probabilistic indicator for the application of the pure LDP mechanism. Specifically, we define

$$\mathcal{I}_\delta(\mathcal{M}(x)) = \begin{cases} \mathcal{M}(x) & \text{w.p. } 1-\delta, \\ x & \text{w.p. } \delta, \end{cases}$$

i.e. with probability $1-\delta$, the mechanism $\mathcal{M}$ is applied, and with probability $\delta$, the original data $x$ is returned. The mechanism $\mathcal{I}_\delta(\mathcal{M}(x))$ satisfies $(\varepsilon, \delta)$-PAC LDP.

**Theorem 17.** *Assume $\mathcal{M}$ is a randomized mechanism that satisfies $\varepsilon$-LDP. Then the mechanism $\mathcal{I}_\delta(\mathcal{M}(x))$ defined above satisfies $(\varepsilon, \delta)$-PAC LDP.*

*Proof.* (Sketch) The proof follows directly from the definition of $\mathcal{I}_\delta(\mathcal{M}(x))$. Appendix E.1.1 provides the details. □

**Privacy indicator for multiple mechanisms.** For a $d$-dimensional data $x$, instead of applying the privacy indicator to each dimension independently, we can treat $d$ mechanisms applied to each dimension as a single mechanism designed for $d$-dimensional data, denoted as $\mathcal{M}^d(x)$. The privacy indicator $\mathcal{I}_\delta(\mathcal{M}^d(x))$ then controls the application of all mechanisms together. By substituting the pure LDP mechanism $\mathcal{M}^d(x)$ with $\mathcal{I}_\delta(\mathcal{M}^d(x))$, we can refine the utility quantification while relaxing the privacy guarantee to $(\varepsilon, \delta)$-PAC LDP.

**Refined utility quantification.** Given a $d$-dimensional classifier $h$, combined LDP mechanism $\mathcal{M}^d_\varepsilon$, privacy indicator $\mathcal{I}_\delta(\cdot)$, and raw data $x$, the utility quantification $\rho(\varepsilon, \theta)$ can be refined to: *With probability at least $\delta + (1-\delta)\rho(\varepsilon, \theta_\diamond)$, the classifier $h$ preserves the correct result under the input perturbed by $\mathcal{I}_\delta(\mathcal{M}^d(x))$, which guarantees $(d\varepsilon, \delta)$-PAC LDP.*

The refined utility guarantee $\delta + (1-\delta)\rho(\varepsilon, \theta_\diamond)$ is always larger than $\rho(\varepsilon, \theta_\diamond)$, as it equals $\rho(\varepsilon, \theta_\diamond) + \delta(1-\rho(\varepsilon, \theta_\diamond))$, and the latter term is non-negative. Therefore, the new utility guarantee is always a refinement over the original one.

**Extended Gaussian Mechanism**

It is already known that the Gaussian mechanism [48] cannot satisfy pure $\varepsilon$-LDP, making the privacy indicator inapplicable. Nonetheless, Dwork has shown that it can satisfy $(\varepsilon, \delta)$-LDP for $\varepsilon \leq 1$. Under the notion of PAC LDP, we provide an extended Gaussian mechanism to work for any $\varepsilon$.*** The following theorem is the form for data domain $x \in [0, 1]$.

**Theorem 18** (Extended Gaussian mechanism)**.** *The Gaussian mechanism $\mathcal{M}_\varepsilon(x) = x + \mathcal{N}(0, \sigma^2)$ satisfies $(\varepsilon, \delta)$-PAC LDP for $x \in [0, 1]$ if $\sigma$ is defined as*

$$\sigma \geq \frac{\sqrt{2}}{2} \left( \frac{\sqrt{\ln(2/\delta) + \varepsilon} + \sqrt{\ln(2/\delta)}}{\varepsilon} \right).$$

*Proof.* (Sketch) The proof generally follows the structure of Dwork et al. [48], but uses a rewritten privacy loss and a different tail bound for the Gaussian distribution. Appendix E.1.2 provides the details and compares this bound with others. □

The above theorem applies to a single dimension. For a $d$-dimensional data, we can apply the Gaussian mechanism to each dimension with the same $\sigma$. Then the combination result from Theorem 16 provides a PAC privacy guarantee.

**Refined utility quantification.** Given a classifier $h$, combined Gaussian mechanism $\mathcal{M}^d$ with $\delta$, and $d$-dimensional private data $x$, the utility quantification $\rho(\varepsilon, \theta_\diamond)$ can be refined to: *With probability at least $\rho(\varepsilon, \theta_\diamond)$, the classifier $h$ preserves the correct classification result under the input perturbed by $\mathcal{M}^d$, which guarantees $(d\varepsilon, 1 - (1 - \delta)^d)$-PAC LDP.*

The extended Gaussian mechanism and the privacy indicator provide two approaches to achieve $(\varepsilon, \delta)$-PAC LDP. While both operate under the same privacy notion, the Gaussian mechanism actually provides a stronger privacy: when it fails to satisfy the pure $\varepsilon$-LDP (with probability $\delta$), the privacy loss exceeds $\varepsilon$ but remains bounded by a larger value. In contrast, the privacy indicator exposes the original data when it fails.

***Analytical Gaussian mechanism [11] works for any $\varepsilon$, but it is based on an alternative privacy definition and lacks analytical form of the noise scale $\sigma$. Appendix E.1.2 provides the detailed comparison.

## 7.5 Discussions

This section discusses detailed privacy questions in the utility quantification framework and the complexity of finding the robustness radius.

### Detailed Privacy Discussions

In the LDP model, it is assumed that the classifier is curious about private data but honest in reporting results, i.e. it outputs $h(\mathcal{M}(x))$ rather than fabricating results. This is a weak assumption, as any utility evaluation becomes meaningless if the classifier fabricates results.

Utility quantification, as a different scenario from *using* the classifier, focuses on a specific input $x$ but does not involve interaction with users' data. It can be conducted either by the classifier designer or by the users, and in both cases, there is no privacy leakage for the users: (i) If conducted by the classifier designer, e.g. for improving design, there is no interaction with users, and thus no privacy concerns arise. (ii) If conducted by the users, e.g. to achieve a better privacy–utility tradeoff for their data, they can obtain the robustness radii without revealing their original data. For white-box classifiers, users know the classifier parameters and can compute the robustness radii. For black-box classifiers, users can uniformly query the classifier over the entire input domain (i.e. prediction for nearly all input $x \in [0,1]^d$), without revealing their specific data. This one-time procedure yields robustness radii for all inputs and privacy parameters. Thus, the robustness radius at any input can be considered public knowledge, which is a reasonable assumption.

### Complexity of Finding the Robustness Radius

The procedure for finding a probabilistic robustness radius is independent of the classifier's structure and parameters. Its complexity arises from the iterations required to determine the robustness radius and the testing of $n$ perturbed data points in each iteration.

Using binary search to find the radius results in a complexity of $\Theta(\log(1/\kappa))$, where $\kappa$ is the precision of the radius. For instance, if $\kappa = 0.01$ (two decimal places), the complexity is $\Theta(\log 100) \approx \Theta(7)$. Meanwhile, classifiers are often implemented with parallel inference on GPUs, which allows for $\Theta(1) \times T$ complexity for $n$ data points, where $T$ is the inference time for a single data point, provided $n$ is not excessively large. Consequently, the total complexity for finding a two-decimal precision robustness radius is $\Theta(7T)$. In practice, $T$ is typically in the order of milliseconds, making the complexity of finding the robustness radius sufficiently low. Moreover, the robustness radius is a one-time computation for a given classifier, which can be used for all future utility quantification.

#### Per-instance Data Utility vs Aggregated Data Utility

The proposed utility quantification framework focuses on classifiers and per-instance data utility, providing utility guarantees at the level of individual data points. This stands in contrast to aggregated data utility analyses such as mean estimation, which evaluate the accuracy of aggregated statistics computed from the perturbed data of many users. Per-instance data utility is crucial for several reasons: (i) Classifiers and other personalized services like recommendation systems operate on individual data points. Their performance depends directly on per-instance utility and has no direct connection to aggregated utility. (ii) Per-instance data utility focuses on mechanism-level utility analysis, characterizing the privacy–utility tradeoff curves without relying on sample size or aggregation effects. These two reasons relate to the focus on per-instance data utility in this dissertation.

#### Fixed Classifier with Noisy Input vs Retrained Classifier with Noisy Data

These are two different scenarios. (i) In the former scenario, users locally apply an LDP mechanism to their data for privacy protection, agnostic to the service provider (i.e. the classifier), and want to understand the resulting utility of the existing classifier. (ii) In the latter scenario, the service provider applies LDP to users' data and retrains a new classifier tailored to the perturbed data, which relates to "robust training".

We focus on the former user-privacy-centered scenario; in the latter scenario, users expose their raw data to the service provider. Moreover, from users' perspective, the classifier is fixed at the time of use (i.e. when they query the service). If a retrained classifier is deployed, then the utility is for the retrained classifier (from users' perspective).

#### Robustness of LDP Mechanisms

Intuitively, LDP mechanisms with higher concentration around raw data are more prone to reconstruction (with sufficient reports) and poisoning attacks (as malicious data are less averaged). In our concentration analysis, PM, SW, and $k$-RR (especially with small $k$) are more concentrated, thus expected to be more vulnerable. This aligns with prior findings: PM and $k$-RR show more vulnerability to poisoning attacks [92, 93].

**Independent LDP Mechanisms vs Correlated LDP Mechanisms**

We adopt independent perturbations for cleanness of presentation. In practice, DP libraries (e.g. Google DP [65]) typically provide independent (L)DP primitives and do not explicitly support correlated perturbations. The essential reason is that the underlying data correlations are often unknown.

When the correlation coefficients are known or estimable, independent LDP can be extended via (i) post-processing of perturbed data to enforce a given correlation structure, or (ii) adopting correlated-perturbation mechanisms (e.g. [138, 147]; although they are primarily tailored to mean estimation). The proposed utility quantification framework also applies to such correlated mechanisms.

## 7.6 Case Studies

This section presents case studies of the proposed utility quantification framework. We use representative LDP mechanisms and classifiers as examples, though the proposed framework is generalizable to any LDP mechanism and classifier type.

### 7.6.1 Setup

We use the following datasets to train three types of classifiers.

- Stroke Prediction [53]: Predicts stroke likelihood based on features like age, hypertension, and BMI. This dataset contains 6 numerical input dimensions and a binary output. The sensitive features are "Age" and "BMI".
- Bank Customer Attrition [107]: Predicts whether a bank customer will leave based on features like age, salary, and tenure. This dataset has 9 numerical input dimensions and a binary output. The sensitive features are "Age" and "Estimated Salary".
- MNIST-7×7 (variant of [35]): Predicts handwritten digits (0 to 9) based on pixel values. This dataset has 49 input dimensions and a 10-class output. All dimensions are treated as sensitive features.

The first two datasets involve direct sensitive user inputs, while the third is a standard image classification dataset. For the first two datasets, we use the `sklearn` library with cross-validation to train Logistic Regression and Random Forest classifiers. These traditional classifiers are widely used in medical and financial domains due to their interpretability and lower data requirements [6]. For the third dataset, which consists of high-dimensional images, we use the `torch` library to train a Convolutional Neural Network (CNN) classifier. All datasets are normalized to $[0, 1]$ for training, as normalization aids classifier convergence and is a common practice.

Given a trained classifier, users apply LDP mechanisms to sensitive features (e.g. age, salary, or specific parts of an image) before sending their data to the classifier for prediction. We consider typical LDP mechanisms in literature, which are summarized in Table 7.1, along with the classifiers used.

We present theoretical utility quantification for the classifiers under the LDP mechanisms discussed above. This quantification is based on the robustness hyperrectangle $\theta_\diamond$ of the classifier at a randomly selected record, determined using the search method described in Section 7.4.1. We also compare

Table 7.1: Summary of LDP mechanisms, types of classifiers, and datasets used in the case studies.

| | |
|---|---|
| **Mechanism** | Laplace [48], Gaussian (Theorem 18), PM [150], SW [95], Exponential [121], $k$-RR [78] |
| **Classifier** | Low-dim: Logistic Regression, Random Forest; High-dim: Neural Network |
| **Dataset** | Low-dim: Stroke Prediction [53], Bank Customer Attrition [107]; High-dim: MNIST-7×7 (variant of [35]) |

the theoretical results with empirical utility observations. Specifically, for a given classifier and mechanism, we provide and compare:

- *Theoretical* $\rho(\varepsilon, \theta_\diamond)$: Derived using the proposed utility quantification framework.
- *Empirical* $\hat{\rho}(\varepsilon)$: Estimated by sampling $n = 2000$ instances $\{\tilde{x}\}_n$ from the LDP mechanism and testing the classifier for each $\varepsilon$.

The theoretical $\rho(\varepsilon, \theta_\diamond)$ is a closed-form expression that can be directly computed from the mechanism's parameters for any $\varepsilon$. It serves as a lower bound of the actual robustness probability and is generally smaller than $\hat{\rho}(\varepsilon)$. A smaller gap indicates more accurate theoretical utility quantification.

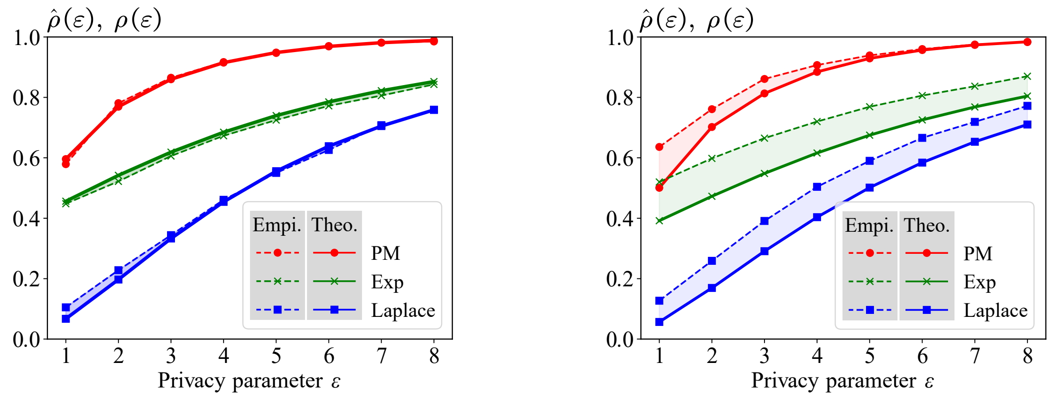
Without loss of generality, given the input data distribution $P_x$ determined by each dataset, we can also provide average-case and worst-case utility quantification: $\rho_{\text{avg}}(\varepsilon), \rho_{\text{wor}}(\varepsilon)$ and their empirical counterparts $\hat{\rho}_{\text{avg}}(\varepsilon), \hat{\rho}_{\text{wor}}(\varepsilon)$.

### 7.6.2 Utility Quantification Results

We present the utility quantification results for the classifiers under different LDP mechanisms.

#### Stroke Prediction

We begin by considering pure LDP for this medical dataset. For a randomly selected record "Age: 79, BMI: 24, Hypertension: 1, ...", we analyze theoretical and empirical utility for the classifiers under the Laplace, PM, and Exponential mechanisms. Additional records and mechanisms can be found in Appendix E.2.5.



(a) Logistic Regression. (b) Random Forest.

Figure 7.5: Empirical and theoretical utility for two classifiers trained on the Stroke Prediction dataset.

**Logistic Regression.** The robustness hyperrectangle at the given record is $\theta_\diamond = [0.63, 1] \times [0, 1]$ for this classifier. From this information, the utility guarantee under the PM mechanism can be directly derived as follows:

*For this trained Logistic Regression classifier on the Stroke Prediction dataset, at the record "Age: 79, Hypertension: 1, …, BMI: 24", with probability at least $\rho(\varepsilon, \theta_\diamond)$, the classifier preserves the correct prediction under the PM mechanism applied to "Age" and "BMI" (i.e. $2\varepsilon$-LDP), where*

$$\begin{aligned}\rho(\varepsilon, \theta_\diamond) =& [F_{\mathrm{PM}_{\varepsilon,\mathrm{Age}}}(1) - F_{\mathrm{PM}_{\varepsilon,\mathrm{Age}}}(0.63)] \cdot [F_{\mathrm{PM}_{\varepsilon,\mathrm{BMI}}}(1) - F_{\mathrm{PM}_{\varepsilon,\mathrm{BMI}}}(0)] \\ =& 1 - F_{\mathrm{PM}_{\varepsilon,\mathrm{Age}}}(0.63).\end{aligned}$$

The last equality holds because the CDF of the PM mechanism is 0 at 0, and 1 at 1. Term $F_{\mathrm{PM}_{\varepsilon,\mathrm{Age}}}(0.63)$ is the CDF of the PM mechanism (applied to the feature "Age") at 0.63, which can be computed from the PM mechanism's parameters, as detailed in Appendix E.2.5.

**Random Forest.** For this classifier, the robustness hyperrectangle at the same record is $\theta_\diamond = [0.50, 1] \times [0, 0.62]$. Based on this information, the utility guarantee under the PM mechanism can be derived as follows:

*For this trained Random Forest classifier on the Stroke Prediction dataset, at the record "Age: 79, Hypertension: 1, …, BMI: 24", with probability at least $\rho(\varepsilon, \theta_\diamond)$, the classifier preserves the correct prediction under the PM mechanism applied to "Age" and "BMI" (i.e. $2\varepsilon$-LDP), where*

$$\rho(\varepsilon, \theta_\diamond) = (1 - F_{PM_{\varepsilon,Age}}(0.5)) \cdot F_{PM_{\varepsilon,BMI}}(0.62).$$

Figure 7.5 compares the theoretical utility $\rho(\varepsilon, \theta_\diamond)$ with the empirical utility $\hat{\rho}(\varepsilon)$ for $\varepsilon$ values ranging from 1 to 8, applied to each sensitive feature. Both utilities increase with $\varepsilon$, with the PM mechanism

Table 7.2: Time cost comparison (in milliseconds).

| | PM | Exponential | Laplace |
|---|---|---|---|
| **Empirical**[a] | $6.56 + 1.38$ | $859.83 + 1.53$ | $11.29 + 1.53$ |
| **Theoretical**[b] | 0.24 | 0.94 | 0.30 |

[a] Time of 2000 samples + inference.

[b] Time to compute $\rho(\varepsilon, \theta_\diamond)$ only; computing $\theta_\diamond$ takes 1.20 ms but is a one-time cost amortized across all $\varepsilon$ values.

consistently providing the highest utility. This observation aligns with the takeaway that PM is the best choice for moderate $\varepsilon$ and $\theta$ values. Similarly, the Exponential mechanism outperforms the Laplace mechanism. For both classifiers, the theoretical utility $\rho(\varepsilon, \theta_\diamond)$ closely matches the empirical utility $\hat{\rho}(\varepsilon)$, particularly for the Logistic Regression classifier, showing the accuracy of the theoretical utility.

The Random Forest classifier exhibits a larger gap between $\rho(\varepsilon, \theta_\diamond)$ and $\hat{\rho}(\varepsilon)$ compared to the Logistic Regression classifier. This discrepancy arises from the robustness hyperrectangle $\theta_\diamond$, which is a conservative estimate of the overall robustness region (difficult to determine precisely). For Logistic Regression, the decision boundary is a hyperplane [136], making it more predictable and easier to approximate a hyperrectangle, resulting in a more accurate $\rho(\varepsilon, \theta_\diamond)$. Appendix E.2.5 provides additional details on decision boundaries.

**Time cost.** In addition to a sampling-independent theoretical guarantee, the proposed utility quantification also has an extremely low time cost. It does not require any sampling from the LDP mechanism, the utility statement can be directly computed for any $\varepsilon$ from the mechanism's parameters and the robustness hyperrectangle $\theta_\diamond$. Therefore, for mechanisms having high sampling complexity, e.g. the Exponential mechanism ($\mathcal{O}(m)$ with $m$ as the domain size), the theoretical utility quantification is much more efficient than the empirical one.

Table 7.2 shows the average time of computing one $\rho(\varepsilon, \theta_\diamond)$ and $\hat{\rho}(\varepsilon)$ for different mechanisms in this case study. For each empirical result $\hat{\rho}(\varepsilon)$, we sample 2000 times from the mechanism and feed the samples to the classifier for inference. The theoretical approach is significantly faster, especially for the Exponential mechanism. For this mechanism, computing one empirical $\hat{\rho}(\varepsilon)$ requires 859.83 milliseconds for 2000 samples, whereas computing one theoretical $\rho(\varepsilon, \theta_\diamond)$ takes only 0.94 milliseconds, which is less than 0.1%. The efficiency stems from directly substituting the privacy parameter $\varepsilon$ and the robustness hyperrectangle $\theta_\diamond$ into the closed-form expression of $\rho(\varepsilon, \theta_\diamond)$. Moreover, although $\rho(\varepsilon, \theta_\diamond)$ requires the robustness hyperrectangle, it can be computed once for all $\varepsilon$, which is much more efficient than the empirical $\hat{\rho}(\varepsilon)$ for each $\varepsilon$.

**Average-case and worst-case analysis.** All data points in the dataset collectively define a data distribution $P_x$ over the sensitive features. Therefore, we evaluate the utility at each $\{\text{Age, BMI}\} \sim P_x$, then compute the mean for the average-case utility and the minimum for the worst-case utility. This approach yields average-case and worst-case utility guarantees that account for the input data distribution. Comprehensive results are provided in Appendix E.2.5.

### Bank Customer Attrition

We now evaluate PAC LDP for this financial dataset. For a randomly selected record "Age: 22, Estimated Salary: 101,348, Credit Score: 619, ...", we analyze both theoretical and empirical utility for the two classifiers under PAC LDP mechanisms. Specifically, we apply the privacy indicator $\mathcal{I}_\delta$ to the Laplace, PM, and Exponential mechanisms, while also including the Gaussian mechanism. The failure probability is fixed at $\delta = 0.1$ for all mechanisms. Detailed utility quantification statements for the privacy indicator and Gaussian mechanism under PAC LDP are provided in Appendix E.2.6.

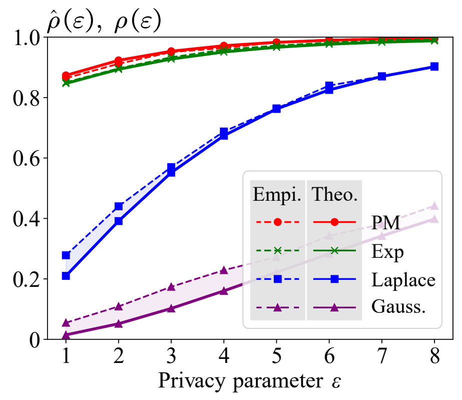
Figure 7.6 compares $\rho(\varepsilon, \theta_\diamond)$ and $\hat{\rho}(\varepsilon)$ for $\varepsilon$ values from 1 to 8 assigned to each sensitive feature. The results show a similar trend to the Stroke Prediction dataset, with the Exponential mechanism performing much better in this case, while the Gaussian mechanism lags behind the others.

Two key observations emerge in this case study: (i) The Exponential mechanism performs comparably to the PM mechanism for Logistic Regression. This is because its robustness hyperrectangle $\theta_\diamond = [0, 0.72] \times [0, 1]$ is large enough to cover most of the sensitive feature ranges, enabling excellent performance for all mechanisms defined on $[0, 1]$. Figure 7.3 illustrates this phenomenon: larger $\theta$ values lead to $\rho(\varepsilon, \theta)$ approaching 1 for all such mechanisms. However, Laplace and Gaussian mechanisms are less effective here due to their probability mass extending beyond $[0, 1]$. (ii) The Gaussian mechanism is not as good as the other mechanisms. The main reason is from the noise scale $\sigma$ in Theorem 18. Although we can extend the Gaussian mechanism to $\varepsilon \geq 1$, this leads to a larger $\sigma$, i.e. more noise added to the data.

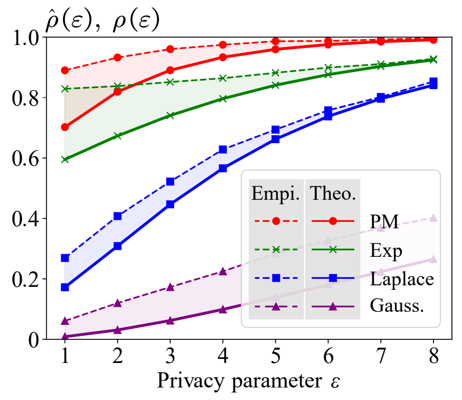
**Average-case and worst-case analysis.** We can similarly provide these utility guarantees as done for the Stroke Prediction dataset. Detailed results are presented in Appendix E.2.6.

### MNIST-7×7

We evaluate both theoretical and empirical utility under the PM, SW, Exponential, and $k$-RR mechanisms for the first two images in this dataset, labeled as "digit 5" and "digit 0." Consistent with the setup in previous case studies, we focus on PAC LDP by applying the privacy indicator



(a) Logistic Regression.



(b) Random Forest.

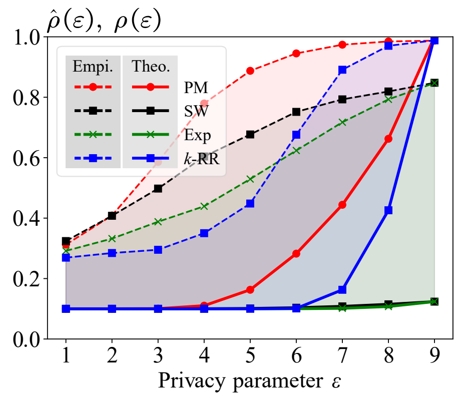
Figure 7.6: Empirical and theoretical utility for two classifiers trained on the Bank Customer Attrition dataset.

$\mathcal{I}_\delta$ to these mechanisms, with the failure probability fixed at $\delta = 0.1$. The Laplace and Gaussian mechanisms are excluded, as they are prone to perturbing pixel values outside the valid range $[0, 1]$ in this high-dimensional dataset, making fair comparisons with other mechanisms difficult.

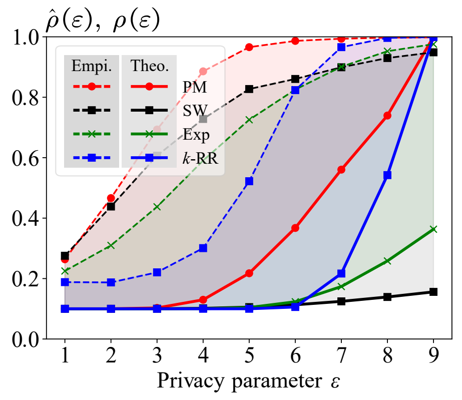
Figure 7.7 compares $\rho(\varepsilon, \theta_\diamond)$ and $\hat{\rho}(\varepsilon)$ for $\varepsilon$ values from 1 to 8 assigned to each pixel. Larger gaps between $\rho(\varepsilon, \theta_\diamond)$ and $\hat{\rho}(\varepsilon)$ are observed in this high-dimensional Neural Network classifier, especially for smaller $\varepsilon$ values. Even for the best mechanism, PM, the theoretical utility only starts to increase significantly when $\varepsilon > 4$, with an exponential growth rate. Similarly, the theoretical utility of the $k$-RR mechanism shows noticeable improvement when $\varepsilon > 6$. Despite the larger gap, the trends between theoretical and empirical results remain consistent: mechanisms with higher theoretical utility also exhibit higher empirical utility.

**Reasons for the results.** The larger gap between $\rho(\varepsilon, \theta_\diamond)$ and $\hat{\rho}(\varepsilon)$ in the Neural Network classifier compared to traditional classifiers arises from two factors: (i) The complex decision boundary of the Neural Network, which is difficult to approximate using a single hyperrectangle. (ii) Error accumulation in high-dimensional data, which is more pronounced than in low-dimensional data. Specifically, the Neural Network classifier partitions the high-dimensional space into intricate regions (one for each digit class), making it challenging to approximate with a single hyperrectangle around a digit. Additionally, theoretical utility is derived from the product of utility analyses for each pixel, making it more sensitive to error accumulation in high-dimensional data. In contrast, empirical utility is unaffected by the robustness hyperrectangle $\theta_\diamond$, avoiding such error accumulation.

**Improvements for high-dimensional classifiers.** (i) In practice, not all dimensions are sensitive and require privacy protection. While we treat all dimensions as sensitive in the MNIST-7×7 dataset, users can opt to apply LDP mechanisms to only a subset of dimensions, i.e. specific parts



(a) Neural Network on the first image in the dataset.



(b) Neural Network on the second image in the dataset.

Figure 7.7: Empirical and theoretical utility for a Neural Network classifier trained on the MNIST-7×7 dataset.

of the image. In such cases, the theoretical utility quantification can be more accurate. (ii) At present, we identify a connected (local) robustness rectangle that contains $x$; however, the classifier may admit additional disconnected and non-rectangular robust regions. One can use Monte Carlo sampling to discover such regions and then combine this with Monte Carlo integration to obtain a more accurate theoretical utility quantification for high-dimensional classifiers. Appendix E.2.7 provides details.

**Average-case and worst-case analysis.** We use the first 50 images from the MNIST-7×7 dataset as representative samples for analyzing the average-case and worst-case utility under LDP mechanisms. Appendix E.2.7 presents the results of this analysis.

### 7.6.3 Summary of Case Studies

We presented case studies of theoretical utility statements for different classifiers under typical LDP mechanisms and compared them with empirical utility. The key findings from the case studies are as follows:

- Theoretical utility quantification is computationally efficient, requiring *negligible* time due to its closed-form expression for any $\varepsilon$.
- For low-dimensional classifiers, theoretical utility often closely matches empirical utility. For high-dimensional classifiers, theoretical utility has a larger gap with empirical utility, due to complex decision boundaries and error accumulation. However, mechanisms with higher theoretical utility also exhibit higher empirical utility.

- The PM mechanism generally outperforms other mechanisms, providing the best utility across the case studies.

## 7.7 Related Work

This dissertation focuses on data utility of classifiers under perturbed data. The first subsection discusses closely related concepts—semantic perturbations, concentration properties of random variables, and classifier robustness—and their differences to this dissertation. This is followed by a discussion of broader related work.

### 7.7.1 Closely Related Concepts

**Semantic perturbations.** A number of the existing literature focuses on empirical evaluations of data utility under semantic perturbations [115], i.e. perturbations that involve semantic meanings. For image classifiers, such perturbations include brightness adjustments [13], blur [36], and data augmentation such as rotation and scaling [80, 124]. For text classification, examples include synonym replacement and random swap [86, 161], etc. For speech recognition, there are acoustic distortions [88]. These perturbations do not provide formal privacy guarantees, and some of them even preserve the semantic meanings of the original data.

This dissertation focuses on a specific category of mathematically tractable perturbations, LDP mechanisms. These perturbations, such as the Laplace or Gaussian noise, have weaker semantic meanings but provide provable privacy guarantees against adversaries. Additionally, there are variants of LDP mechanisms tailored for semantic perturbations, such as sentence embedding perturbation [39]. The LDP mechanisms examined in this dissertation are typical building blocks for numerous LDP protocols in practical applications.

**Concentration property of random variables.** The term concentration property mostly relates to the *concentration inequalities* in probability theory, e.g. Chernoff bound and Hoeffding inequality (see book [41]). These inequalities provide upper bounds on the probability that the sum of independent random variables deviates from its expected value. From these inequalities, the notion $(\alpha, \beta)$-accuracy [26, 41] is defined as: the error is bounded by $\alpha$ with probability at least $\beta$. In DP, this notion has been used to analyze the error of the summation query in the shuffle model [26]. It also appears in accuracy analysis of data exploring languages under DP guarantee, such as Haskell and SQL [58, 146].

In comparison to $(\alpha, \beta)$-accuracy, the theoretical utility quantification $\rho(\varepsilon, \theta)$ includes the concentration property of LDP mechanisms and the robustness property of classifiers. Among these two properties, the concentration property shares similarities with $(\alpha, \beta)$-accuracy by providing a probabilistic error bound for perturbed data, but it specifically focuses on the distribution of an LDP mechanism. Meanwhile, this property is used to connect with the robustness of classifiers in this dissertation.

**Robustness of classifiers.** Robustness refers to a classifier's ability to maintain performance (utility) under perturbations [52]. This property is crucial as classifiers often encounter data that deviates from their training distribution. Robustness also serves as a metric for evaluating a classifier's generalization ability [185]. Beyond semantic perturbations, robustness has been extensively studied under $\ell_p$-norm perturbations [52, 102, 103], where input data is perturbed within an $\ell_p$-norm ball. Within this perturbation model, various verification techniques have been developed to analyze classifier robustness. In addition to approximating the robustness of a classifier as described in Section 7.3.2, we can also approximate the classifier using a "smoothed" classifier via randomized smoothing [108] and then analyze the robustness radius of the smoothed classifier.

Unlike these studies, this dissertation focuses on classifier robustness under LDP perturbations. These perturbations can be viewed as occurring within *probabilistic* balls defined by the concentration property, controlled by the privacy parameter $\varepsilon$.

### 7.7.2 Broader Related Work

**Impact of (L)DP mechanisms on fairness of classifiers.** Fairness dictates that two individuals who are similar w.r.t. a particular task should be classified similarly. Under some metrics of similarity, it relates to the concept of DP [46]. DP mechanisms can impact the fairness of classifiers by changing biases in the input data distribution. For instance, if certain groups are over- or under-represented in the data shared with classifiers, it can lead to biased predictions after the application of DP mechanisms [8, 104].

**DP mechanisms for privacy-preserving classifier training.** This dissertation addresses data privacy when (L)DP mechanisms are applied by users before sharing data with classifiers, i.e. ensuring privacy during classifier usage. An orthogonal line of research examines privacy risks posed by classifiers themselves, focusing on data leakage during training. Such works aim to counter membership inference attacks [56, 139], which infer training data from model parameters. DP-SGD [4] is a well-known defense, applying Gaussian noise to gradient updates during training.

Federated learning [108] is a stricter privacy setting, where sensitive training data is distributed across multiple users and cannot be shared with a central server. In this context, DP-Fed-SGD [109] perturbs local gradients before aggregation to the central server. For a comprehensive overview, see a recent survey [57].

While these approaches focus on protecting training data privacy, this dissertation examines the utility of trained classifiers under input perturbations caused by LDP mechanisms. In the former case, the classifier is trusted and used for inference. In the latter, the classifier is untrusted, and users perturb their input data before sharing it.

**Utility of LDP mechanisms.** Data utility is the most crucial metric for LDP mechanisms. For simple queries such as mean, summation, and histogram [150, 153, 154], data utility is essentially determined by the theoretical mean squared error (MSE) of the mechanism. While empirical evaluations can be conducted by sampling from the mechanism, the empirical utility will converge to the theoretical MSE when the sample size is large enough. Meanwhile, MSE is also the most common metric in mechanism design and analysis.

However, a mechanism with lower MSE does not always guarantee better utility for all applications, particularly classifiers. This dissertation has shown that classifier utility depends on both the concentration property of the mechanism and the classifier's robustness. We provide a framework to quantify classifier utility under LDP mechanisms, offering an alternative to empirical evaluations.

## 7.8 Conclusions

This chapter introduces a framework for quantifying classifier utility under LDP mechanisms. The framework connects the concentration analysis of LDP mechanisms with the robustness analysis of classifiers. It provides guidelines for selecting the best mechanism and privacy parameter for a given classifier. Beyond the core framework, we introduce two novel refinement techniques that further improve utility quantification. Case studies on typical classifiers under LDP mechanisms demonstrate the framework's effectiveness and applicability.

# Bibliography

[1] MotionSense Dataset : Smartphone Sensor Data - HAR, 2018.

[2] Gurobi Help Center, 2024.

[3] Law of total variance, May 2024. Page Version ID: 1226284431.

[4] Martín Abadi, Andy Chu, Ian J. Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*, pages 308–318. ACM, 2016.

[5] Amazon. Ratings (beauty products). Downloaded from `https://www.kaggle.com/skillsmuggler/amazon-ratings`, August 2013.

[6] Nasrin Amini, Mahdi Mahdavi, Hadi Choubdar, Atefeh Abedini, Ahmad Shalbaf, and Reza Lashgari. Automated prediction of covid-19 mortality outcome using clinical and laboratory data based on hierarchical feature selection and random forest classifier. *Computer Methods in Biomechanics and Biomedical Engineering*, 26(2):160–173, 2023. PMID: 35297747.

[7] Miguel E. Andrés, Nicolás Emilio Bordenabe, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. Geo-indistinguishability: differential privacy for location-based systems. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *2013 ACM SIGSAC Conference on Computer and Communications Security, CCS'13, Berlin, Germany, November 4-8, 2013*, pages 901–914. ACM, 2013.

[8] Héber Hwang Arcolezi, Karima Makhlouf, and Catuscia Palamidessi. (local) differential privacy has NO disparate impact on fairness. In Vijayalakshmi Atluri and Anna Lisa Ferrara, editors, *Data and Applications Security and Privacy XXXVII - 37th Annual IFIP WG 11.3 Conference, DBSec 2023, Sophia-Antipolis, France, July 19-21, 2023, Proceedings*, volume 13942 of *Lecture Notes in Computer Science*, pages 3–21. Springer, 2023.

[9] Sergio Bacallado. Lecture 9: Classification, lda, qda.

[10] Borja Balle, James Bell, Adrià Gascón, and Kobbi Nissim. The privacy blanket of the shuffle model. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part II*, volume 11693 of *Lecture Notes in Computer Science*, pages 638–667. Springer, 2019.

[11] Borja Balle and Yu-Xiang Wang. Improving the gaussian mechanism for differential privacy: Analytical calibration and optimal denoising. In Jennifer G. Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 403–412. PMLR, 2018.

[12] Raef Bassily and Adam D. Smith. Local, private, efficient protocols for succinct histograms. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 127–135. ACM, 2015.

[13] Ridha Ilyas Bendjillali, Mohammed Beladgham, Khaled Merit, and Abdelmalik Taleb-Ahmed. Illumination-robust face recognition based on deep convolutional neural networks architectures. *Indonesian Journal of Electrical Engineering and Computer Science*, 18(2):1015–1027, 2020.

[14] Austin R. Benson, Ravi Kumar, and Andrew Tomkins. A discrete choice model for subset selection. In *Proceedings of the eleventh ACM International Conference on Web Search and Data Mining (WSDM)*. ACM, 2018.

[15] Stephen P Boyd and Lieven Vandenberghe. *Convex optimization.* Cambridge university press, 2004.

[16] Hai Brenner and Kobbi Nissim. Impossibility of differentially private universally optimal mechanisms. *SIAM J. Comput.*, 43(5):1513–1540, 2014.

[17] Erik Buchholz, Alsharif Abuadbba, Shuo Wang, Surya Nepal, and Salil S. Kanhere. Sok: Can trajectory generation combine privacy and utility? *Proc. Priv. Enhancing Technol.*, 2024(3):75–93, 2024.

[18] Hui Cai, Chen Lan, Biyun Sheng, Jian Zhou, Yuanyuan Yang, Yanmin Zhu, and Fu Xiao. Adgtrace: Achieving adaptive trajectory synthesis with generated data. *IEEE Trans. Mob. Comput.*, 25(1):148–163, 2026.

[19] T. Tony Cai, Yichen Wang, and Linjun Zhang. The cost of privacy: Optimal rates of convergence for parameter estimation with differential privacy. *CoRR*, abs/1902.04495, 2019.

[20] Clément L. Canonne, Gautam Kamath, and Thomas Steinke. The discrete gaussian for differential privacy. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[21] Bo Chen and Matthew T. Hale. The bounded gaussian mechanism for differential privacy. *J. Priv. Confidentiality*, 14(1), 2024.

[22] Hongge Chen, Huan Zhang, Si Si, Yang Li, Duane S. Boning, and Cho-Jui Hsieh. Robustness verification of tree-based models. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 12317–12328, 2019.

[23] Jianwei Chen, Huadong Ma, Dong Zhao, and Liang Liu. Correlated differential privacy protection for mobile crowdsensing. *IEEE Trans. Big Data*, 7(4):784–795, 2021.

[24] Keke Chen and Ling Liu. Privacy preserving data classification with rotation perturbation. In *Proceedings of the 5th IEEE International Conference on Data Mining ICDM 2005, 27-30 November 2005, Houston, Texas, USA*, pages 589–592. IEEE Computer Society, 2005.

[25] Rui Chen, Gergely Ács, and Claude Castelluccia. Differentially private sequential data publication via variable-length n-grams. In Ting Yu, George Danezis, and Virgil D. Gligor, editors, *the ACM Conference on Computer and Communications Security, CCS'12, Raleigh, NC, USA, October 16-18, 2012*, pages 638–649. ACM, 2012.

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

[27] Eunjoon Cho, Seth A. Myers, and Jure Leskovec. Friendship and mobility: user movement in location-based social networks. In Chid Apté, Joydeep Ghosh, and Padhraic Smyth, editors, *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, CA, USA, August 21-24, 2011*, pages 1082–1090. ACM, 2011.

[28] Amrita Roy Chowdhury, Chenghong Wang, Xi He, Ashwin Machanavajjhala, and Somesh Jha. Crypt?: Crypto-assisted differential privacy on untrusted servers. In David Maier,

Rachel Pottinger, AnHai Doan, Wang-Chiew Tan, Abdussalam Alawini, and Hung Q. Ngo, editors, *Proceedings of the 2020 International Conference on Management of Data, SIGMOD Conference 2020, online conference [Portland, OR, USA], June 14-19, 2020*, pages 603–619. ACM, 2020.

[29] Graham Cormode, Tejas Kulkarni, and Divesh Srivastava. Answering range queries under local differential privacy. *Proc. VLDB Endow.*, 12(10):1126–1138, 2019.

[30] Graham Cormode, Samuel Maddock, and Carsten Maple. Frequency estimation under local differential privacy. *Proc. VLDB Endow.*, 14(11):2046–2058, 2021.

[31] William L. Croft, Jörg-Rüdiger Sack, and Wei Shi. Differential privacy via a truncated and normalized laplace mechanism. *J. Comput. Sci. Technol.*, 37(2):369–388, 2022.

[32] Paul Cuff and Lanqing Yu. Differential privacy as a mutual information constraint. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*, pages 43–54. ACM, 2016.

[33] Teddy Cunningham, Graham Cormode, and Hakan Ferhatosmanoglu. Privacy-preserving synthetic location data in the real world. In Erik Hoel, Dev Oliver, Raymond Chi-Wing Wong, and Ahmed Eldawy, editors, *Proceedings of the 17th International Symposium on Spatial and Temporal Databases, SSTD 2021, Virtual Event, USA, August 23-25, 2021*, pages 23–33. ACM, 2021.

[34] Teddy Cunningham, Graham Cormode, Hakan Ferhatosmanoglu, and Divesh Srivastava. Real-world trajectory sharing with local differential privacy. *Proc. VLDB Endow.*, 14(11):2283–2295, 2021.

[35] Li Deng. The MNIST database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Process. Mag.*, 29(6):141–142, 2012.

[36] Samuel Fuller Dodge and Lina J. Karam. Understanding how image quality affects deep neural networks. In *Eighth International Conference on Quality of Multimedia Experience, QoMEX 2016, Lisbon, Portugal, June 6-8, 2016*, pages 1–6. IEEE, 2016.

[37] Jinshuo Dong, David Durfee, and Ryan Rogers. Optimal differential privacy composition for exponential mechanisms. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 2597–2606. PMLR, 2020.

[38] Jinshuo Dong, Aaron Roth, and Weijie J. Su. Gaussian differential privacy. *CoRR*, abs/1905.02383, 2019.

[39] Minxin Du, Xiang Yue, Sherman S. M. Chow, and Huan Sun. Sanitizing sentence embeddings (and labels) for local differential privacy. In Ying Ding, Jie Tang, Juan F. Sequeda, Lora Aroyo, Carlos Castillo, and Geert-Jan Houben, editors, *Proceedings of the ACM Web Conference 2023, WWW 2023, Austin, TX, USA, 30 April 2023 - 4 May 2023*, pages 2349–2359. ACM, 2023.

[40] Yuntao Du, Yujia Hu, Zhikun Zhang, Ziquan Fang, Lu Chen, Baihua Zheng, and Yunjun Gao. Ldptrace: Locally differentially private trajectory synthesis. *Proc. VLDB Endow.*, 16(8):1897–1909, 2023.

[41] D.P. Dubhashi and A. Panconesi. *Concentration of Measure for the Analysis of Randomized Algorithms*. Cambridge University Press, 2009.

[42] John C. Duchi, Michael I. Jordan, and Martin J. Wainwright. Local privacy and statistical minimax rates. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, Berkeley, CA, USA, October, 26-29, 2013*, pages 429–438. IEEE Computer Society, 2013.

[43] John C. Duchi, Michael I. Jordan, and Martin J. Wainwright. Local privacy, data processing inequalities, and statistical minimax rates. *arXiv: Statistics Theory*, 2013.

[44] John C. Duchi, Martin J. Wainwright, and Michael I. Jordan. Minimax optimal procedures for locally private estimation. *CoRR*, abs/1604.02390, 2016.

[45] Cynthia Dwork. Differential privacy. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *Automata, Languages and Programming, 33rd International Colloquium, ICALP 2006, Venice, Italy, July 10-14, 2006, Proceedings, Part II*, volume 4052 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2006.

[46] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard S. Zemel. Fairness through awareness. In Shafi Goldwasser, editor, *Innovations in Theoretical Computer Science 2012, Cambridge, MA, USA, January 8-10, 2012*, pages 214–226. ACM, 2012.

[47] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam D. Smith. Calibrating noise to sensitivity in private data analysis. In Shai Halevi and Tal Rabin, editors, *Theory of Cryptography, Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006, Proceedings*, volume 3876 of *Lecture Notes in Computer Science*, pages 265–284. Springer, 2006.

[48] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3-4):211–407, 2014.

[49] Úlfar Erlingsson, Vitaly Feldman, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Abhradeep Thakurta. Amplification by shuffling: From local to central differential privacy via anonymity. In Timothy M. Chan, editor, *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 2468–2479. SIAM, 2019.

[50] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. RAPPOR: randomized aggregatable privacy-preserving ordinal response. In Gail-Joon Ahn, Moti Yung, and Ninghui Li, editors, *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, AZ, USA, November 3-7, 2014*, pages 1054–1067. ACM, 2014.

[51] Huiyu Fang, Liquan Chen, Yali Liu, and Yuan Gao. Locally differentially private frequency estimation based on convolution framework. In *44th IEEE Symposium on Security and Privacy, SP 2023, San Francisco, CA, USA, May 21-25, 2023*, pages 2208–2222. IEEE, 2023.

[52] Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. Robustness of classifiers: from adversarial to random noise. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 1624–1632, 2016.

[53] Fedesoriano. Stroke Prediction Dataset.

[54] Natasha Fernandes, Annabelle McIver, and Carroll Morgan. The laplace mechanism has optimal utility for differential privacy over continuous queries. In *36th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2021, Rome, Italy, June 29 - July 2, 2021*, pages 1–12. IEEE, 2021.

[55] Nicholas I. Fisher. *Statistical analysis of circular data*. Cambridge University Press, Cambridge, Mass., transferred to digital printing edition, 2000.

[56] Matthew Fredrikson, Eric Lantz, Somesh Jha, Simon M. Lin, David Page, and Thomas Ristenpart. Privacy in pharmacogenetics: An end-to-end case study of personalized warfarin dosing. In Kevin Fu and Jaeyeon Jung, editors, *Proceedings of the 23rd USENIX Security Symposium, San Diego, CA, USA, August 20-22, 2014*, pages 17–32. USENIX Association, 2014.

[57] Jie Fu, Yuan Hong, Xinpeng Ling, Leixia Wang, Xun Ran, Zhiyu Sun, Wendy Hui Wang, Zhili Chen, and Yang Cao. Differentially private federated learning: A systematic review. *CoRR*, abs/2405.08299:36, 2024.

[58] Chang Ge, Xi He, Ihab F. Ilyas, and Ashwin Machanavajjhala. Apex: Accuracy-aware differentially private data exploration. In Peter A. Boncz, Stefan Manegold, Anastasia Ailamaki, Amol Deshpande, and Tim Kraska, editors, *Proceedings of the 2019 International Conference on Management of Data, SIGMOD Conference 2019, Amsterdam, The Netherlands, June 30 - July 5, 2019*, pages 177–194. ACM, 2019.

[59] Quan Geng, Wei Ding, Ruiqi Guo, and Sanjiv Kumar. Tight analysis of privacy and utility tradeoff in approximate differential privacy. In Silvia Chiappa and Roberto Calandra, editors, *The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020, 26-28 August 2020, Online [Palermo, Sicily, Italy]*, volume 108 of *Proceedings of Machine Learning Research*, pages 89–99. PMLR, 2020.

[60] Quan Geng and Pramod Viswanath. The optimal mechanism in differential privacy. In *2014 IEEE International Symposium on Information Theory, Honolulu, HI, USA, June 29 - July 4, 2014*, pages 2371–2375. IEEE, 2014.

[61] Badih Ghazi, Noah Golowich, Ravi Kumar, Rasmus Pagh, and Ameya Velingker. On the power of multiple anonymous messages. *IACR Cryptol. ePrint Arch.*, page 1382, 2019.

[62] Badih Ghazi, Ravi Kumar, Pasin Manurangsi, Rasmus Pagh, and Amer Sinha. Differentially private aggregation in the shuffle model: Almost central accuracy in almost a single message. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 3692–3701. PMLR, 2021.

[63] Benyamin Ghojogh and Mark Crowley. Linear and quadratic discriminant analysis: Tutorial. *CoRR*, abs/1906.02590, 2019.

[64] Arpita Ghosh, Tim Roughgarden, and Mukund Sundararajan. Universally utility-maximizing privacy mechanisms. In Michael Mitzenmacher, editor, *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*, pages 351–360. ACM, 2009.

[65] Google. google/differential-privacy. original-date: 2019-09-04T13:04:15Z.

[66] Xiaolan Gu, Ming Li, Yueqiang Cheng, Li Xiong, and Yang Cao. PCKV: locally differentially private correlated key-value data collection with optimized utility. In Srdjan Capkun and Franziska Roesner, editors, *29th USENIX Security Symposium, USENIX Security 2020, August 12-14, 2020*, pages 967–984. USENIX Association, 2020.

[67] Xiaolan Gu, Ming Li, Li Xiong, and Yang Cao. Providing input-discriminative protection for local differential privacy. In *36th IEEE International Conference on Data Engineering, ICDE 2020, Dallas, TX, USA, April 20-24, 2020*, pages 505–516. IEEE, 2020.

[68] Akshay Gupte, Shabbir Ahmed, Santanu S. Dey, and Myun-Seok Cheon. Relaxations and discretizations for the pooling problem. *J. Glob. Optim.*, 67(3):631–669, 2017.

[69] Moritz Hardt and Kunal Talwar. On the geometry of differential privacy. In Leonard J. Schulman, editor, *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*, pages 705–714. ACM, 2010.

[70] Mark Hasegawa-Johnson. ECE 417 Multimedia Signal Processing.

[71] Trevor Hastie, Robert Tibshirani, and Jerome H. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd Edition*. Springer Series in Statistics. Springer, 2009.

[72] Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963.

[73] Naoise Holohan, Spiros Antonatos, Stefano Braghin, and Pól Mac Aonghusa. The bounded laplace mechanism in differential privacy. *J. Priv. Confidentiality*, 10(1), 2020.

[74] I-Jung Hsu, Chih-Hsun Lin, Chia-Mu Yu, Sy-Yen Kuo, and Chun-Ying Huang. Poisoning attacks to local differential privacy protocols for trajectory data. *CoRR*, abs/2503.07483, 2025.

[75] Anil K. Jain, M. Narasimha Murty, and Patrick J. Flynn. Data clustering: A review. *ACM Comput. Surv.*, 31(3):264–323, 1999.

[76] Hongbo Jiang, Jie Li, Ping Zhao, Fanzi Zeng, Zhu Xiao, and Arun Iyengar. Location privacy-preserving mechanisms in location-based services: A comprehensive survey. *ACM Comput. Surv.*, 54(1):4:1–4:36, 2022.

[77] Kaifeng Jiang, Dongxu Shao, Stéphane Bressan, Thomas Kister, and Kian-Lee Tan. Publishing trajectories with differential privacy guarantees. In Alex Szalay, Tamas Budavari, Magdalena Balazinska, Alexandra Meliou, and Ahmet Sacan, editors, *Conference on Scientific and Statistical Database Management, SSDBM '13, Baltimore, MD, USA, July 29 - 31, 2013*, pages 12:1–12:12. ACM, 2013.

[78] Peter Kairouz, Sewoong Oh, and Pramod Viswanath. Extremal mechanisms for local differential privacy. In Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27: Annual*

*Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 2879–2887, 2014.

[79] Peter Kairouz, Sewoong Oh, and Pramod Viswanath. The composition theorem for differential privacy. *IEEE Trans. Inf. Theory*, 63(6):4037–4049, 2017.

[80] Can Kanbak, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. Geometric robustness of deep networks: Analysis and improvement. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 4441–4449. Computer Vision Foundation / IEEE Computer Society, 2018.

[81] Shiva Prasad Kasiviswanathan, Homin K. Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam D. Smith. What can we learn privately? *SIAM J. Comput.*, 40(3):793–826, 2011.

[82] Guy Katz, Clark W. Barrett, David L. Dill, Kyle Julian, and Mykel J. Kochenderfer. Reluplex: An efficient SMT solver for verifying deep neural networks. In Rupak Majumdar and Viktor Kuncak, editors, *Computer Aided Verification - 29th International Conference, CAV 2017, Heidelberg, Germany, July 24-28, 2017, Proceedings, Part I*, volume 10426 of *Lecture Notes in Computer Science*, pages 97–117. Springer, 2017.

[83] Daniel Kifer and Ashwin Machanavajjhala. No free lunch in data privacy. In Timos K. Sellis, Renée J. Miller, Anastasios Kementsietsidis, and Yannis Velegrakis, editors, *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2011, Athens, Greece, June 12-16, 2011*, pages 193–204. ACM, 2011.

[84] Daniel Kifer and Ashwin Machanavajjhala. Pufferfish: A framework for mathematical privacy definitions. *ACM Trans. Database Syst.*, 39(1):3:1–3:36, 2014.

[85] Muah Kim, Onur Günlü, and Rafael F. Schaefer. Federated learning with local differential privacy: Trade-offs between privacy, utility, and communication. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2021, Toronto, ON, Canada, June 6-11, 2021*, pages 2650–2654. IEEE, 2021.

[86] Sosuke Kobayashi. Contextual augmentation: Data augmentation by words with paradigmatic relations. In Marilyn A. Walker, Heng Ji, and Amanda Stent, editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers)*, pages 452–457. Association for Computational Linguistics, 2018.

[87] Kamran Kowsari, Kiana Jafari Meimandi, Mojtaba Heidarysafa, Sanjana Mendu, Laura E. Barnes, and Donald E. Brown. Text classification algorithms: A survey. *Inf.*, 10(4):150, 2019.

[88] Jinyu Li, Li Deng, Yifan Gong, and Reinhold Haeb-Umbach. An overview of noise-robust automatic speech recognition. *IEEE ACM Trans. Audio Speech Lang. Process.*, 22(4):745–777, 2014.

[89] Ninghui Li, Tiancheng Li, and Suresh Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In Rada Chirkova, Asuman Dogac, M. Tamer Özsu, and Timos K. Sellis, editors, *Proceedings of the 23rd International Conference on Data Engineering, ICDE 2007, The Marmara Hotel, Istanbul, Turkey, April 15-20, 2007*, pages 106–115. IEEE Computer Society, 2007.

[90] Ninghui Li, Wahbeh H. Qardaji, Dong Su, and Jianneng Cao. Privbasis: Frequent itemset mining with differential privacy. *Proc. VLDB Endow.*, 5(11):1340–1351, 2012.

[91] X. Li, C. Zhang, T. Jung, J. Qian, and L. Chen. Graph-based privacy-preserving data publication. In *Proceedings of the 35th Annual IEEE International Conference on Computer Communications (INFOCOM 2016)*, pages 1–9, April 2016.

[92] Xiaoguang Li, Ninghui Li, Wenhai Sun, Neil Zhenqiang Gong, and Hui Li. Fine-grained poisoning attack to local differential privacy protocols for mean and variance estimation. In Joseph A. Calandrino and Carmela Troncoso, editors, *32nd USENIX Security Symposium, USENIX Security 2023, Anaheim, CA, USA, August 9-11, 2023*, pages 1739–1756. USENIX Association, 2023.

[93] Xiaoguang Li, Zitao Li, Ninghui Li, and Wenhai Sun. On the robustness of LDP protocols for numerical attributes under data poisoning attacks. In *32nd Annual Network and Distributed System Security Symposium, NDSS 2025, San Diego, California, USA, February 24-28, 2025*. The Internet Society, 2025.

[94] Yan-zi Li, Li Xu, Jing Zhang, and Liao-ru-xing Zhang. WF-LDPSR: A local differential privacy mechanism based on water-filling for secure release of trajectory statistics data. *Comput. Secur.*, 148:104165, 2025.

[95] Zitao Li, Tianhao Wang, Milan Lopuhaä-Zwakenberg, Ninghui Li, and Boris Skoric. Estimating numerical distributions under local differential privacy. In David Maier, Rachel Pottinger, AnHai Doan, Wang-Chiew Tan, Abdussalam Alawini, and Hung Q. Ngo, editors, *Proceedings of the 2020 International Conference on Management of Data, SIGMOD Conference 2020, online conference [Portland, OR, USA], June 14-19, 2020*, pages 621–635. ACM, 2020.

[96] Chun Liu, Youliang Tian, Jinchuan Tang, Shuping Dang, and Gaojie Chen. A novel local differential privacy federated learning under multi-privacy regimes. *Expert Syst. Appl.*, 227:120266, 2023.

[97] Fang Liu. Statistical properties of sanitized results from differentially private laplace mechanism with univariate bounding constraints. *Trans. Data Priv.*, 12(3):169–195, 2019.

[98] Xinyi Liu, Ye Zheng, Zhengxiong Li, and Yidan Hu. Multi-sensor data privacy protection with adaptive privacy budget for iot systems. In *2024 IEEE Conference on Communications and Network Security (CNS)*, pages 1–9, 2024.

[99] D. Lu and Q. Weng. A survey of image classification methods and techniques for improving classification performance. *International Journal of Remote Sensing*, 28(5):823–870, 2007.

[100] Qiyao Luo, Yilei Wang, and Ke Yi. Frequency estimation in the shuffle model with almost a single message. In Heng Yin, Angelos Stavrou, Cas Cremers, and Elaine Shi, editors, *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, CCS 2022, Los Angeles, CA, USA, November 7-11, 2022*, pages 2219–2232. ACM, 2022.

[101] Fei Ma, Renbo Zhu, and Ping Wang. PTT: piecewise transformation technique for analyzing numerical data under local differential privacy. *IEEE Trans. Mob. Comput.*, 23(10):9518–9531, 2024.

[102] Gabriel Resende Machado, Eugênio Silva, and Ronaldo Ribeiro Goldschmidt. Adversarial machine learning in image classification: A survey toward the defender's perspective. *ACM Comput. Surv.*, 55(2):8:1–8:38, 2023.

[103] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[104] Karima Makhlouf, Tamara Stefanovic, Héber Hwang Arcolezi, and Catuscia Palamidessi. A systematic and formal study of the impact of local differential privacy on fairness: Preliminary results. In *37th IEEE Computer Security Foundations Symposium, CSF 2024, Enschede, Netherlands, July 8-12, 2024*, pages 1–16. IEEE, 2024.

[105] Mohammad Malekzadeh, Richard G. Clegg, Andrea Cavallaro, and Hamed Haddadi. Mobile sensor data anonymization. In *Proceedings of the International Conference on Internet of Things Design and Implementation*, IoTDI '19, pages 49–58, New York, NY, USA, 2019. ACM.

[106] Kanti V Mardia and Peter E Jupp. *Directional statistics*. John Wiley & Sons, 2009.

[107] Sagar Maru. Bank Customer Attrition Insights.

[108] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In Aarti

Singh and Jerry Zhu, editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 1273–1282. PMLR, 20–22 Apr 2017.

[109] H. Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. In *International Conference on Learning Representations*, 2018.

[110] Frank McSherry and Kunal Talwar. Mechanism design via differential privacy. In *48th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2007, Providence, RI, USA, October 20-23, 2007, Proceedings*, pages 94–103. IEEE Computer Society, 2007.

[111] Ricardo Mendes, Mariana Cunha, and João P. Vilela. Impact of frequency of location reports on the privacy level of geo-indistinguishability. *Proc. Priv. Enhancing Technol.*, 2020(2):379–396, 2020.

[112] Mark Huasong Meng, Guangdong Bai, Sin Gee Teo, Zhe Hou, Yan Xiao, Yun Lin, and Jin Song Dong. Adversarial robustness of deep neural networks: A survey from a formal verification perspective. *CoRR*, abs/2206.12227, 2022.

[113] Àlex Miranda-Pascual, Patricia Guerra-Balboa, Javier Parra-Arnau, Jordi Forné, and Thorsten Strufe. Sok: Differentially private publication of trajectory data. *Proc. Priv. Enhancing Technol.*, 2023(2):496–516, 2023.

[114] Ilya Mironov. Rényi differential privacy. In *30th IEEE Computer Security Foundations Symposium, CSF 2017, Santa Barbara, CA, USA, August 21-25, 2017*, pages 263–275. IEEE Computer Society, 2017.

[115] Jeet Mohapatra, Tsui-Wei Weng, Pin-Yu Chen, Sijia Liu, and Luca Daniel. Towards verifying robustness of neural networks against A family of semantic perturbations. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 241–249. Computer Vision Foundation / IEEE, 2020.

[116] Mahnush Movahedi, Jared Saia, and Mahdi Zamani. Secure multi-party shuffling. *IACR Cryptol. ePrint Arch.*, page 664, 2015.

[117] Jack Murtagh and Salil P. Vadhan. The complexity of computing the optimal composition of differential privacy. *Theory Comput.*, 14(1):1–35, 2018.

[118] Arvind Narayanan and Vitaly Shmatikov. Robust de-anonymization of large sparse datasets. In *2008 IEEE Symposium on Security and Privacy (SP 2008), 18-21 May 2008, Oakland, California, USA*, pages 111–125. IEEE Computer Society, 2008.

[119] Joseph P Near, David Darais, Naomi Lefkovitz, and Gary S Howarth. Guidelines for evaluating differential privacy guarantees.

[120] Nekst-Online. What Is the Pooling Problem?, July 2016.

[121] Kobbi Nissim, Rann Smorodinsky, and Moshe Tennenholtz. Approximately optimal mechanism design via differential privacy. In Shafi Goldwasser, editor, *Innovations in Theoretical Computer Science 2012, Cambridge, MA, USA, January 8-10, 2012*, pages 203–213. ACM, 2012.

[122] Chaoyue Niu, Zhenzhe Zheng, Fan Wu, Shaojie Tang, Xiaofeng Gao, and Guihai Chen. Unlocking the value of privacy: Trading aggregate statistics over private correlated data. In Yike Guo and Faisal Farooq, editors, *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, pages 2031–2040. ACM, 2018.

[123] L. Ou, Z. Qin, S. Liao, Y. Hong, and X. Jia. Releasing correlated trajectories: Towards high utility and optimal differential privacy. *IEEE Transactions on Dependable and Secure Computing*, pages 1–13, 2018.

[124] Luis Perez and Jason Wang. The effectiveness of data augmentation in image classification using deep learning. *CoRR*, abs/1712.04621, 2017.

[125] Natalia Ponomareva, Hussein Hazimeh, Alex Kurakin, Zheng Xu, Carson Denison, H. Brendan McMahan, Sergei Vassilvitskii, Steve Chien, and Abhradeep Guha Thakurta. How to dp-fy ML: A practical guide to machine learning with differential privacy. *J. Artif. Intell. Res.*, 77:1113–1201, 2023.

[126] Nica Potato. Women's e-commerce clothing reviews. Downloaded from `https://www.kaggle.com/nicapotato/womens-ecommerce-clothing-reviews`, 2018.

[127] Gauri Pradhan, Joonas Jälkö, Santiago Zanella-Bèguelin, and Antti Honkela. Beyond membership: Limitations of add/remove adjacency in differential privacy.

[128] 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*, CCS '16, page 192–203, 2016.

[129] Chenxi Qiu, Ruiyao Liu, Primal Pappachan, Anna Cinzia Squicciarini, and Xinpeng Xie. Time-efficient locally relevant geo-location privacy protection. *Proc. Priv. Enhancing Technol.*, 2025(2):5–22, 2025.

[130] Xuebin Ren, Liang Shi, Weiren Yu, Shusen Yang, Cong Zhao, and Zongben Xu. LDP-IDS: local differential privacy for infinite data streams. In Zachary G. Ives, Angela Bonifati, and

Amr El Abbadi, editors, *SIGMOD '22: International Conference on Management of Data, Philadelphia, PA, USA, June 12 - 17, 2022*, pages 1064–1077. ACM, 2022.

[131] Xuebin Ren, Chia-Mu Yu, Weiren Yu, Shusen Yang, Xinyu Yang, Julie McCann, and Philip Yu. Lopub: High-dimensional crowdsourced data publication with local differential privacy. *IEEE Transactions on Information Forensics and Security*, 13(9):2151–2166, Sep. 2018.

[132] Apple Machine Learning Research. Learning with privacy at scale, 2017.

[133] R L Rivest, L Adleman, and M L Dertouzos. On data banks and privacy homomorphisms. *Foundations of Secure Computation, Academia Press*, pages 169–179, 1978.

[134] Reuven Y. Rubinstein. *Simulation and the Monte Carlo method*. Wiley series in probability and mathematical statistics. Wiley, 1981.

[135] Steven Ruggles, J Trent Alexander, Katie Genadek, Ronald Goeken, Matthew B Schroeder, Matthew Sobek, et al. Integrated public use microdata series: Version 5.0 [machine-readable database]. *Minneapolis: University of Minnesota*, 42, 2010.

[136] Suchismita Sahu. Decision boundary for classifiers: An introduction, February 2024.

[137] Sharmistha Chatterjee Sapient, Senior Manager Data Sciences at Publicis. A guide to differential privacy at scale, December 2020.

[138] Shafizur Rahman Seeam, Ye Zheng, and Yidan Hu. Frequency estimation of correlated multi-attribute data under local differential privacy. *CoRR*, abs/2507.17516, 2025.

[139] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017*, pages 3–18. IEEE Computer Society, 2017.

[140] Shuang Song, Yizhen Wang, and Kamalika Chaudhuri. Pufferfish privacy mechanisms for correlated data. In Semih Salihoglu, Wenchao Zhou, Rada Chirkova, Jun Yang, and Dan Suciu, editors, *Proceedings of the 2017 ACM International Conference on Management of Data, SIGMOD Conference 2017, Chicago, IL, USA, May 14-19, 2017*, pages 1291–1306. ACM, 2017.

[141] Jordi Soria-Comas and Josep Domingo-Ferrer. Optimal data-independent noise for differential privacy. *Inf. Sci.*, 250:200–214, 2013.

[142] 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 *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security (CCS'19)*, page 703–717, 2019.

[143] Latanya Sweeney. k-anonymity: A model for protecting privacy. *Int. J. Uncertain. Fuzziness Knowl. Based Syst.*, 10(5):557–570, 2002.

[144] R. Tempo, E.W. Bai, and F. Dabbene. Probabilistic robustness analysis: explicit bounds for the minimum number of samples. In *Proceedings of 35th IEEE Conference on Decision and Control*, volume 3, pages 3424–3428 vol.3, 1996.

[145] Montana State University. Lat/Lon and UTM Conversion - Yellowstone Research Coordination Network, n.d.

[146] Elisabet Lobo Vesga, Alejandro Russo, and Marco Gaboardi. A programming framework for differential privacy with accuracy concentration bounds. In *2020 IEEE Symposium on Security and Privacy, SP 2020, San Francisco, CA, USA, May 18-21, 2020*, pages 411–428. IEEE, 2020.

[147] Sajani Vithana, Viveck R. Cadambe, Flávio P. Calmon, and Haewon Jeong. Correlated privacy mechanisms for differentially private distributed mean estimation. In *IEEE Conference on Secure and Trustworthy Machine Learning, SaTML 2025, Copenhagen, Denmark, April 9-11, 2025*, pages 590–614. IEEE, 2025.

[148] Han Wang, Hanbin Hong, Li Xiong, Zhan Qin, and Yuan Hong. L-SRR: local differential privacy for location-based services with staircase randomized response. In Heng Yin, Angelos Stavrou, Cas Cremers, and Elaine Shi, editors, *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, CCS 2022, Los Angeles, CA, USA, November 7-11, 2022*, pages 2809–2823. ACM, 2022.

[149] Nana Wang and Mohan S. Kankanhalli. Protecting sensitive place visits in privacy-preserving trajectory publishing. *Comput. Secur.*, 97:101949, 2020.

[150] 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 *35th IEEE International Conference on Data Engineering, ICDE 2019, Macao, China, April 8-11, 2019*, pages 638–649. IEEE, 2019.

[151] Shaowei Wang, Liusheng Huang, Yiwen Nie, Pengzhan Wang, Hongli Xu, and Wei Yang. Privset: Set-valued data analyses with locale differential privacy. In *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, pages 1088–1096, 2018.

[152] Shaowei Wang, Yuqiu Qian, Jiachun Du, Wei Yang, Liusheng Huang, and Hongli Xu. Set-valued data publication with local privacy: Tight error bounds and efficient mechanisms. *Proc. VLDB Endow.*, 13(8):1234–1247, April 2020.

[153] Tianhao Wang, Jeremiah Blocki, Ninghui Li, and Somesh Jha. Locally differentially private protocols for frequency estimation. In Engin Kirda and Thomas Ristenpart, editors, *26th USENIX Security Symposium, USENIX Security 2017, Vancouver, BC, Canada, August 16-18, 2017*, pages 729–745. USENIX Association, 2017.

[154] Tianhao Wang, Ninghui Li, and Somesh Jha. Locally differentially private frequent itemset mining. In *2018 IEEE Symposium on Security and Privacy, SP 2018, Proceedings, 21-23 May 2018, San Francisco, California, USA*, pages 127–143. IEEE Computer Society, 2018.

[155] Tianhao Wang, Ninghui Li, and Somesh Jha. Locally differentially private heavy hitter identification. *IEEE Trans. Dependable Secur. Comput.*, 18(2):982–993, 2021.

[156] Tianhao Wang, Milan Lopuhaä-Zwakenberg, Zitao Li, Boris Skoric, and Ninghui Li. Locally differentially private frequency estimation with consistency. In *27th Annual Network and Distributed System Security Symposium, NDSS 2020, San Diego, California, USA, February 23-26, 2020*. The Internet Society, 2020.

[157] Yong Wang, Mingxing Gao, Xun Ran, Jun Ma, and Leo Yu Zhang. An improved matrix factorization with local differential privacy based on piecewise mechanism for recommendation systems. *Expert Syst. Appl.*, 216:119457, 2023.

[158] Zhibo Wang, Wenxin Liu, Xiaoyi Pang, Ju Ren, Zhe Liu, and Yongle Chen. Towards pattern-aware privacy-preserving real-time data collection. In *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, pages 109–118, 2020.

[159] Stanley L. Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60(309):63–69, 1965.

[160] Benjamin Weggenmann and Florian Kerschbaum. Differential privacy for directional data. In Yongdae Kim, Jong Kim, Giovanni Vigna, and Elaine Shi, editors, *CCS '21: 2021 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, Republic of Korea, November 15 - 19, 2021*, pages 1205–1222. ACM, 2021.

[161] Jason W. Wei and Kai Zou. EDA: easy data augmentation techniques for boosting performance on text classification tasks. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 6381–6387. Association for Computational Linguistics, 2019.

[162] Kilian Weinberger. Machine learning for intelligent systems.

[163] Tsui-Wei Weng, Huan Zhang, Pin-Yu Chen, Jinfeng Yi, Dong Su, Yupeng Gao, Cho-Jui Hsieh, and Luca Daniel. Evaluating the robustness of neural networks: An extreme value theory approach. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[164] Wikipedia. $k$-means clustering.

[165] Wikipedia. Bauer maximum principle, February 2024.

[166] Wikipedia contributors. Mutual information. Page Version ID: 1328849415.

[167] Hanshen Xiao and Srinivas Devadas. PAC privacy: Automatic privacy measurement and control of data processing. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology - CRYPTO 2023 - 43rd Annual International Cryptology Conference, CRYPTO 2023, Santa Barbara, CA, USA, August 20-24, 2023, Proceedings, Part II*, volume 14082 of *Lecture Notes in Computer Science*, pages 611–644. Springer, 2023.

[168] Hanshen Xiao, Jun Wan, and Srinivas Devadas. Geometry of sensitivity: Twice sampling and hybrid clipping in differential privacy with optimal gaussian noise and application to deep learning. In Weizhi Meng, Christian Damsgaard Jensen, Cas Cremers, and Engin Kirda, editors, *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security, CCS 2023, Copenhagen, Denmark, November 26-30, 2023*, pages 2636–2650. ACM, 2023.

[169] Min Xu, Bolin Ding, Tianhao Wang, and Jingren Zhou. Collecting and analyzing data jointly from multiple services under local differential privacy. *Proc. VLDB Endow.*, 13(12):2760–2772, July 2020.

[170] Bin Yang, Issei Sato, and Hiroshi Nakagawa. Bayesian differential privacy on correlated data. In Timos K. Sellis, Susan B. Davidson, and Zachary G. Ives, editors, *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, Melbourne, Victoria, Australia, May 31 - June 4, 2015*, pages 747–762. ACM, 2015.

[171] Dingqi Yang, Daqing Zhang, Vincent W. Zheng, and Zhiyong Yu. Modeling user activity preference by leveraging user spatial temporal characteristics in lbsns. *IEEE Trans. Syst. Man Cybern. Syst.*, 45(1):129–142, 2015.

[172] Haolong Yang, Dingyuan Shi, Yuanyuan Zhang, Yi Xu, and Ke Xu. An efficient local differential privacy approach for trajectory publishing with high utility. In Makoto Onizuka, Jae-Gil Lee, Yongxin Tong, Chuan Xiao, Yoshiharu Ishikawa, Sihem Amer-Yahia, H. V. Jagadish, and Kejing Lu, editors, *Database Systems for Advanced Applications*, pages 71–88, Singapore, 2024. Springer Nature Singapore.

[173] Andrew Chi-Chih Yao. Protocols for secure computations (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November 1982*, pages 160–164. IEEE Computer Society, 1982.

[174] Qingqing Ye, Haibo Hu, Xiaofeng Meng, and Huadi Zheng. Privkv: Key-value data collection with local differential privacy. In *2019 IEEE Symposium on Security and Privacy, SP 2019, San Francisco, CA, USA, May 19-23, 2019*, pages 317–331. IEEE, 2019.

[175] Dongyue Zhang, Weiwei Ni, Nan Fu, Lihe Hou, and Ruyu Zhang. Locally differentially private multi-dimensional data collection via haar transform. *Comput. Secur.*, 130:103291, 2023.

[176] Dongyue Zhang, Weiwei Ni, Nan Fu, Lihe Hou, and Ruyu Zhang. Locally differentially private trajectory publication based on regional popularity awareness. *IEEE Trans. Inf. Forensics Secur.*, 20:6719–6732, 2025.

[177] Jun Zhang, Xiaokui Xiao, and Xing Xie. Privtree: A differentially private algorithm for hierarchical decompositions. In Fatma Özcan, Georgia Koutrika, and Sam Madden, editors, *Proceedings of the 2016 International Conference on Management of Data, SIGMOD Conference 2016, San Francisco, CA, USA, June 26 - July 01, 2016*, pages 155–170. ACM, 2016.

[178] Kai Zhang, Yanjun Zhang, Ruoxi Sun, Pei-Wei Tsai, Muneeb Ul Hassan, Xin Yuan, Minhui Xue, and Jinjun Chen. Bounded and unbiased composite differential privacy. In *IEEE Symposium on Security and Privacy, SP 2024, San Francisco, CA, USA, May 19-23, 2024*, pages 972–990. IEEE, 2024.

[179] Nan Zhang, Shengquan Wang, and Wei Zhao. A new scheme on privacy-preserving data classification. In Robert Grossman, Roberto J. Bayardo, and Kristin P. Bennett, editors, *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Chicago, Illinois, USA, August 21-24, 2005*, pages 374–383. ACM, 2005.

[180] Tianle Zhang, Wenjie Ruan, and Jonathan E. Fieldsend. Proa: A probabilistic robustness assessment against functional perturbations. In Massih-Reza Amini, Stéphane Canu, Asja Fischer, Tias Guns, Petra Kralj Novak, and Grigorios Tsoumakas, editors, *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2022, Grenoble, France, September 19-23, 2022, Proceedings, Part III*, volume 13715 of *Lecture Notes in Computer Science*, pages 154–170. Springer, 2022.

[181] Yuemin Zhang, Qingqing Ye, Rui Chen, Haibo Hu, and Qilong Han. Trajectory data collection with local differential privacy. *Proc. VLDB Endow.*, 16(10):2591–2604, 2023.

[182] Xiaodong Zhao, Dechang Pi, and Junfu Chen. Novel trajectory privacy-preserving method based on clustering using differential privacy. *Expert Syst. Appl.*, 149:113241, 2020.

[183] Yang Zhao, Jun Zhao, Mengmeng Yang, Teng Wang, Ning Wang, Lingjuan Lyu, Dusit Niyato, and Kwok-Yan Lam. Local differential privacy-based federated learning for internet of things. *IEEE Internet Things J.*, 8(11):8836–8853, 2021.

[184] Ye Zheng, Sumita Mishra, and Yidan Hu. Optimal piecewise-based mechanism for collecting bounded numerical data under local differential privacy. *Proc. Priv. Enhancing Technol.*, 2025(4):146–165, 2025.

[185] Kaiyang Zhou, Ziwei Liu, Yu Qiao, Tao Xiang, and Chen Change Loy. Domain generalization: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.*, 45(4):4396–4415, 2023.

[186] Ezgi Zorarpaci and Selma Ayse Özel. Privacy preserving classification over differentially private data. *WIREs Data Mining Knowl. Discov.*, 11(3), 2021.

# Appendices

# Appendix A

# Appendix of Background (Chapter 2)

## A.1 Proofs

### A.1.1 Proof of Proposition 1

We prove the discrete output case; the continuous output case is similar.

*Proof.* For any measurable output event $\mathcal{Y}_{\text{sub}} \subseteq \mathcal{Y}$,

$$\begin{aligned}\Pr[\mathcal{M}(x_1) \in \mathcal{Y}_{\text{sub}}] &= \sum_{y \in \mathcal{Y}_{\text{sub}}} \Pr[\mathcal{M}(x_1) = y] \\ &\leq \sum_{y \in \mathcal{Y}_{\text{sub}}} e^{\varepsilon} \cdot \Pr[\mathcal{M}(x_2) = y] = e^{\varepsilon} \cdot \sum_{y \in \mathcal{Y}_{\text{sub}}} \Pr[\mathcal{M}(x_2) = y] \\ &= e^{\varepsilon} \cdot \Pr[\mathcal{M}(x_2) \in \mathcal{Y}_{\text{sub}}].\end{aligned}$$

Thus, the per-output definition implies Definition 5.

Conversely, let $\mathcal{Y}_{\text{sub}} = \{y\}$ be a singleton set containing only output $y$. Then, Definition 5 implies

$$\Pr[\mathcal{M}(x_1) = y] = \Pr[\mathcal{M}(x_1) \in \mathcal{Y}_{\text{sub}}] \leq e^{\varepsilon} \cdot \Pr[\mathcal{M}(x_2) \in \mathcal{Y}_{\text{sub}}] = e^{\varepsilon} \cdot \Pr[\mathcal{M}(x_2) = y].$$

Thus, Definition 5 implies the per-output definition. $\square$

### A.1.2 Proof of Theorem 1

*Proof.* Fix any $x_1, x_2 \in \mathcal{X}$ and $y' \in \mathcal{Y}'$. By the law of total probability,

$$\Pr[f \circ \mathcal{M}(x_1) = y'] = \sum_{y \in \mathcal{Y}} \Pr[\mathcal{M}(x_1) = y] \Pr[f(y) = y'].$$

Now apply the pointwise LDP inequality term-by-term inside the sum:

$$\forall y, \quad \Pr[\mathcal{M}(x_1) = y] \le e^{\varepsilon} \Pr[\mathcal{M}(x_2) = y].$$

Multiply both sides by the nonnegative number $\Pr[f(y) = y'] \ge 0$, then sum over $y$:

$$\sum_{y} \Pr[\mathcal{M}(x_1) = y] \, \Pr[f(y) = y'] \le e^{\varepsilon} \sum_{y} \Pr[\mathcal{M}(x_2) = y] \, \Pr[f(y) = y'].$$

This is exactly

$$\Pr[f \circ \mathcal{M}(x_1) = y'] \le e^{\varepsilon} \Pr[f \circ \mathcal{M}(x_2) = y'],$$

which proves that $f \circ \mathcal{M}$ is $\varepsilon$-LDP. □

### A.1.3 Proof of Theorem 2

*Proof.* Without loss of generality, we prove the case of two mechanisms, i.e. $k = 2$. Consider any $x_1, x_2 \in \mathcal{X}$ and $y_1, y_2$. By the law of total probability,

$$\Pr[\mathcal{M}(x_1) = (y_1, y_2)] = \Pr[\mathcal{M}_1(x_1) = y_1] \cdot \Pr[\mathcal{M}_2(x_1) = y_2 \mid \mathcal{M}_1(x_1) = y_1].$$

Similarly, for input $x_2$,

$$\Pr[\mathcal{M}(x_2) = (y_1, y_2)] = \Pr[\mathcal{M}_1(x_2) = y_1] \cdot \Pr[\mathcal{M}_2(x_2) = y_2 \mid \mathcal{M}_1(x_2) = y_1].$$

Now take the ratio:

$$\frac{\Pr[\mathcal{M}(x_1) = (y_1, y_2)]}{\Pr[\mathcal{M}(x_2) = (y_1, y_2)]} = \frac{\Pr[\mathcal{M}_1(x_1) = y_1]}{\Pr[\mathcal{M}_1(x_2) = y_1]} \cdot \frac{\Pr[\mathcal{M}_2(x_1) = y_2 \mid \mathcal{M}_1(x_1) = y_1]}{\Pr[\mathcal{M}_2(x_2) = y_2 \mid \mathcal{M}_1(x_2) = y_1]}.$$

By the $\varepsilon_1$-LDP property of $\mathcal{M}_1$, the first ratio is at most $e^{\varepsilon_1}$. For the second ratio: given this fixed $y_1$, the mechanism $\mathcal{M}_2(\cdot, y_1)$ is $\varepsilon_2$-LDP, so the second ratio is at most $e^{\varepsilon_2}$. Therefore,

$$\frac{\Pr[\mathcal{M}(x_1) = (y_1, y_2)]}{\Pr[\mathcal{M}(x_2) = (y_1, y_2)]} \le e^{\varepsilon_1} \cdot e^{\varepsilon_2} = e^{\varepsilon_1 + \varepsilon_2}.$$

This proves that the composed mechanism $\mathcal{M}$ is $(\varepsilon_1 + \varepsilon_2)$-LDP. □

## A.2 Complementary Materials

### A.2.1 Definition of LDP When $\varepsilon < 0$

Note that in Definition 2, the inequality condition must hold for all pairs $x_1, x_2$. For any fixed pair $x_1, x_2$, swapping $x_1$ and $x_2$ gives

$$\Pr[\mathcal{M}(x_2) = y] \leq e^{\varepsilon} \Pr[\mathcal{M}(x_1) = y].$$

Combining the original inequality and the swapped one yields

$$\Pr[\mathcal{M}(x_1) = y] \leq e^{\varepsilon} \Pr[\mathcal{M}(x_2) = y] \leq e^{2\varepsilon} \Pr[\mathcal{M}(x_1) = y].$$

If $\varepsilon < 0$, then $e^{2\varepsilon} < 1$, which implies

$$\forall x_1 \in \mathcal{X}, y \in \mathcal{Y}, \quad \Pr[\mathcal{M}(x_1) = y] = 0.$$

Hence, when $\varepsilon < 0$, the definition is not meaningful, as it would require the output probability to be zero for all inputs.

### A.2.2 Mutual Information Bound for LDP

This section establishes a mutual-information bound for $\varepsilon$-LDP mechanisms.

**Theorem 19** (Mutual-information bound under LDP)**.** *If a mechanism $\mathcal{M}$ is $\varepsilon$-LDP and $x$ is distributed according to any prior $P_x$ over $\mathcal{X}$, then*

$$I(x; \mathcal{M}(x)) \leq \min\{\log |\mathcal{X}|,\ \varepsilon\}.$$

*Proof.* To simplify notation, let $x \sim P_x$ be an arbitrary prior over the (finite) input domain $\mathcal{X}$, and let $y = \mathcal{M}(X)$ be the corresponding output random variable. For each $x \in \mathcal{X}$, write $Q_x(\cdot) \coloneqq \Pr[y = \cdot \mid x]$ for the conditional output distribution. The $\varepsilon$-LDP guarantee implies that for all $x, x' \in \mathcal{X}$ and all outputs $y$, we have $e^{-\varepsilon} \leq Q_x(y)/Q_{x'}(y) \leq e^{\varepsilon}$.

(i) The easy part is the domain-dependent bound. Mutual information is bounded by the entropy of the input [166], hence

$$I(x; y) \leq H(x) \leq \log |\mathcal{X}|.$$

(ii) The hard part is the $\varepsilon$-dependent bound. By the definition of mutual information [166],

$$I(x; y) = \mathrm{E}_x \left[ D_{\mathrm{KL}}(Q_x \parallel P_y) \right].$$

The key idea is to view $P_y$ as a mixture over the conditional output distributions $\{Q_{x'}\}_{x' \in \mathcal{X}}$, which allows us to relate the KL divergence to the $\varepsilon$-LDP property. In particular, since $P_y$ is the marginal induced by the prior $P_x$,

$$P_y(\cdot) = \sum_{x' \in \mathcal{X}} P_x(x')\, Q_{x'}(\cdot).$$

By the joint convexity of KL divergence in its second argument, we can upper bound the divergence to this mixture by an average of divergences to its components:

$$D_{\mathrm{KL}}(Q_x \;||\; P_y) \leq \sum_{x'} P_x(x') D_{\mathrm{KL}}(Q_x \;||\; Q_{x'}).$$

This introduces an auxiliary reference distribution $Q_{x'}$ (indexed by an independent draw $x'$) into the KL divergence. Averaging over $x$ then yields

$$I(x; y) = \mathrm{E}_x[D_{\mathrm{KL}}(Q_x \;||\; P_y)] \leq \mathrm{E}_{x,x'}[D_{\mathrm{KL}}(Q_x \;||\; Q_{x'})],$$

where $x, x' \sim P_x$ are i.i.d. samples. Now it suffices to upper bound $D_{\mathrm{KL}}(Q_x \;||\; Q_{x'})$.

By the definition of KL divergence and Jensen's inequality,

$$D_{\mathrm{KL}}(Q_x \;||\; Q_{x'}) = \sum_y Q_x(y) \log \frac{Q_x(y)}{Q_{x'}(y)} \leq \log \mathrm{E}_y \left[ \frac{Q_x(y)}{Q_{x'}(y)} \right] \leq \log e^{\varepsilon} = \varepsilon.$$

Plugging in this bound into the average over $x, x'$ gives

$$I(x; y) \leq \mathrm{E}_{x,x'}[D_{\mathrm{KL}}(Q_x \;||\; Q_{x'})] \leq \varepsilon.^*$$

(iii) Combining the two bounds (i) and (ii) gives the final result: $I(x; y) \leq \min\{\log |\mathcal{X}|, \varepsilon\}$. $\square$

*For small $\varepsilon$, this bound can be sharpened to $\mathcal{O}(\varepsilon^2)$ via a more refined analysis of inequalities for the KL divergence.

# Appendix B

# Appendix of Chapter 4

## B.1 Proofs

### B.1.1 Correlation Coefficient of JRR (Section 4.3.1)

*Proof.* The correlation coefficient between two random variables $T_{2i-1}$ and $T_{2i}$ is given by

$$\frac{\mathrm{Cov}[T_{2i-1}, T_{2i}]}{\sigma_{T_{2i-1}}\sigma_{T_{2i}}} = \frac{\mathrm{E}[T_{2i-1}T_{2i}] - \mathrm{E}[T_{2i-1}]\mathrm{E}[T_{2i}]}{\sigma_{T_{2i-1}}\sigma_{T_{2i}}},$$

where $\mathrm{Cov}(T_{2i-1}, T_{2i})$ is the covariance, $\sigma_1$ and $\sigma_2$ are the standard deviation of $T_{2i-1}$ and $T_{2i}$, respectively. According to the joint probability distribution in Table 4.2,

$$\begin{aligned}
\mathrm{E}[T_{2i-1}T_{2i}] &= \Pr[T_{2i-1}T_{2i} = 1] \cdot 1 + \Pr[T_{2i-1}T_{2i} = 0] \cdot 0 \\
&= \rho pq + p^2. \\
\mathrm{E}[T_{2i}] &= \Pr[T_{2i} = 1] \cdot 1 + \Pr[T_{2i} = 0] \cdot 0 \\
&= p.
\end{aligned}$$

$\mathrm{E}[T_{2i-1}]$ has the same result as $\mathrm{E}[T_{2i}]$. Meanwhile,

$$\begin{aligned}
\sigma^2_{T_{2i}} &= \mathrm{E}[T^2_{2i}] - \mathrm{E}^2[T_{2i}] \\
&= \Pr[T^2_{2i} = 1] \cdot 1 + \Pr[T^2_{2i} = 0] \cdot 0 - (\Pr[T_{2i} = 1] \cdot 1 + \Pr[T_{2i} = 0] \cdot 0)^2 \\
&= \Pr[T^2_{2i} = 1] - \Pr[T_{2i} = 1]^2 \\
&= p - p^2,
\end{aligned}$$

and $\sigma^2_{T_{2i-1}}$ has the same result. Hence $\sigma_{T_{2i-1}}\sigma_{T_{2i}} = p - p^2$.

Combining the above results, it follows that

$$\frac{\mathrm{Cov}[T_{2i-1}, T_{2i}]}{\sigma_{T_{2i-1}}\sigma_{T_{2i}}} = \frac{\rho pq + p^2 - p^2}{p - p^2} = \rho.$$

Note that $\rho$ must be greater than $1 - 1/p$ to ensure that the probability values in Table 4.2 are non-negative. $\square$

### B.1.2 Unbiasedness of the Estimator under JRR (Section 4.3.1)

*Proof.* The marginal distributions of $T_{2i-1}$ and $T_{2i}$ in Table 4.2 coincide. More specifically, for any user $u_j$,

$$T_j = \begin{cases} 1 & \text{with probability } p, \\ 0 & \text{with probability } q. \end{cases}$$

Let $y_j$ be the indicator variable that equals 1 if user $u_j$ reports a perturbed value of 1 (and 0 otherwise). Then

- if $x_j = 0$, $\Pr[y_j = 1 \mid x_j = 0] = \Pr[T_j = 0] = q$;
- if $x_j = 1$, $\Pr[y_j = 1 \mid x_j = 1] = \Pr[T_j = 1] = p$.

Denote $I_x$ as the number of users reporting value $x \in \{0, 1\}$. For $x = 1$ we have $I_1 = \sum_{j=1}^{n} y_j$. Taking expectations gives

$$\begin{aligned} \mathrm{E}[I_1] &= \sum_{j=1}^{n} \mathrm{E}[y_j] = \sum_{j=1}^{n} \Pr[y_j = 1] \\ &= n_1 \Pr[T_j = 1] + (n - n_1) \Pr[T_j = 0] \\ &= n_1 p + (n - n_1)q \\ &= (p - q)n_1 + nq. \end{aligned}$$

Hence the estimator

$$\hat{n}_1 = \frac{I_1 - nq}{p - q}$$

is unbiased for $n_1$. By the same argument, $\mathrm{E}[I_0] = (p - q)n_0 + nq$ and, since $I_0 = n - I_1$, the estimator for $n_0$ is unbiased as well. $\square$

Table B.1: Notations in the privacy and utility proofs.

| Notation | Meaning | Notation | Meaning |
|---|---|---|---|
| $\mathcal{T}_c$ | Truthfulness of colluding users | $I_x$ | Number of collected value $x$ |
| $\mathcal{C}$ | Indexes of colluding users | $n_1$ | Number of users having original value 1 |
| $m$ | Number of colluding users | $\text{Var}[\cdot]$ | Variance of a random variable |
| $T_j$ | Reporting truthfulness of user $j$ | $g_{1,1}$ | Number of $(1,1)$-groups |
| $y_i$ | $i$-th reported value | $V_{1,1}$ | Variance of a $(1,1)$-group |
| $n$ | Total users | | |

### B.1.3 Proof of Theorem 3

*Proof.* Let $\mathcal{C}$ denote the set of users colluding with the data collector and $\mathcal{T}_c = \{T_j \mid j \in \mathcal{C}\}$. The theorem follows from the bound

$$\begin{aligned}\frac{\Pr[\mathcal{M}(x_i) = y_i \mid \mathcal{T}_c]}{\Pr[\mathcal{M}(x_i') = y_i \mid \mathcal{T}_c]} &\le \frac{\max \Pr[\mathcal{M}(x_i) = y_i \mid \mathcal{T}_c]}{\min \Pr[\mathcal{M}(x_i') = y_i \mid \mathcal{T}_c]} \\ &= \frac{mp_{\max} + (n-m-1)p}{mp_{\min} + (n-m-1)q},\end{aligned}$$

for all $x_i, x_i', y_i \in \{0,1\}$.

We analyze $\Pr[\mathcal{M}(x_i) = y_i \mid \mathcal{T}_c]$. Let $u_j$ be the user paired with $u_i$ in the same group. Then

$$\begin{aligned}&\Pr[\mathcal{M}(x_i) = y_i \mid \mathcal{T}_c] \\ =&\Pr[\mathcal{M}(x_i) = y_i \mid \mathcal{T}_c, j \notin \mathcal{C}] \cdot \Pr[j \notin \mathcal{C}] + \Pr[\mathcal{M}(x_i) = y_i \mid \mathcal{T}_c, j \in \mathcal{C}] \cdot \Pr[j \in \mathcal{C}]\end{aligned}$$

Under uniform random grouping, we have

$$\Pr[j \notin \mathcal{C}] = \frac{n-m-1}{n-1}, \quad \text{and} \quad \Pr[j \in \mathcal{C}] = \frac{m}{n-1},$$

where $m$ is the number of users who collude with the data collector.

Now let us consider the two conditional cases.

**Case 1:** $j \notin \mathcal{C}$. If $u_j$ is not colluding, the probability of $u_i$ reporting $y_i$ is independent of $\mathcal{T}_c$, hence

$$\Pr[\mathcal{M}(x_i) = y_i \mid \mathcal{T}_c, j \notin \mathcal{C}] = \Pr[\mathcal{M}(x_i) = y_i].$$

Consequently,

$$\max_{x_i, y_i \in \{0,1\}} \Pr[\mathcal{M}(x_i) = y_i \mid \mathcal{T}_c, j \notin \mathcal{C}] = \Pr[\mathcal{M}(x_i) = x_i] = p,$$

where the first equality means that the maximum is achieved when reporting truthfully (i.e. $y_i = x_i$). Similarly,

$$\min_{x_i, y_i \in \{0,1\}} \Pr[\mathcal{M}(x_i) = y_i \mid \mathcal{T}_c, j \notin \mathcal{C}] = \Pr[\mathcal{M}(x_i) = 1 - x_i] = q.$$

**Case 2:** $j \in \mathcal{C}$. If $u_j$ colludes, the conditional probability of $u_i$ reporting $y_i$ depends on $T_j$, so

$$\Pr[\mathcal{M}(x_i) = y_i \mid \mathcal{T}_c, j \in \mathcal{C}] = \Pr[\mathcal{M}(x_i) = y_i \mid T_j].$$

There are four cases:

- Case 2.1: If $T_i = 1, T_j = 1$, then
$$\Pr[\mathcal{M}(x_i) = y_i \mid T_j] = \frac{\Pr[y_i = x_i, T_j = 1]}{\Pr[T_j = 1]} = \frac{p^2 + \rho pq}{p} = p + \rho q.$$
- Case 2.2: If $T_i = 0, T_j = 1$, then
$$\Pr[\mathcal{M}(x_i) = y_i \mid T_j] = \frac{\Pr[y_i = 1 - x_i, T_j = 1]}{\Pr[T_j = 1]} = \frac{(1-\rho)pq}{p} = (1-\rho)q.$$
- Case 2.3: If $T_i = 1, T_j = 0$, then
$$\Pr[\mathcal{M}(x_i) = y_i \mid T_j] = \frac{\Pr[y_i = x_i, T_j = 0]}{\Pr[T_j = 0]} = \frac{(1-\rho)pq}{q} = (1-\rho)p.$$
- Case 2.4: If $T_i = 0, T_j = 0$, then
$$\Pr[\mathcal{M}(x_i) = y_i \mid T_j] = \frac{\Pr[y_i = 1 - x_i, T_j = 0]}{\Pr[T_j = 0]} = \frac{q^2 + \rho pq}{q} = q + \rho p.$$

The maximum and minimum of these four values are

$$\begin{aligned} p_{\max} &= \max\{p + \rho q, (1-\rho)q, (1-\rho)p, q + \rho p\} \\ &= \max\{p + \rho q, (1-\rho)p\}, \\ p_{\min} &= \min\{p + \rho q, (1-\rho)q, (1-\rho)p, q + \rho p\} \\ &= \min\{q + \rho p, (1-\rho)q\}. \end{aligned}$$

It follows that

$$\max_{x_i, y_i \in \{0,1\}} \Pr[\mathcal{M}(x_i) = y_i | \mathcal{T}_c, j \in \mathcal{C}] = p_{\max}$$

and

$$\min_{x_i, y_i \in \{0,1\}} \Pr[\mathcal{M}(x_i) = y_i | \mathcal{T}_c, j \in \mathcal{C}] = p_{\min}$$

**Final result.** Substituting into the decomposition of $\Pr[\mathcal{M}(x_i) = y_i \mid \mathcal{T}_c]$ yields

$$\max_{x_i, y_i \in \{0,1\}} \Pr[\mathcal{M}(x_i) = y_i | \mathcal{T}_c] = \frac{mp_{\max}}{n-1} + \frac{n-1-m}{n-1} \cdot p,$$

Similarly, substituting these results into the expression of $\Pr[\mathcal{M}(x_i) = y_i \mid \mathcal{T}_c]$ gives

$$\min_{x_i, y_i \in \{0,1\}} \Pr[\mathcal{M}(x_i) = y_i | \mathcal{T}_c] = \frac{mp_{\min}}{n-1} + \frac{n-1-m}{n-1} \cdot q,$$

It follows that

$$\frac{\max \Pr[\mathcal{M}(x_i) = y_i \mid \mathcal{T}_c]}{\min \Pr[\mathcal{M}(x_i') = y_i \mid \mathcal{T}_c]} = \frac{mp_{\max} + (n-m-1)p}{mp_{\min} + (n-m-1)q},$$

for all $x_i, x_i', y_i \in \{0, 1\}$. The theorem is thus proved. □

### B.1.4 Proof of Theorem 4

*Proof.* First, the variance of the estimator $\hat{n}_x$ is

$$\mathrm{Var}[\hat{n}_x] = \mathrm{Var}\left[\frac{I_x - nq}{p-q}\right] = \frac{\mathrm{Var}[I_x - nq]}{(p-q)^2} = \frac{\mathrm{Var}[I_x]}{(p-q)^2}, \tag{B.1}$$

where the last equality holds because term $nq$ is a constant. Since $n = n_0 + n_1$, we have $\mathrm{Var}[\hat{n}_0] = \mathrm{Var}[n - \hat{n}_1] = \mathrm{Var}[\hat{n}_1]$. Therefore, it suffices to analyze $\mathrm{Var}[\hat{n}_1]$ in what follows.

Without loss of generality, assume that group $G_i$ consists of users $u_{2i-1}$ and $u_{2i}$ for all $1 \le i \le n/2$. Since the perturbation of different groups is independent of each other, we have

$$\mathrm{Var}[I_1] = \mathrm{Var}[\sum_{j=1}^{n} y_j] = \sum_{i=1}^{n/2} \mathrm{Var}[y_{2i-1} + y_{2i}].$$

The $n/2$ groups can be classified into three categories:

- $(1, 1)$-groups: both users have original value 1; denote the number of such groups by $g_{1,1}$;
- $(1, 0)$-groups: one user has original value 1 and the other has original value 0; denote the number of such groups by $g_{1,0}$;
- $(0, 0)$-groups: both users have original value 0; denote the number of such groups by $g_{0,0}$.

Random variables $g_{1,1}$, $g_{1,0}$, and $g_{0,0}$ satisfy $g_{1,1} + g_{1,0} + g_{0,0} = n/2$. Meanwhile, since there are $n_1$ users with original value 1, we have $2g_{1,1} + g_{1,0} = n_1$. It follows that

$$g_{1,0} = n_1 - 2g_{1,1}, \quad \text{and} \quad g_{0,0} = g_{1,1} + \frac{n}{2} - n_1,$$

which indicates that the random grouping only produces one independent random variable $g_{1,1}$.

The variance of each group's variance $\mathrm{Var}[y_{2i-1} + y_{2i}]$ depends on its type, and groups of the same type have the same variance. The reminder of the proof define $V_z = \mathrm{Var}[y_{2i-1} + y_{2i}]$ if group $G_i$ is a type-$z$ group for cleanness.

For any given $g_{1,1}$, the conditional variance of $I_1$ is given by

$$\begin{aligned}\mathrm{Var}[I_1|g_{1,1}] =& \sum_{i=1}^{n/2} \mathrm{Var}[y_{2i-1} + y_{2i}] \\ =& g_{1,1}V_{1,1} + g_{1,0}V_{1,0} + g_{0,0}V_{0,0} \\ =& g_{1,1}V_{1,1} + (n_1 - 2g_{1,1})V_{1,0} + (g_{1,1} + \frac{n}{2} - n_1)V_{0,0}.\end{aligned} \tag{B.2}$$

According to the law of total variance [3], the (unconditional) variance of $I_1$ is given by

$$\mathrm{Var}[I_1] = \mathrm{E}[\mathrm{Var}[I_1|g_{1,1}]] + \mathrm{Var}[\mathrm{E}[I_1|g_{1,1}]].$$

Next, we calculate the two terms in $\mathrm{Var}[I_1]$ one by one.

**The first term $\mathbf{E}[\mathbf{Var}[I_1|g_{1,1}]]$.** First calculate $V_1$, $V_2$ and $V_3$. For any group $G_i$,

$$\begin{aligned}\mathrm{Var}[y_{2i-1} + y_{2i}] =& \mathrm{Var}[y_{2i-1}] + \mathrm{Var}[y_{2i}] + 2\mathrm{Cov}[y_{2i-1}, y_{2i}] \\ =& \mathrm{Var}[y_{2i-1}] + \mathrm{Var}[y_{2i}] + 2(\mathrm{E}[y_{2i-1}y_{2i}] - \mathrm{E}[y_{2i-1}]\mathrm{E}[y_{2i}]).\end{aligned}$$

There are three cases.

- Case 1: If $G_i$ is $(1,1)$-group, then

$$\begin{aligned}\mathrm{E}[y_{2i-1}y_{2i}] &= \Pr[T_{2i-1} = 1, T_{2i} = 1] = \rho pq + p^2, \\ \mathrm{E}[y_{2i-1}]\mathrm{E}[y_{2i}] &= \Pr[T_{2i-1} = 1] \cdot \Pr[T_{2i} = 1] = p^2.\end{aligned}$$

- Case 2: If $G_i$ is $(1,0)$-group, then

$$\begin{aligned}\mathrm{E}[y_{2i-1}y_{2i}] &= \Pr[T_{2i-1} = 1, T_{2i} = 0] = (1-\rho)pq, \\ \mathrm{E}[y_{2i-1}]\mathrm{E}[y_{2i}] &= \Pr[T_{2i-1} = 1] \cdot \Pr[T_{2i} = 0] = pq.\end{aligned}$$

- Case 3: If $G_i$ is $(0,0)$-group, then

$$\mathrm{E}[y_{2i-1}y_{2i}] = q^2 + \rho pq, \quad \mathrm{E}[y_{2i-1}]\mathrm{E}[y_{2i}] = q^2.$$

Substituting the three cases into the conditional variance of $G_i$ yields

$$V_{1,1} = 2pq(1+\rho) \quad V_{1,0} = 2pq(1-\rho) \quad V_{0,0} = 2pq(1+\rho).$$

Substituting these three variances into the conditional variance of $I_1$ given $g_{1,1}$ (Equation (B.2)) yields

$$\mathrm{Var}[I_1|g_{1,1}] = npq + (8g_{1,1} + n - 4n_1)\rho pq.$$

Taking the expectation on both sides,

$$\begin{aligned}\mathrm{E}[\mathrm{Var}[I_1|g_{1,1}]] =& \mathrm{E}[npq + (8g_{1,1} + n - 4n_1)\rho pq] \\ =& npq + (8\mathrm{E}[g_{1,1}] + n - 4n_1)\rho pq.\end{aligned}$$

Since the expectation of $g_{1,1}$ is

$$\mathrm{E}[g_{1,1}] = \frac{n}{2} \cdot \frac{n_1(n_1-1)}{n(n-1)} = \frac{n_1(n_1-1)}{2(n-1)},$$

the first term is given by

$$\mathrm{E}[\mathrm{Var}[I_1|g_{1,1}]] = npq + \frac{(2n_1-n)^2 - n}{n-1}\rho pq.$$

**The second term $\mathbf{Var[E[}\boldsymbol{I_1|g_{1,1}}\mathbf{]]}$.** According to the definition of conditional expectation, we have

$$\begin{aligned}\mathrm{E}[I_1|g_{1,1}] &= \mathrm{E}\left[\sum_{j=1}^{n} y_j|g_{1,1}\right] = \sum_{j=1}^{n} \mathrm{E}\left[y_j|g_{1,1}\right] \\ &= n \cdot 1 \cdot \Pr(y_j = 1|g_{1,1}).\end{aligned}$$

Under JRR, whether an arbitrary user $u_j$ reports 1 or 0 only depends on the user's original value $x_j$ and the identical marginal probability distribution $\Pr[T_j]$. Since the numbers of users with the original value 1 and 0, $n_1$ and $n_0$, are predetermined. Thus, we have

$$\begin{aligned}\mathrm{E}[I_1|g_{1,1}] =& n \cdot \Pr[y_j = 1|g_{1,1}] \\ =& n_1 \cdot \Pr[T_j = 1] + (n - n_1) \cdot \Pr[T_j = 0] \\ =& n_1 p + (n - n_1)(1-p) \\ =& (2n_1 - n)p + n - n_1,\end{aligned}$$

which is a constant independent with $g_{1,1}$. It follows that

$$\mathrm{Var}[\mathrm{E}[I_1|g_{1,1}]] = 0.$$

**Final result.** Combining the two terms, we have

$$\mathrm{Var}[I_1] = npq + \frac{(2n_1 - n)^2 - n}{n-1}\rho pq.$$

Finally, plug $\mathrm{Var}[I_1]$ into $\mathrm{Var}[\hat{n}_x]$ gives

$$\mathrm{Var}[\hat{n}_x] = \frac{pq}{(p-q)^2} \cdot (n + \frac{\rho((2n_1 - n)^2 - n)}{n-1}).$$

The theorem is thus proved. □

### B.1.5 Redesign the Joint Reporting Probability

This proof corresponds to the extended JRR to non-binary data in Section 4.5. It proves the marginal reporting probability and the unbiasedness of the estimator.

*Proof.* For each user, the marginal reporting probability of $y_1$ is $\Pr[y_1] = p$ if $y_1 = x_1$ and $\Pr[y_1] = q$ if $y_1 \neq x_1$.

$$\begin{aligned}\Pr[y_1 = x_1] = &\sum_{y_2 \in [k]} \Pr[y_1 = x_1, y_2] \\ =&p^2 + \rho pq + (k-1)(pq - \frac{1}{k-1}\rho pq) \\ =&p.\end{aligned}$$

Similarly, for each $y_1 \neq x_1$ in the data domain, we have

$$\begin{aligned}\Pr[y_1 \neq x_1] = &\sum_{y_2 \in [k]} \Pr[y_1 \neq x_1, y_2] \\ =&(pq - \frac{1}{k-1}\rho pq) + (k-1)(q^2 + \frac{1}{(k-1)^2}\rho pq) \\ =&q.\end{aligned}$$

To prove the unbiasedness of $\hat{n}_x$, we need to prove $\mathrm{E}[\hat{n}_x] = n_x$, which is reduced to calculating $\mathrm{E}[I_x]$.

$$\begin{aligned}\mathrm{E}[I_x] =&n_x \cdot \Pr[x' = x] + (n - n_x) \cdot \Pr[x' \neq x] \\ =&n_x p + (n - n_x)q.\end{aligned}$$

Plugging $\mathrm{E}[I_x]$ into $\hat{n}_x$ gives

$$\begin{aligned}\mathrm{E}[\hat{n}_x] =&\frac{\mathrm{E}[I_x] - nq}{p-q} \\ =&\frac{n_x p + (n - n_x)q - nq}{p-q} \\ =&n_x,\end{aligned}$$

i.e. $\hat{n}_x$ is an unbiased estimator for $n_x$. □

### B.1.6 Proof of Theorem 5

*Proof.* According to the design of $T_{2i-1}$ and $T_{2i}$, we have

$$(T_{2i-1}, T_{2i}) = (1,1) \implies ([R < R_2], [R < R_1] + [R < R_3] - [R < R_2]) = (1,1).$$

Note that $R_1 \leq R_2 \leq R_3$ are scaled cumulative probabilities; thus, $[R < R_2] = 1$ implies $[R < R_3] = 1$, which further implies

$$\Pr[(T_{2i-1}, T_{2i}) = (1,1)] = \Pr[R < R_1] = p^2 + \rho pq.$$

Similarly, we can calculate the probabilities of the other three cases, which are given by

$$\begin{cases} \Pr[(T_{2i-1}, T_{2i}) = (1,0)] = \Pr[R_1 < R < R_2] = (1-\rho)pq, \\ \Pr[(T_{2i-1}, T_{2i}) = (0,1)] = \Pr[R_2 < R < R_3] = (1-\rho)pq, \\ \Pr[(T_{2i-1}, T_{2i}) = (0,0)] = \Pr[R > R_3] = q^2 + \rho pq. \end{cases}$$

The correctness of the protocol follows from the above probabilities. □

## B.2 Complementary Materials

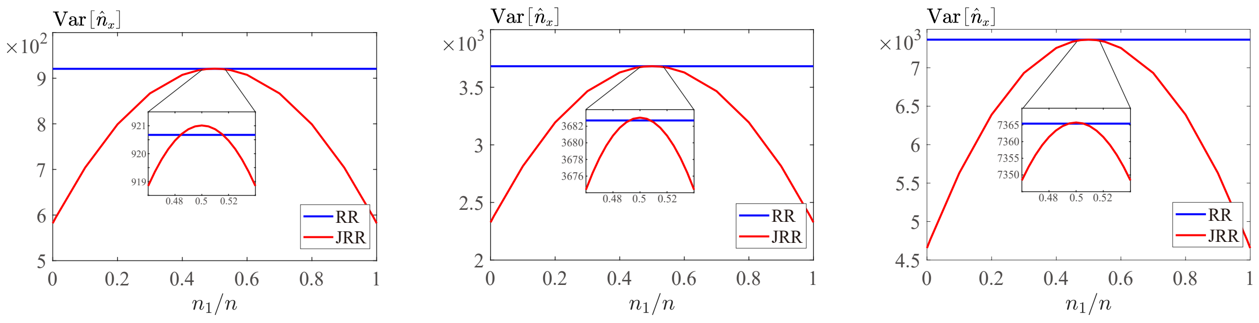
### B.2.1 Effect of $n_1/n$ (Section 4.3.4)

Figure B.1 shows the effect of $n_1/n$ on the variance of the estimator $\hat{n}_x$ for different values of the total user number $n$. As $n$ increases, the heuristic $\rho$ yields a larger reduction in variance, resulting in an almost domain-wide advantage over RR.

### B.2.2 Details of Real-world Datasets

The evaluations on real-world datasets are conducted on the following four datasets:

- Kosarak [14]: a clickstream dataset from a Hungarian news website containing about 8 million events across 41 270 pages. We randomly select 100 target pages and 20 000 click events (users). A user's true value is 1 if the visited page is among the target pages, and 0 otherwise. The frequency of clicks on the target pages is used as ground truth.



(a) $n = 1 \times 10^3$.

(b) $n = 4 \times 10^3$.

(c) $n = 8 \times 10^3$.

Figure B.1: Effect of $n_1/n$: Comparison of Var$[\hat{n}_x]$ with $\varepsilon = 1$, $p = 0.73$, and $\rho = 1 - (1/p)$.

- Amazon Rating Dataset [5]: contains over 2 million customer ratings for beauty-related products. We randomly sample 10 000 users and set the true value to 1 if the user's rating is "1 star", and 0 otherwise.
- E-commerce [126]: an e-commerce dataset on women's clothing with 23 486 records and 10 features. We use the binary variable "Recommended IND" as each user's true value.
- Census [135]: the 2010 US Census data from the Integrated Public Use Microdata Series (IPUMS). We randomly sample 10 000 records and set the true value to 1 if the group-quarter (GQ) code equals 1, and 0 otherwise.

# Appendix C

# Appendix of Chapter 5

## C.1 Proofs

### C.1.1 Proof of Lemma 1

*Proof.* We prove the inner $\max_x$ problem has a closed form. According to the definition of $\mathcal{P}_{\mathcal{M}(x)}$, it is

$$\max_x \int_{\mathcal{Y}} \mathcal{L}(y,x)\mathcal{P}_{\mathcal{M}(x)}\mathrm{d}y = \max_x \sum_{i=1}^{m} p_i \int_{l_i}^{r_i} \mathcal{L}(y,x)\mathrm{d}y.$$

Denote $f_i(x) \coloneqq \int_{l_i}^{r_i} \mathcal{L}(y,x)\mathrm{d}y$, where $\mathcal{L}(y,x) = |y-x|^p$. First, we prove that $f_i(x)$ is a convex function w.r.t. $x$. Specifically, based on the relationship between $x$ and $[l_i, r_i)$, the value of $x$ is split into three cases: (i) $x \in [a, l_i)$, (ii) $x \in [l_i, r_i)$, and (iii) $x \in [r_i, b)$. We prove the second derivative of $f_i(x)$ w.r.t. $x$ is non-negative in each case, thus $f_i(x)$ is convex.

**Case (i):** $x \in [a, l_i)$. The integral is:

$$\begin{aligned} f_i(x) &= \int_{l_i}^{r_i} |y-x|^p \mathrm{d}y = \int_{l_i}^{r_i} (y-x)^p \mathrm{d}y \\ &= \frac{(r_i - x)^{p+1} - (l_i - x)^{p+1}}{p+1}. \end{aligned}$$

The second derivative w.r.t. $x$ is

$$\frac{\partial^2}{\partial x^2} f_i(x) = p(r_i - x)^{p-1} - p(l_i - x)^{p-1} \geq 0.$$

The inequality holds because $(r_i - x)^{p-1} \geq (l_i - x)^{p-1}$ for $x \in [a, l_i)$.

**Case (ii):** $x \in [l_i, r_i)$. The integral is

$$\begin{aligned} f_i(x) &= \int_{l_i}^{r_i} |y-x|^p \mathrm{d}y \\ &= \int_{l_i}^{x} (x-y)^p \mathrm{d}y + \int_{x}^{r_i} (y-x)^p \mathrm{d}y \\ &= \frac{(x-l_i)^{p+1}}{p+1} + \frac{(r_i-x)^{p+1}}{p+1}. \end{aligned}$$

The second derivative w.r.t. $x$ is

$$\frac{\partial^2}{\partial x^2} f_i(x) = p(x-l_i)^{p-1} + p(r_i-x)^{p-1} \geq 0.$$

The inequality holds because both $(x-l_i)^{p-1}$ and $(r_i-x)^{p-1}$ are non-negative for $x \in [l_i, r_i)$.

**Case (iii):** $x \in [r_i, b)$. The integral is

$$\begin{aligned} f_i(x) &= \int_{l_i}^{r_i} |y-x|^p \mathrm{d}y = \int_{l_i}^{r_i} (x-y)^p \mathrm{d}y \\ &= \frac{(x-l_i)^{p+1} - (x-r_i)^{p+1}}{p+1}. \end{aligned}$$

The second derivative w.r.t. $x$ is

$$\frac{\partial^2}{\partial x^2} f_i(x) = p(x-l_i)^{p-1} - p(x-r_i)^{p-1} \geq 0.$$

The inequality holds because $(x-l_i)^{p-1} \geq (x-r_i)^{p-1}$ for $x \in [r_i, b)$.

**Final result.** The above three cases show that the second derivative of $f_i(x)$ w.r.t. $x \in [a, b)$ is non-negative. Thus, the non-negative weighted sum $\sum_{i=1}^{m} p_i f_i(x)$ is also convex w.r.t. $x$ [15]. According to the Bauer maximum principle [165]: any function that is convex attains its maximum at some extreme points of set. This means that the optimal $x$ is achieved at the endpoints of $x \in \mathcal{X}$, i.e. $x = a$ or $x = b$. Therefore, we have

$$\max_x \int_{\mathcal{Y}} \mathcal{L}(y, x) \mathcal{P}_{\mathcal{M}(x)} \mathrm{d}y = \max_{\{a,b\}} \int_{\mathcal{Y}} \mathcal{L}(y, x) \mathcal{P}_{\mathcal{M}(x)} \mathrm{d}y,$$

which completes the proof. □

*Remark:* This lemma can be empirically validated by the whole-domain error plots in Figure 5.8 and Figure 5.10, where the maximum of the whole-domain error is achieved at the endpoints.

### C.1.2 Proof of Lemma 2

*Proof.* The optimal $(m+1)$-piecewise mechanism and the optimal $m$-piecewise mechanism may superficially differ due to the extra piece. Thus, we define a piece-merging operation to merge the redundant pieces. We will show that if the optimal $(m+1)$-piecewise mechanism is the same as the optimal $m$-piecewise mechanism after merging redundant pieces, then increasing $m$ does not decrease the optimal error, i.e. the optimal piece number is $m$.

Assume the optimal $m$-piecewise mechanism is determined by the tuple set

$$S_m = \{(p_i, l_i, r_i) : i \in [m]\}.$$

To merge redundant pieces, we define a piece-merging operation:

$$(p_i, l_i, r_i) \uplus (p_j, l_j, r_j) := \begin{cases} (p_i, l_i, r_j) & \text{if } p_i = p_j \text{ and } i+1 = j, \\ \{(p_i, l_i, r_i), (p_j, l_j, r_j)\} & \text{otherwise.} \end{cases}$$

Because the optimal $(m+1)$-piecewise mechanism is the same as the $m$-piecewise mechanism, it follows that

$$\uplus_{i,j=1}^{m+1} S_{m+1} = \uplus_{i,j=1}^{m} S_m,$$

where $\uplus_{i,j=1}^{m} S_m$ merges all consecutive pieces with the same $p$. Denote the merged optimal $m$-piecewise mechanism as $\uplus_{i,j=1}^{m} S_m := S_m^{\uplus}$ and the piece number as $|S_m^{\uplus}| = m^*$. Because both sides of the above equation are optimal, this means that if $(p_k, l_k, r_k)$ is an arbitrary piece in the optimal $(m+1)$-piecewise mechanism, then merging it with $S_m^{\uplus}$ remains $S_m^{\uplus}$, i.e.

$$(p_k, l_k, r_k) \uplus_{i=1}^{m^*} S_m^{\uplus} = S_m^{\uplus}.$$

This premise indicates that there does not exist a piece $(p_k, l_k, r_k)$ besides $S_m^{\uplus}$ lowers the error.

Without loss of generality, we can consider the optimal $(m+2)$-piecewise mechanism, which allows an extra piece besides the optimal $(m+1)$-piecewise mechanism. We can claim that the extra piece is still captured by $S_m^{\uplus}$. The key insight is: adding an extra optimal piece to the optimal $(m+1)$-piecewise mechanism is the *same* as adding it to the optimal $m$-piecewise mechanism, because the optimal $(m+1)$-piecewise mechanism is the same as the optimal $m$-piecewise mechanism.

Since adding an extra optimal piece to the optimal $m$-piecewise mechanism remains $S_m^{\uplus}$, then for any $k \in [m+2]$, merging piece $(p_k, l_k, r_k)$ in the optimal $(m+2)$-piecewise mechanism remains an $m$-piecewise mechanism $S_m^{\uplus}$.

For $m+3$ or more, it follows the same logic. Adding an arbitrary piece is equivalent to adding it to the optimal $(m+2)$-piecewise mechanism, which is the same as adding it to the optimal $m$-piecewise mechanism. Thus, the optimal $m$-piecewise mechanism is the same as the optimal $(m+3)$-piecewise mechanism after merging redundant pieces, and so on. $\square$

*Remark:* Intuitively, the extra pieces (of the optimal $(m+1)$-piecewise mechanism and beyond) is similar to the redundant variables in optimization theory: adding more non-negative variables to a minimization objective does not decrease the *optimal* value. Here the error from each piece is a variable, and it is non-negative, which leads to the same conclusion: adding more pieces (one or more) to the optimal $m$-piecewise mechanism does not decrease the optimal error.

This lemma means that we can determine the optimal $m$-piecewise mechanism for $m = 1, 2, \ldots$, until the optimal $(m+1)$-piecewise mechanism is identical to the optimal $m$-piecewise mechanism for all $x$ and $\varepsilon$. This statement can be empirically validated by attempting to find counterexamples using larger $m$ than the optimal $m$. The source code of the framework provides scripts and results to empirically validate this lemma.

### C.1.3 Proof of Theorem 6

*Proof.* **Privacy invariant:** For any input $v, v' \in \mathcal{X}'$ and any output $y \in \mathcal{Y}'$:

$$\frac{\mathrm{pdf}\left[\mathcal{M}'(v) = y\right]}{\mathrm{pdf}\left[\mathcal{M}'(v') = y\right]} \leq \frac{p}{c} \div \frac{p}{e^{\varepsilon} c} = e^{\varepsilon}.$$

**Utility invariant:** For any $x' = cx + d \in \mathcal{X}'$, we can calculate the error difference between $\mathcal{M}'(x')$ and $\mathcal{M}'_{\text{bad}}(x')$ as follows:

$$\begin{aligned} Err(x', \mathcal{M}') - Err(x', \mathcal{M}'_{\text{bad}}) =& Err(cx+d, \mathcal{M}') - Err(cx+d, \mathcal{M}'_{\text{bad}}) \\ =& \int_{\mathcal{Y}'} \mathcal{L}(y, cx+d) \left(\mathcal{P}_{\mathcal{M}'(cx+d)} - \mathcal{P}_{\mathcal{M}'_{\text{bad}}(cx+d)}\right) \mathrm{d}y. \end{aligned}$$

Let $y_t = (y-d)/c$, then $\mathrm{d}y = c\mathrm{d}y_t$ and $y = cy_t + d$, where $y_t \in \mathcal{Y}$. The above equation is equivalent to

$$\begin{aligned} Err(x', \mathcal{M}') - Err(x', \mathcal{M}'_{\text{bad}}) =& \int_{\mathcal{Y}} \mathcal{L}(cy_t + d, cx+d) \left(\mathcal{P}_{\mathcal{M}'(cx+d)} - \mathcal{P}_{\mathcal{M}'_{\text{bad}}(cx+d)}\right) c\mathrm{d}y_t \\ =& \int_{\mathcal{Y}} \mathcal{L}(cy_t + d, cx+d) \frac{1}{c} \left(\mathcal{P}_{\mathcal{M}(x)} - \mathcal{P}_{\mathcal{M}_{\text{bad}}(x)}\right) c\mathrm{d}y_t. \end{aligned}$$

The last equality holds due to the definition of $\mathcal{T} : \mathcal{M} \to \mathcal{M}'$. For $\ell_p$-similar error metric $\mathcal{L}$ (i.e. $\mathcal{L}(y, x) := |y - x|^p$), it follows that

$$\mathcal{L}(cy_t + d, cx + d) = \mathcal{L}(cy_t, cx) = c^p \mathcal{L}(y_t, x).$$

Thus, the above difference of $Err$ is equivalent to

$$\begin{aligned}Err(x', \mathcal{M}') - Err(x', \mathcal{M}'_{\text{bad}}) =& c^p \int_{\mathcal{Y}} \mathcal{L}(y_t, x) \left(\mathcal{M}(x) - \mathcal{M}_{\text{bad}}(x)\right) \mathrm{d}y_t \\ =& c^p \left(Err(x, \mathcal{M}) - Err(x, \mathcal{M}_{\text{bad}})\right) \leq 0,\end{aligned}$$

due to the known fact $c > 0$ and $Err(x, \mathcal{M}) - Err(x, \mathcal{M}_{\text{bad}}) \leq 0$. □

*Remark:* Intuitively, this theorem is to prove: if $\mathcal{M}$ is a better mechanism than $\mathcal{M}_{\text{bad}}$ on $\mathcal{X}$, then it is still a better mechanism than $\mathcal{M}_{\text{bad}}$ after linearly mapping their outputs to $\mathcal{X}'$.

### C.1.4 Proof of Theorem 7

*Proof.* Appendix C.2.4 provides the formalized procedure. Following this procedure, we show the optimal GPM under $\mathcal{X} \to \mathcal{Y} = [0, 1) \to [0, 1)$ and $\mathcal{L}(y, x) = |y - x|$.

The variables in TPM are $p$, $l$, and $r$. Since it is a family of probability distributions, the normalization constraint is

$$(r - l) \cdot p + (1 - (r - l)) \cdot p/e^{\varepsilon} = 1,$$

which means the length of the central piece is

$$s := r - l = \frac{e^{\varepsilon} - p}{p(e^{\varepsilon} - 1)}.$$

Without loss of generality, assume $x = 0$ is the optimal point ($x = 1$ is symmetric). The optimization problem for solving the optimal $p$ is

$$\begin{aligned}\arg\min_p \left( \int_0^s y \cdot p \,\mathrm{d}y + \int_s^1 y \cdot \frac{p}{e^{\varepsilon}} \,\mathrm{d}y \right) =& \arg\min_p \left( \frac{s^2}{2} \left( p - \frac{p}{e^{\varepsilon}} \right) + \frac{1}{2} \frac{p}{e^{\varepsilon}} \right) \\ =& \arg\min_p \frac{1}{2} \left( \frac{(e^{\varepsilon} - p)^2}{p(e^{\varepsilon} - 1)e^{\varepsilon}} + \frac{p}{e^{\varepsilon}} \right).\end{aligned}$$

To solve the optimal $p$, we take the first-order derivative w.r.t. $p$ and set it to 0, i.e.

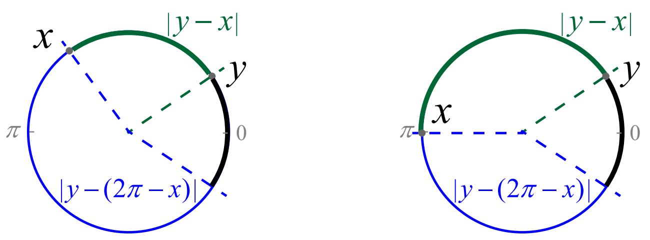
$$\frac{\partial}{\partial p} \left( \frac{(e^{\varepsilon} - p)^2}{p(e^{\varepsilon} - 1)e^{\varepsilon}} + \frac{p}{e^{\varepsilon}} \right) = 0,$$

This leads to $p = e^{\varepsilon/2}$. Then

$$s = \frac{e^{\varepsilon/2} - 1}{e^{\varepsilon} - 1}.$$

Having solved $p$ and $s$, the optimal $r$ is $r = l + s$. Then the optimal $l$ is determined by

$$\arg\min_l \int_0^l (x - y) \cdot \frac{p}{e^{\varepsilon}} \,\mathrm{d}y + \int_l^x (x - y) \cdot p \,\mathrm{d}y + \int_x^{l+s} (y - x) \cdot p \,\mathrm{d}y + \int_{l+s}^1 (y - x) \cdot \frac{p}{e^{\varepsilon}} \,\mathrm{d}y$$



(a) $x = 3\pi/4$. (b) $x = \pi$.

Figure C.1: Examples of $\mathcal{L}_{\text{mod}}(y, x)$ w.r.t. $x$. Given a specific $y$, the shorter between the blue and green arcs is $\mathcal{L}_{\text{mod}}(y, x)$. $\max_x \mathcal{L}_{\text{mod}}(y, x)$ is achieved at $x = \pi$.

This is a univariate optimization problem w.r.t. $l$. Moreover, it is a two-order polynomial w.r.t. $l$ and can be solved by analyzing the first-order and second-order derivatives. The solution is

$$l = \frac{2x(p - pe^{-\varepsilon}) - s(p - pe^{-\varepsilon})}{2(p - pe^{-\varepsilon})} = x - \frac{s}{2}.$$

Note that when $x - s/2 < 0$, the above $l$ is outside the domain $[0, 1)$. In this case, the optimal $l$ is $l = 0$.

Relating the above deduction to Theorem 7, the term $s/2$ corresponds to $C$. Then the optimal $p$ is $e^{\varepsilon/2}$, $l = x - C$, and $r = x + C$ when $x \in [C, 1 - C)$, which completes the proof for $\mathcal{L}(y, x) = |y - x|$. The proof for $\mathcal{L}(y, x) = |y - x|^2$ is similar. □

*Remark:* This proof is the same as finding the optimal 3-piecewise distribution in domain $[0, 1)$. The source code of our framework provides the validation.

### C.1.5 Proof of Lemma 3

*Proof.* Note that $\mathcal{L}_{\text{mod}}(y, x) = \min(\mathcal{L}(y, x), \mathcal{L}(y, 2\pi - x))$. The key observation is that for any $y$ and $\ell_p$-similar distance metric $\mathcal{L}$, we have

$$\max_x \mathcal{L}_{\text{mod}}(y, x) = \mathcal{L}_{\text{mod}}(y, \pi) = \mathcal{L}(y, \pi).$$

Figure C.1 illustrates the intuition. For any fixed $y$, it compares the length of $|y-x|$ and $|y-(2\pi-x)|$ w.r.t $x \in [0, 2\pi)$. $\mathcal{L}_{\text{mod}}(y, x)$ is determined by the minimum of the two, and $\max_x \mathcal{L}_{\text{mod}}(y, x)$ will achieved at $x = \pi$.

The remained proof is more intuitive than the proof of Lemma 1, as the inner integrand $\mathcal{L}_{\text{mod}}(y, x)$ has a *unique* maximum at $x = \pi$ for any $y \in [l_i, r_i)$, making the swap of $\max_x$ and integration valid.

Specifically, we have

$$\begin{aligned}\max_x \sum_i^m p_i \int_{l_i}^{r_i} \mathcal{L}_{\text{mod}}(y,x)\mathrm{d}y &= \sum_i^m p_i \int_{l_i}^{r_i} \max_x \mathcal{L}_{\text{mod}}(y,x)\mathrm{d}y \\ &= \sum_i^m p_i \int_{l_i}^{r_i} \mathcal{L}(y,\pi)\mathrm{d}y\end{aligned}$$

holds trivially, because all other $x$ values always result in smaller $\mathcal{L}_{\text{mod}}(y,x)$. Therefore, for any values of $p_i \geq 0$, $l_i \leq r_i$, the integration of $\mathcal{L}_{\text{mod}}(y,x)$ is bounded by the integration of $\mathcal{L}(y,\pi)$. □

### C.1.6 Proof of Theorem 10

*Proof.* We prove the unbiasedness of $\mathcal{M}$. The expectation of the given $\mathcal{M}$ is

$$\begin{aligned}\mathrm{E}[\mathcal{M}(x)] &= \int_{-C}^{C+1} y \cdot \mathcal{P}_{\mathcal{M}(x)}\mathrm{d}y \\ &= \int_{-C}^{l} y\frac{p}{e^\varepsilon}\mathrm{d}y + \int_{l}^{r} yp\mathrm{d}y + \int_{r}^{C+1} y\frac{p}{e^\varepsilon}\mathrm{d}y \\ &= \frac{r^2-l^2}{2}\left(p - \frac{p}{e^\varepsilon}\right) + \left(C+\frac{1}{2}\right)\frac{p}{e^\varepsilon}.\end{aligned}$$

Denote $s \coloneqq (2C+1)(C-1)/(2C)$, which rewrites $l$ and $r$ as

$$l = \frac{C+1}{2}x - \frac{C-1}{4} - \frac{s}{2}, \quad r = \frac{C+1}{2}x - \frac{C-1}{4} + \frac{s}{2}.$$

Then the above $\mathrm{E}[\mathcal{M}(x)]$ is equivalent to

$$\begin{aligned}&\frac{(C+1)sx-(C-1)s/2}{2} \cdot \frac{4C}{(C^2-1)(2C+1)} + \frac{e^{-\varepsilon/2}}{2} \\ =&x - \frac{C-1}{2(C+1)} + \frac{e^{-\varepsilon/2}}{2} = x,\end{aligned}$$

leading to $\mathrm{E}[\mathcal{M}] = x$, i.e. $\mathcal{M}$ is unbiased. □

## C.2 Complementary Materials

### C.2.1 Detailed Comparison with PTT (Section 5.1.2)

Piecewise transformation technique (PTT) [101] is a framework for 3-piecewise mechanisms. It shows that (i) many PTT mechanisms are asymptotically optimal when used to obtain an unbiased estimator for mean of numerical data, and (ii) there is a PTT that reaches the theoretical lower bound on variance.

Under the viewpoint of GPM, type-I PTT focuses on TPM that constrains the probabilities of the central interval ($p$) and the two side intervals ($q$) as

$$p = \frac{1}{2ak}\frac{e^{\varepsilon}}{e^{\varepsilon}-1}, \quad q = \frac{e^{\varepsilon}}{k(e^{\varepsilon}-1)},$$

where $a$ and $k$ are parameters to be determined. Type-II PTT focuses on TPM that constrains $p$ and $q$ as

$$p = \frac{1}{ak}\frac{e^{\varepsilon}}{e^{\varepsilon}-1}, \quad q = \frac{e^{\varepsilon}}{k(e^{\varepsilon}-1)}.$$

Therefore, PTT is still a specific case of the TPM framework. Additionally, when discussing optimality of PTT, it gives a value of $a$ for type-I PTT but do not provide the optimal $k$.

### C.2.2 Related Optimality (Section 5.2.1)

In this dissertation, the optimality of GPM is defined with respect to: (i) the worst-case $\ell_p$-similar error metric, (ii) bounded numerical domains $\mathcal{X} \to \mathcal{Y}$ and mechanisms based on piecewise distributions, (iii) minimization of error value (not asymptotic or order-of-magnitude optimality), and (iv) without post-processing. $\ell_p$-similar error metrics are natural choices for evaluating data utility [60, 69, 150]. Bounded numerical domains are common in real-world applications. Focusing on error values allows for more precise comparisons between different mechanisms. By excluding post-processing, we can analyze the optimality of the mechanism itself, which provides a more fundamental understanding than considering the mechanism combined with a specific post-processing.

Other types of optimality have been explored in the literature, particularly for variants of Laplace mechanisms. The staircase mechanism [60] adopts the same utility model without prior knowledge or post-processing as this dissertation. It claims optimality under specific assumptions, one of which is that a staircase (piecewise) distribution *can* achieve the optimal error. The mechanism demonstrates better $\ell_1$-error performance than the Laplace mechanism on $\mathcal{Y} = (-\infty, \infty)$, and its asymptotic optimality has been formally proven. *Universal optimality* is another type of optimality, defined from the perspective of a user's prior knowledge and post-processing ability [64]. In this utility model, the user observes the output of the mechanism and selects another value based on the output and their prior knowledge, i.e. under a Bayesian utility framework. Formally, if the user's prior is denoted as $p_i$ on the data domain $i \in N$ (i.e. a discrete domain) and the user's post-processing is represented as a remap $z_{i,j}$ that reinterprets the output of the mechanism (on the sensitive value $i$) to $j$, then the utility model is defined as

$$Err(i) = \sum_{i \in N} p_i \sum_{j \in N} z_{i,j} \cdot \mathcal{L}(i,j).$$

This utility model incorporates the user's prior knowledge and post-processing ability. A mechanism is called universally optimal if, for any prior $p_i$, there exists an optimal remap $z_{i,j}$. Under this utility model, it was proven that the truncated geometric mechanism (a discretized version of the Laplace mechanism) can achieve universal optimality for count queries* and a legal error metric $\mathcal{L}(i,j)$. Such universal optimality was shown to be unachievable for more complex queries [16]. Under the same utility model, the universal optimality was extended to the truncated Laplace mechanism for a bounded numerical domain $\mathcal{X} = [0, 1]$ by approximating the geometric mechanism with the Laplace mechanism and post-processing [54].

These results do not hold in our utility model, i.e. utility model without prior and post-processing. Figure 5.13 has shown that OGPM generally has a smaller error than the truncated Laplace mechanism, especially when the privacy parameter $\varepsilon$ is not small, indicating the sub-optimality of the truncated Laplace mechanism in the absence of using prior and post-processing.

### C.2.3 Directions for Analytically Proving Optimal $m = 3$ (Section 5.2.2)

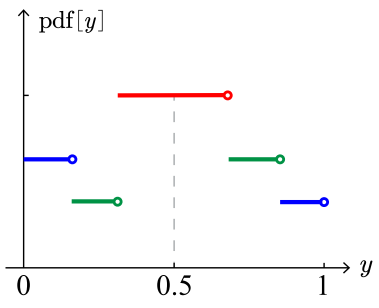
This appendix outlines two potential directions for analytically proving that the optimal $m$ is 3, along with the challenges associated with each approach.

Mathematically, finding the optimal $m$-piecewise mechanism is equivalent to identifying the optimal $m$-piecewise distribution under an $\ell_p$-similar error metric. It is seemingly true that the optimal $m$ is 3: if the optimal $m$-piecewise distribution is not 3 but 4 or more, we can always shift the probability mass from the two side intervals (i.e. other pieces) to the central interval, thereby reducing the error. At the very least, the following fact holds:

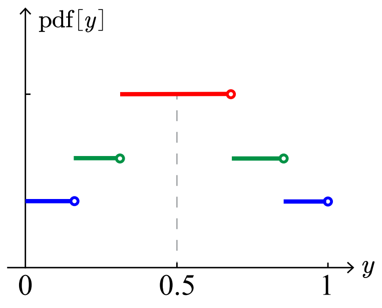
**Fact 1.** *The optimal $m$-piecewise distribution has a strict staircase shape, i.e. the probability density of the central interval is greater than that of the two side intervals.*

Figure C.2 illustrates this fact. Moving pieces while keeping their probabilities unchanged clearly maintains both the $\varepsilon$-LDP constraint and the probability normalization constraint. This observation reduces the problem to proving that a 3-staircase distribution can achieve the same optimal error as a 4-staircase distribution under the $\varepsilon$-LDP and probability normalization constraints.

*This is in the centralized DP setting, where the data curator holds the dataset and uses *one* mechanism.



(a) Non-staircase distribution.



(b) Staircase distribution.

Figure C.2: A non-staircase distribution (left) can always be shifted into a staircase distribution (right) by moving some pieces closer to $x$, which reduces the error.

**Direction 1:** If we can further move the green piece in Figure C.2b "into" the red central piece while keeping the probabilities of the red and blue pieces unchanged, i.e. transform it into a 3-staircase distribution while ensuring a decrease in the error, then we can prove that the optimal $m$ is 3. However, this is challenging, as it breaks the probability normalization constraint (i.e. the sum of probabilities is no longer 1), requiring adjustments to the probabilities of each piece to satisfy the $\varepsilon$-LDP constraint. The difficulty lies in ensuring that these adjustments will indeed decrease the error.

**Direction 2:** Another approach is to formulate the problems for 3-staircase and 4-staircase distributions as two constrained optimization problems. The goal would be to prove that the optimal error of the 3-staircase distribution is equivalent to that of the 4-staircase distribution. Ideally, these two multi-variable optimization problems could be solved analytically, resulting in two closed-form error expressions w.r.t. $x$ and $\varepsilon$, thereby completing the proof for any $x$ and $\varepsilon$ by showing that the two expressions are equal. This direction aligns with our framework. However, the challenge lies in the complexity of solving such multi-variable optimization problems analytically. This is why we rely on an off-the-shelf optimization solver, which, while effective, only provides numerical solutions for specific $x$ and $\varepsilon$ values.

### C.2.4 Analytical Deduction for the Optimal GPM (Section 5.2.2)

If the optimal GPM under $\mathcal{L}$ is proved to be TPM, then the closed-form optimal can be derived by deduction.

Denote $\mathcal{X} = [a, b)$ and $\mathcal{Y} = [\tilde{a}, \tilde{b})$. Notice that the normalization constraint of probability is

$$(r - l) \cdot p + [(b - a) - (r - l)] \cdot \frac{p}{e^{\varepsilon}} = 1.$$

This means the central interval length is

$$s := r - l = \frac{1}{p(1-e^{-\varepsilon})} - \frac{\tilde{b}-\tilde{a}}{e^{\varepsilon}-1}.$$

If the minimal worst-case error is achieved at $x = a$ in Lemma 1, then solving for the optimal $p$ is reduced to

$$\arg\min_{p} \left( \int_{\tilde{a}}^{s} \mathcal{L}(y,a) p \, \mathrm{d}y + \int_{s}^{\tilde{b}} \mathcal{L}(y,a) \frac{p}{e^{\varepsilon}} \mathrm{d}y \right),$$

which is a univariate optimization problem w.r.t. $p$ and can be solved analytically. With the solved $p$, Formulation (5.3) is also reduced to a univariate optimization problem w.r.t. $l$:

$$\arg\min_{l} \left( \int_{\tilde{a}}^{l} P_1 \, \mathrm{d}y + \int_{l}^{l+s} P_2 \, \mathrm{d}y + \int_{l+s}^{\tilde{b}} P_1 \, \mathrm{d}y \right),$$

where $P_1 = \mathcal{L}(y,x)p$ and $P_2 = \mathcal{L}(y,x)p/e^{\varepsilon}$. This univariate optimization problem solves the optimal $l$, and the optimal $r$ is $r = l + s$. Note that $l$ and $r$ should be restricted in $[\tilde{a}, \tilde{b})$ when analyzing the first-order derivative.

### C.2.5 MSE of the Optimal GPM (Section 5.2.3)

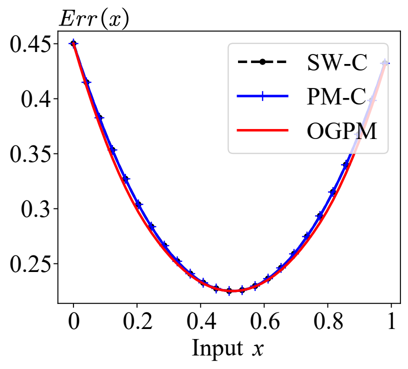
Denote $p_\varepsilon$ and $C$ as the same as the instantiations of $\mathcal{M}$ in Theorem 7. The MSE of $\mathcal{M}$ is

1. If $x \in [0, C)$:
$$\frac{p_\varepsilon}{3}\left((2C-x)^3 + x^3\right) + \frac{p_\varepsilon}{3e^{\varepsilon}}\left((1-x)^3 - (2C-x)^3\right).$$
2. If $x \in [C, 1-C)$:
$$\frac{p_\varepsilon}{3e^{\varepsilon}}\left(-2C^3 + 3x^2 - 3x + 1\right) + \frac{p_\varepsilon}{3}\left(2C^3\right).$$
3. If $x \in [1-C, 1)$:
$$\frac{p_\varepsilon}{3e^{\varepsilon}}\left((1-2C-x)^3 + x^3\right) + \frac{p_\varepsilon}{3}\left((1-x)^3 - (1-2C-x)^3\right).$$

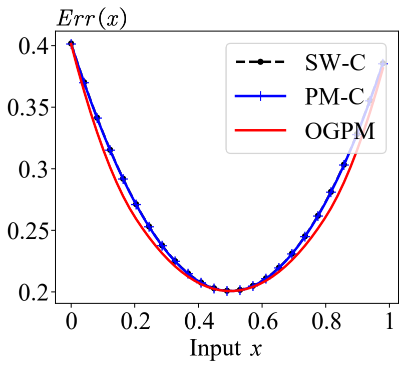
For example, when $x = 0$, the MSE of $\mathcal{M}$ is

$$\mathrm{MSE}[\mathcal{M}(0)] = \frac{p_\varepsilon}{3}\left(8C^3\right) + \frac{p_\varepsilon}{3e^{\varepsilon}}\left(1 - 8C^3\right).$$

Setting $\varepsilon = 1$ results in $\mathrm{MSE}[\mathcal{M}(0)] = 0.22$ of OGPM. As a comparison, SW [95], which also designed for $\mathcal{X} = [0,1)$, has an MSE of 0.29 at $x = 0$.



(a) Privacy parameter $\varepsilon = 0.4$.



(b) Privacy parameter $\varepsilon = 0.8$.

Figure C.3: Whole-domain error comparison in the classical domain with error metric $\mathcal{L} = |y - x|$.

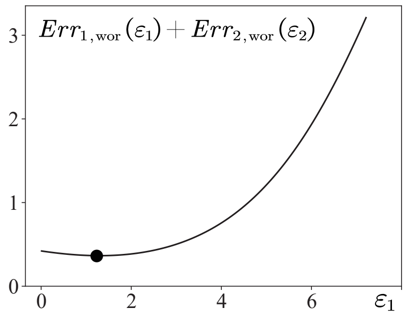
### C.2.6 Optimal Assignment of Privacy Parameter and an Example (Section 5.5)

The objective function to minimize the given total error in 2D polar coordinates is

$$\min_{\varepsilon_1, \varepsilon_2} Err_{1,\text{wor}}(\varepsilon_1) + Err_{2,\text{wor}}(\varepsilon_2),$$

where $Err_{1,\text{wor}}(\varepsilon_1)$ and $Err_{2,\text{wor}}(\varepsilon_2)$ are the worst-case errors of the classical domain and the circular domain, respectively.

Without loss of generality, we can assume the polar coordinate data is in $[0, 1) \times [0, 2\pi)$ and $\mathcal{L} = |y_1 - x_1|^2$, then $Err_{1,\text{wor}}(\varepsilon_1)$ and $Err_{2,\text{wor}}(\varepsilon_2)$ are already given by our MSE analysis. However, the above optimization problem as a function of $\varepsilon_1$ and $\varepsilon_2$ is generally non-linear, thus hard to be analytically solved. Therefore, a simple and practical way to find the optimal $\varepsilon_1$ and $\varepsilon_2$ is numerical testing.



Under distance metric $\mathcal{L}(y, x) = |y - x|^2$, the worst-case error of the classical domain $[0, 1)$ is achieved at $x = 0$. Therefore, $Err_{1,\text{wor}}$ equals to MSE$[\mathcal{M}(0)]$ calculated before. For the circular domain $[0, 2\pi)$, the worst-case error $Err_{2,\text{wor}}$ is stated in Theorem 9. If $\varepsilon = 1 + 2\pi$ and we assign $\varepsilon_1$ to the classical domain and $\varepsilon_2 = \varepsilon - \varepsilon_1$ to the circular domain, then the total error is plotted in the right figure. In this figure, the optimal assignment is $\varepsilon_1 = 1.32$ and $\varepsilon_2 = 5.69$.

From the curve of the total error, we can see that $\varepsilon_2$ affects the total error more than $\varepsilon_1$. Even if $\varepsilon_1$ is set to 0, the total error is not significantly affected, and it is still is significantly smaller than the case of $\varepsilon_2 = 0$. This is because the circular domain has a larger range than the classical domain, thus the error of the circular domain is more sensitive to the privacy parameter.

Note that the optimality for the polar coordinate data is under the specific error metric $\mathcal{L}_{2\mathrm{D}} := \mathcal{L}(y_1, x_1) + \mathcal{L}_{\mathrm{mod}}(y_2, x_2)$. If the error metric differs, the optimal error might not be preserved.

### C.2.7 Comparison under Small $\varepsilon$ (Section 5.6.1)

Figure C.3 presents the whole-domain error comparison of OGPM, PM-C, and SW-C under smaller $\varepsilon$ values, specifically $\varepsilon = 0.4$ and $\varepsilon = 0.8$. In these scenarios, all three mechanisms approach the uniform distribution more closely compared to cases with larger $\varepsilon$. Consequently, their errors are also more similar to each other. Statistically, when $\varepsilon = 0.4$, the error of OGPM is at most 0.008 smaller than that of PM-C and SW-C. For $\varepsilon = 0.8$, the error of OGPM is at most 0.015 smaller than that of PM-C and SW-C.

### C.2.8 Expected Error of the B-Laplace Mechanism (Section 5.6.1)

The B-Laplace mechanism redefines a Laplace-shaped distribution on a bounded domain as the perturbation mechanism. For the data domain $\mathcal{X} \to \mathcal{Y} = [0,1) \to [0,1)$, the B-Laplace mechanism is defined as follows:

**Definition 20** (Bounded Laplace Mechanism, adapted from [73])**.** *The B-Laplace mechanism $\mathcal{M}(x) : [0,1) \to [0,1)$ is given by the probability density function (PDF) as follows:*

$$\mathrm{pdf}[\mathcal{M}(x) = y] = \frac{1}{C_y} \cdot \frac{1}{2b} e^{-\frac{|y-x|}{b}} \quad \forall y \in [0,1),$$

*where $b$ is the scale parameter, and $C_y = \int_0^1 \frac{1}{2b} e^{-\frac{|y-x|}{b}} \mathrm{d}x$ is the normalization constant.*

According to Theorem 3.5 and Corollary 4.5 in [73], the B-Laplace mechanism satisfies $\varepsilon$-LDP whenever $b \geq 1/\varepsilon$. Using the best scale parameter $b = 1/\varepsilon$, the normalization constant becomes $C_y = (1 - e^{-\varepsilon})/2$. We can compute the expected $\ell_1$ error of the B-Laplace mechanism as follows (this computation is not included in [73]):

$$\begin{aligned} Err(x, \mathcal{M}) &= \int_0^1 |y - x| \cdot \mathrm{pdf}[\mathcal{M}(x) = y] \mathrm{d}y \\ &= \int_0^1 |y - x| \cdot \frac{1}{C_y} \cdot \frac{1}{2b} e^{-\frac{|y-x|}{b}} \mathrm{d}y \\ &= \frac{\varepsilon}{1 - e^{-\varepsilon}} \int_0^1 |y - x| e^{-\varepsilon |y-x|} \mathrm{d}y. \end{aligned}$$

This integral can be numerically computed by the Python library function `scipy.stats.laplace.expect()`

or analytically solved. The final result for the expected error is

$$\frac{2-(1+\varepsilon x)e^{-\varepsilon x}-(1+\varepsilon(1-x))e^{-\varepsilon(1-x)}}{\varepsilon(1-e^{-\varepsilon})},$$

which is a closed-form expression w.r.t. $x$ and $\varepsilon$.

# Appendix D

# Appendix of Chapter 6

## D.1 Proofs

### D.1.1 LDP Proof of Definition 9

The proof applies to all piecewise-based mechanisms, including the direction perturbation mechanism in Definition 9 and the distance perturbation mechanism in Definition 10.

*Proof.* For any two sensitive directions $\varphi_1$, $\varphi_2$ and any $\varphi' \in [0, 2\pi)$, we have:

$$\frac{pdf[\mathcal{M}_\circ(\varphi_1) = \varphi']}{pdf[\mathcal{M}_\circ(\varphi_2) = \varphi']} \leq \frac{p_\varepsilon}{p_\varepsilon / \exp(\varepsilon)} = \exp(\varepsilon).$$

The inequality holds because any $\varphi'$ is sampled with a probability of at most $p_\varepsilon$ and at least $p_\varepsilon / \exp(\varepsilon)$, regardless of whether the input is $\varphi_1$, $\varphi_2$ or any other value. □

### D.1.2 Proof of Theorem 12

*Proof.* Denote TraCS-D in Algorithm 1 as $\mathcal{M}$. We need to prove that $\mathcal{M}$ satisfies $\varepsilon$-LDP for each location $\tau \in \mathcal{S}$, i.e.

$$\forall \tau_1, \tau_2, \tau' \in \mathcal{S} : \frac{pdf[\mathcal{M}(\tau_1) = \tau']}{pdf[\mathcal{M}(\tau_2) = \tau']} \leq \exp(\varepsilon).$$

In TraCS-D, each location $\tau$ is represented in a direction-distance space with unique coordinates $(\varphi, r(\varphi))$. The key insight of the proof is that the perturbation of $\varphi$ and $r(\varphi)$ satisfies $\varepsilon_d$-LDP and $(\varepsilon - \varepsilon_d)$-LDP, respectively. Then TraCS-D ensures $\varepsilon$-LDP for the location space $\mathcal{S} = \mathcal{D}_\varphi \otimes \mathcal{D}_{r(\varphi)}$.

We provide proofs by the Composition Theorem 11 and by computing the 2D *pdf* ratios in the LDP definition.

**By Composition Theorem 11.** The definitions of piecewise-based mechanisms $\mathcal{M}_\circ$ and $\mathcal{M}_-$ imply that they satisfy LDP. Specifically, for $\mathcal{M}_\circ(\varphi;\varepsilon_d)$,

$$\forall \varphi_1, \varphi_2, \varphi' \in [0, 2\pi) : \frac{pdf[\mathcal{M}_\circ(\varphi_1) = \varphi']}{pdf[\mathcal{M}_\circ(\varphi_2) = \varphi']} \le \exp(\varepsilon_d),$$

is guaranteed by Definition 9 of $\mathcal{M}_\circ$. Similarly, $\mathcal{M}_-(\overline{r}(\varphi);\varepsilon-\varepsilon_d)$ in Definition 10 ensures $(\varepsilon-\varepsilon_d)$-LDP for $\overline{r}(\varphi)$. For clarity, let $\overline{r} \coloneqq \overline{r}(\varphi)$, which yields

$$\forall \overline{r}_1, \overline{r}_2, \overline{r}' \in [0, 1) : \frac{pdf[\mathcal{M}_-(\overline{r}_1) = \overline{r}']}{pdf[\mathcal{M}_-(\overline{r}_2) = \overline{r}']} \le \exp(\varepsilon - \varepsilon_d).$$

The subsequent linear mapping from $\overline{r} \in [0, 1)$ to $r \in \mathcal{D}_{r(\varphi')}$ is a post-processing step that does not affect the LDP property. Specifically, denote the linear mapping as $g : [0, 1) \to \mathcal{D}_{r(\varphi')}$, and denote $r_1 = g(\mathcal{M}_-(\overline{r}_1))$ and $r_2 = g(\mathcal{M}_-(\overline{r}_2))$ as two random variables. It is clear that

$$\forall r_1, r_2, r' \in \mathcal{D}_{r(\varphi')} : \frac{pdf[r_1 = r']}{pdf[r_2 = r']} \le \exp(\varepsilon - \varepsilon_d),$$

because the linear mapping $g$ is a deterministic function without randomness. Thus, $\mathcal{M}_-$ ensures $(\varepsilon - \varepsilon_d)$-LDP for $\mathcal{D}_{r(\varphi')}$.

Combining these results with Sequential Composition Theorem 11, we can conclude that $\mathcal{M} = (\mathcal{M}_\circ, \mathcal{M}_-)$ ensures $\varepsilon$-LDP for each location $\tau = (\varphi, r(\varphi)) \in \mathcal{S}$. Consequently, the entire trajectory with $n$ locations satisfies $n\varepsilon$-LDP.

**By computing the 2D *pdf* ratio.** Assuming two sensitive locations $\tau_1 = (\varphi_1, \overline{r}_1)$ and $\tau_2 = (\varphi_2, \overline{r}_2)$, where $\overline{r}_1$ and $\overline{r}_2$ are normalized distances from the reference location along $\varphi_1$ and $\varphi_2$, respectively. For any output $\tau' = (\varphi', \overline{r}')$, the 2D *pdf* of getting it from $\tau_1$ and $\tau_2$ are

$$pdf[\mathcal{M}(\tau_1) = \tau'] = pdf[\mathcal{M}_\circ(\varphi_1) = \varphi'] \cdot pdf[\mathcal{M}_-(\overline{r}_1) = \overline{r}'],$$
$$pdf[\mathcal{M}(\tau_2) = \tau'] = pdf[\mathcal{M}_\circ(\varphi_2) = \varphi'] \cdot pdf[\mathcal{M}_-(\overline{r}_2) = \overline{r}'],$$

where $\mathcal{M}_-$ is applied to the *same* normalized distance space (from the reference location to the boundary of $S$ along $\varphi'$). The ratio is bounded by $\exp(\varepsilon_d) \cdot \exp(\varepsilon - \varepsilon_d) = \exp(\varepsilon)$ due to the definition of $\mathcal{M}_\circ$ and $\mathcal{M}_-$. Consequently, the entire trajectory with $n$ locations satisfies $n\varepsilon$-LDP. □

### D.1.3 Proof of Theorem 13

*Proof.* We prove the $\Theta(e^{-\varepsilon/2})$ convergence rate by calculating the MSE of the perturbation mechanism. Specifically, the MSE of $\mathcal{M}_-$ is calculated as follows:

$$\text{MSE}[\mathcal{M}_-(\overline{r})] = \int_0^1 (\overline{r}^* - \overline{r})^2 pdf[\mathcal{M}_-(\overline{r}) = \overline{r}^*] d\overline{r}^*.$$

The worst-case MSE is achieved when $\overline{r} = 0$ or $\overline{r} = 1$, i.e. at the endpoints (similar to the variance calculation in [150]). In this case,

$$\begin{aligned}\text{MSE}[\mathcal{M}_-(0)] &= \int_0^{2C} (\overline{r}^* - 0)^2\, p_\varepsilon d\overline{r}^* + \int_{2C}^{1} (\overline{r}^* - 0)^2\, \frac{p_\varepsilon}{e^\varepsilon} d\overline{r}^* \\ &= \frac{8C^3 p_\varepsilon}{3} + \frac{(1-8C^3)p_\varepsilon}{3e^\varepsilon}.\end{aligned}$$

with $p_\varepsilon$ and $C$ defined in Definition 10. We omit the calculation for $\overline{r} = 1$ as it is symmetric to $\overline{r} = 0$. Plugging in the values of $p_\varepsilon$ and $C$, we can simplify the MSE using big $\mathcal{O}$ notation:

$$\begin{aligned}\text{MSE}[\mathcal{M}_-(0)] &= \frac{1}{3}e^{-\varepsilon/2} + \frac{(e^{\varepsilon/2}-1)^3}{3e^{\varepsilon/2}(e^\varepsilon - 1)^2} \\ &= \frac{1}{3}e^{-\varepsilon/2} + \mathcal{O}(e^{-\varepsilon}).\end{aligned}$$

Therefore, as $\varepsilon \to \infty$, the worst-case MSE of $\mathcal{M}_-$ converges to zero at a rate of $\Theta(e^{-\varepsilon/2})$. The MSE of $\mathcal{M}_\circ$ can be calculated in the same way, resulting in a convergence rate of $\Theta(e^{-\varepsilon/2})$ as well. □

### D.1.4 Proof of Theorem 14

*Proof.* Denote TraCS-C in Algorithm 2 as $\mathcal{M}$. We show that $\mathcal{M}$ satisfies $\varepsilon$-LDP for each location $\tau \in \mathcal{S}$. In TraCS-C, each location $\tau$ is represented in the Cartesian space $\mathcal{D}_a \times \mathcal{D}_b$ with unique coordinates $(a, b)$. Thus, it suffices to prove that the perturbations of $a$ and $b$ each satisfy $\varepsilon/2$-LDP.

From Theorem 12, $\mathcal{M}_-(\overline{a}; \varepsilon/2)$ satisfies $\varepsilon/2$-LDP for the normalized coordinate $\overline{a} \in [0, 1)$. The subsequent linear mapping from $\overline{a} \in [0, 1)$ to $a \in \mathcal{D}_a$ is post-processing and therefore preserves the LDP guarantee. Hence, $\mathcal{M}_-$ ensures $\varepsilon/2$-LDP over $\mathcal{D}_a$. By the same argument, $\mathcal{M}_-$ also ensures $\varepsilon/2$-LDP over $\mathcal{D}_b$.

Therefore, $\mathcal{M} \coloneqq (\mathcal{M}_-, \mathcal{M}_-)$ satisfies $\varepsilon$-LDP for each location $\tau = (a, b) \in \mathcal{S}$ by the Sequential Composition Theorem 11. Equivalently, one can verify the guarantee by directly bounding the 2D *pdf* ratio induced by $\mathcal{M}$. Consequently, the entire trajectory with $n$ locations satisfies $n\varepsilon$-LDP. □

## D.2 Complementary Materials

### D.2.1 Space-Dependence of Indistinguishability (Section 3.3)

Indistinguishability of a data point is always relative to a specified data space, which determines the set of alternatives that an adversary may try to distinguish it from. Accordingly, indistinguishability cannot be meaningfully discussed without first specifying the underlying space, for two main reasons.

**(i) LDP perspective.** The LDP guarantee is defined with respect to a particular input domain. If two algorithms operate on the same domain and use the same privacy parameter $\varepsilon$, then their privacy guarantees are comparable (in the sense of the LDP definition); if their domains differ, then their guarantees may not be directly comparable, even when they share the same $\varepsilon$.

**(ii) Information-theoretic perspective.** For an $\varepsilon$-LDP mechanism $\mathcal{M}$, the mutual information $I(x; \mathcal{M}(x))$ is upper bounded by a function of both $\varepsilon$ and the size of the input space, i.e. $I(x; \mathcal{M}(x)) \leq \mathcal{O}(\varepsilon, |\mathcal{X}|)$, where $|\mathcal{X}|$ denotes the cardinality of the input space $\mathcal{X}$. A similar bound can be found in [43].[*]

This observation is particularly relevant to trajectory collection or synthesis under LDP when discretization-based methods are used. Even for a fixed continuous area, different discretization strategies (e.g. uniform grids, adaptive grids, or different sets of points of interest (POIs)) induce different discrete location spaces with different sizes and spatial layouts. Consequently, the effective indistinguishability provided by the same LDP mechanism can vary substantially across discretizations, even when the same $\varepsilon$ is used.

### D.2.2 Limitations of the Exponential Mechanism (Section 3.3 and Section 6.1.3)

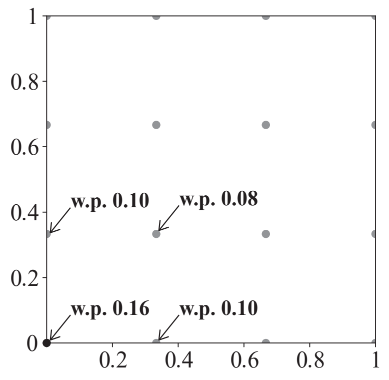
#### High Computational Complexity

The biggest limitation of the Exponential mechanism is the complexity in calculating the score function and sampling from the probability distribution. Before perturbing location $x$, the Exponential mechanism requires computing the score function $d(x, y)$ for every possible location $y$ in the location space. If the location space has $m$ locations, the time complexity of computing the score function for a single location is $\mathcal{O}(m)$, which is computationally expensive for large-scale spaces. Moreover, ATP's dynamic strategy to constrain the location space for perturbation makes it need to recompute the score function for each location, resulting in $\mathcal{O}(m)$ time complexity, where $m$ is the size of the location space.

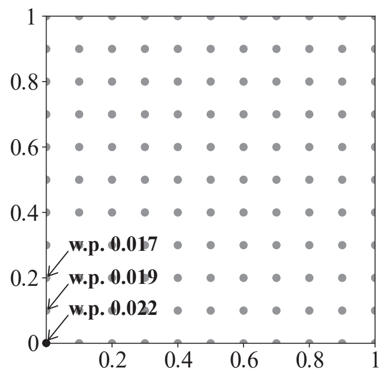
The complexity of sampling from the Exponential mechanism is also $\mathcal{O}(m)$, as its cumulative distribution function (CDF) is an $m$-piece function. Sampling requires comparing a random number with the CDF values of all $m$ locations.

In contrast, we have seen that piecewise-based perturbation mechanisms in TraCS do not require

[*]Intuitively, LDP enforces *pairwise* indistinguishability but does not directly account for how indistinguishability accumulates over many alternatives, which can be captured by mutual information.

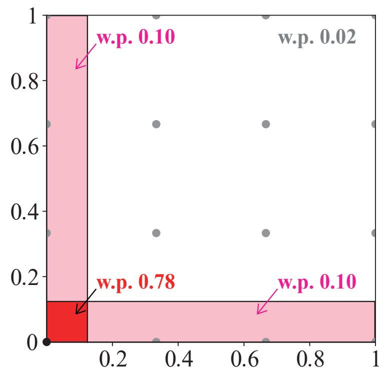


(a) $m = 4 \times 4$ (gray points)

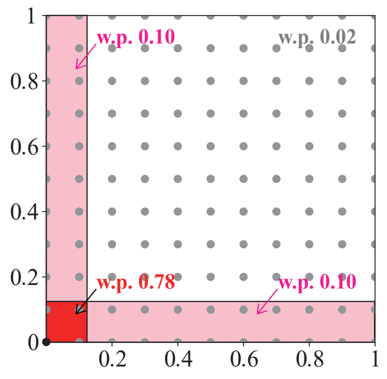


(b) $m = 11 \times 11$ (gray points)

Figure D.1: The Exponential mechanism $\mathcal{M}_{\text{exp}}(0,0)$ with $\varepsilon = 4$ on two discrete location spaces within $[0,1] \times [0,1]$. The number and arrangement of points affect the mechanism.



(a) $\mathcal{S} = [0,1] \times [0,1]$



(b) $\mathcal{S} = [0,1] \times [0,1]$

Figure D.2: TraCS-C mechanism $\mathcal{M}_{\text{TraCS-C}}(0,0)$ with $\varepsilon = 4$ on $\mathcal{S} = [0,1] \times [0,1]$, which encompasses both discrete location spaces in Figure D.1. The sampling probability is defined over areas rather than individual points.

score functions, and they sample from a 3-piece CDF at each perturbation, resulting in a negligible constant time complexity, i.e. $\Theta(1)$, for each perturbation.

## Affected by the Number and Arrangement of Locations

Because the sampling probability depends on the score function, the spatial distribution and relative distances between locations further affect the probability assigned to each location.

Figure D.1 illustrates the Exponential mechanism $\mathcal{M}_{\text{exp}}$ applied to the sensitive location $\tau = (0,0)$ with $\varepsilon = 4$ on two discrete location spaces within $[0,1] \times [0,1]$. Specifically, Figure D.1a presents a $4 \times 4$ grid of locations, while Figure D.1b depicts an $11 \times 11$ grid. We can observe that the sampling probabilities of $\mathcal{M}_{\text{exp}}$ differ for the same sensitive location $\tau = (0, 0)$, even though both location spaces are contained within the same $[0, 1] \times [0, 1]$ area. When the number of locations increases and the arrangement becomes denser (i.e. $m = 11 \times 11$), the sampling probabilities for each location become smaller and more uniform.

In contrast, Figure D.2 illustrates the TraCS-C mechanism $\mathcal{M}_{\text{TraCS-C}}$ applied to the same sensitive location $\tau = (0, 0)$ with $\varepsilon = 4$ over the $[0, 1] \times [0, 1]$ area. Unlike the Exponential mechanism, TraCS-C is designed for continuous spaces, where the sampling probability is defined over areas rather than individual points.

To apply TraCS-C to the discrete spaces shown in Figure D.1, one can simply round each perturbed location to its nearest discrete point. Since this rounding is a post-processing step, it does not affect the LDP guarantee.

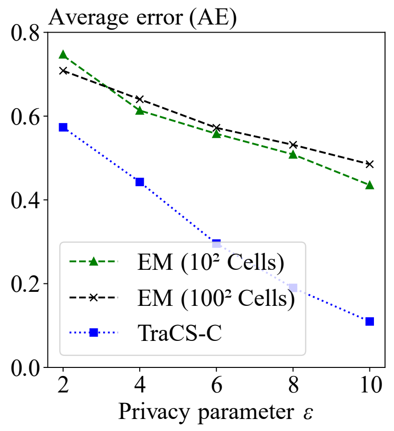
### D.2.3 Limitations of Applying Discrete LDP Mechanisms to Continuous Spaces (Section 3.3)

Although a continuous space can be discretized before applying discrete LDP mechanisms, this approach inherits the privacy–utility–efficiency issues of discrete methods and introduces additional challenges in choosing an appropriate discretization strategy.

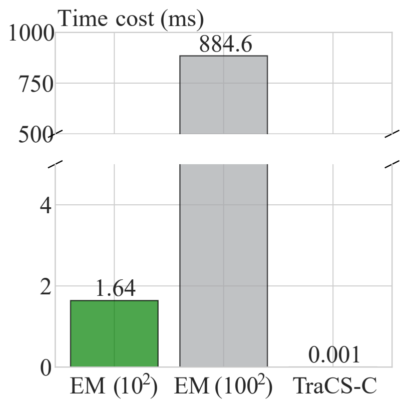
**Gridding the continuous space.** A common discretization approach is to partition the continuous space into uniform grid cells and treat each cell as a discrete input and output of the discrete mechanism. The reported cell can then be post-processed by randomly sampling a location within that cell.

**Dilemma.** However, the gridding approach faces a fundamental dilemma in choosing the grid cell size, which directly affects the privacy–utility–efficiency tradeoff. (i) If the grid cells are too large (i.e. coarse-grained gridding), the mechanism is more likely to output the true cell, but the intra-cell error (i.e. the distance between the true location and the post-processed location sampled within the same cell) can be large. (ii) If the grid cells are too small (i.e. fine-grained gridding), the intra-cell error can be reduced, but the probability of outputting the true cell decreases. Meanwhile, the computational cost increases with the number of cells.

We empirically evaluate the gridding approach under different cell sizes in a continuous space and compare it with TraCS-C. Specifically, we consider the $[0, 1] \times [0, 1]$ location space and two grid resolutions: $10 \times 10$ and $100 \times 100$. For a fixed location, we generate 500 perturbed locations using (i) the gridding approach with the Exponential mechanism (EM), where the score is defined by the



(a) Error of the gridding approach with different cell sizes.



(b) Average time cost of perturbing one location.

Figure D.3: Performance of the gridding approach with different cell sizes in a continuous space. Coarse-grained grids (e.g. $10 \times 10$) perform better when $\varepsilon$ is small, whereas fine-grained grids (e.g. $100 \times 100$) perform better when $\varepsilon$ is large, at the cost of higher computational time. In contrast, TraCS achieves better utility than both gridding baselines across all $\varepsilon$ values, with negligible time overhead.

distance between cell centers, and (ii) TraCS-C. We then report the average error and the average time required to perturb a single location. Figure D.3 summarizes the results. In the small-$\varepsilon$ regime, coarse-grained grids (e.g. $10 \times 10$) outperform fine-grained grids (e.g. $100 \times 100$), whereas in the large-$\varepsilon$ regime, fine-grained grids perform better. This trend is consistent with the above dilemma: it is difficult to select a single grid resolution that performs well across different $\varepsilon$ values. In contrast, TraCS-C consistently achieves better utility than both gridding baselines for all tested $\varepsilon$ values, while incurring negligible time overhead.

### D.2.4 Dominant Sector Comparison Between the Strawman Approach and TraCS-D (Example 10)

For direction perturbation, a small-size and high-probability dominant sector is desirable for more accurate perturbation. However, these two properties are generally conflicting with each other due to the LDP constraint: a smaller dominant sector typically has a lower probability. We will show that TraCS-D achieves a better trade-off between the size and probability of the dominant sector compared to the strawman approach with any $k$ value. Figure D.4 shows two exemplary quantitative comparisons between them.

**Small $\varepsilon$ region.** In this case, the dominant sector of TraCS-D (red solid line) is larger than the strawman approach, but the probability of the dominant sector (black solid line) is significantly

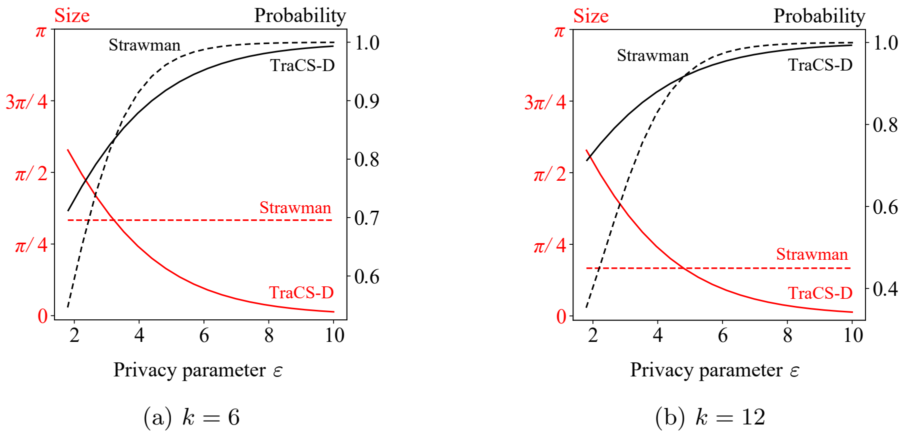


Figure D.4: Comparison of the size and probability of the dominant sector between TraCS-D and the strawman approach. TraCS-D (solid line) achieves a better tradeoff between the size and probability of the dominant sector compared to the strawman approach (dashed line), having better adaptiveness across different $\varepsilon$ regions.

higher, especially when $k$ is large. This indicates that, although the dominant sector of TraCS-D is larger in this region, it is sampled with a higher probability.

**Large $\varepsilon$ region.** In such cases, the dominant sector of TraCS-D is significantly smaller than the strawman approach, while maintaining almost the same probability as the strawman approach.

We can observe the effect of $k$ on the strawman approach. Larger $k$ values in the strawman approach reduce the inner-sector error compared to TraCS-D, but also significantly decrease the probability of the dominant sector when $\varepsilon$ is small. Conversely, smaller $k$ values increase the probability of the dominant sector when $\varepsilon$ is small, but result in a large inner-sector error when $\varepsilon$ is large. In contrast, TraCS-D adaptively balances these trade-offs by adjusting the dominant sector's size and probability according to $\varepsilon$.

### D.2.5 Detailed Form of $|\mathcal{D}_{r(\varphi)}|$ (Section 6.2.2)

For each location $\tau_i$, denote its directions to the four endpoints of the rectangular location space as $\varphi_1, \varphi_2, \varphi_3, \varphi_4$. We have:

$$\begin{aligned}
\varphi_1 &= \text{atan2}\left(b_{\text{end}} - b_i, a_{\text{end}} - a_i\right), \\
\varphi_2 &= \text{atan2}\left(b_{\text{end}} - b_i, a_{\text{sta}} - a_i\right), \\
\varphi_3 &= \text{atan2}\left(b_{\text{sta}} - b_i, a_{\text{sta}} - a_i\right) + 2\pi, \\
\varphi_4 &= \text{atan2}\left(b_{\text{sta}} - b_i, a_{\text{end}} - a_i\right) + 2\pi.
\end{aligned}$$

Then, $|\mathcal{D}_{r(\varphi)}|$ can be derived using trigonometric functions, which leads to the following four cases:

$$|\mathcal{D}_{r(\varphi)}| = \begin{cases} \dfrac{a_{\text{end}} - a_i}{\cos\varphi} & \text{if } \varphi \in [0, \varphi_1) \cup [\varphi_4, 2\pi), \\ \dfrac{b_{\text{end}} - b_i}{\sin\varphi} & \text{if } \varphi \in [\varphi_1, \varphi_2), \\ \dfrac{a_i - a_{\text{sta}}}{-\cos\varphi} & \text{if } \varphi \in [\varphi_2, \varphi_3), \\ \dfrac{b_i - b_{\text{sta}}}{-\sin\varphi} & \text{if } \varphi \in [\varphi_3, \varphi_4), \end{cases}$$

depending on the direction $\varphi$ from reference location $\tau_i = (a_i, b_i)$ to the boundary of the rectangular location space $[a_{\text{sta}}, a_{\text{end}}] \times [b_{\text{sta}}, b_{\text{end}}]$.

### D.2.6 Redesigned SW (Section 6.2.5)

We redesign the SW mechanism to achieve $\varepsilon$-LDP for the circular space $[0, 2\pi)$ and the linear space $[0, 1)$. The key idea is to transform the original sampling distribution to these spaces while preserving the LDP constraint. Similar to Definition 9, the redesigned SW mechanism for direction perturbation is defined as follows.

**Definition 21** (Redesigned SW for direction perturbation)**.** *Given a sensitive direction $\varphi$ and a privacy parameter $\varepsilon$, redesigned SW for direction perturbation is a mechanism $\mathcal{M}_\circ : [0, 2\pi) \to [0, 2\pi)$ defined by:*

$$pdf[\mathcal{M}_\circ(\varphi) = \varphi'] = \begin{cases} p_\varepsilon & \text{if } \varphi' \in [l_{\varphi,\varepsilon}, r_{\varphi,\varepsilon}), \\ p_\varepsilon / \exp(\varepsilon) & \text{otherwise}, \end{cases}$$

*where $p_\varepsilon = \frac{1}{2\pi\varepsilon}(\exp(\varepsilon) - 1)$ is the sampling probability, and $[l_{\varphi,\varepsilon}, r_{\varphi,\varepsilon})$ is the sampling interval that*

$$l_{\varphi,\varepsilon} = \left(\varphi - \pi \frac{\exp(\varepsilon)(\varepsilon - 1) + 1}{(\exp(\varepsilon) - 1)^2}\right) \mod 2\pi,$$
$$r_{\varphi,\varepsilon} = \left(\varphi + \pi \frac{\exp(\varepsilon)(\varepsilon - 1) + 1}{(\exp(\varepsilon) - 1)^2}\right) \mod 2\pi.$$

Similar to Definition 10, the redesigned SW mechanism for distance perturbation is defined as follows.

**Definition 22** (Redesigned SW for distance perturbation)**.** *Given a sensitive distance $\overline{r}(\varphi)$ and a privacy parameter $\varepsilon$, redesigned SW for distance perturbation is a mechanism $\mathcal{M}_- : [0, 1) \to [0, 1)$ that*

$$pdf[\mathcal{M}_-(\overline{r}(\varphi)) = \overline{r}'(\varphi)] = \begin{cases} p_\varepsilon & \text{if } \overline{r}'(\varphi) \in [u, v), \\ p_\varepsilon / \exp(\varepsilon) & \text{otherwise}, \end{cases}$$

*where* $p_\varepsilon = (\exp(\varepsilon) - 1)/\varepsilon$ *is the sampling probability, and* $[u, v)$ *is the sampling interval that*

$$[u, v) = \begin{cases} \overline{r}(\varphi) + [-C, C) & \textit{if } \overline{r}'(\varphi) \in [C, 1 - C), \\ [0, 2C) & \textit{if } \overline{r}'(\varphi) \in [0, C), \\ [1 - 2C, 1) & \textit{otherwise}, \end{cases}$$

*with* $C = (\exp(\varepsilon)(\varepsilon - 1) + 1)/(2(\exp(\varepsilon) - 1)^2)$.

Compared with Definition 9 and Definition 10, the redesigned SW mechanisms have a larger $p_\varepsilon$ and a narrower $[l, r)$ or $[u, v]$ interval for sampling. We can treat the redesigned SW mechanisms as a more aggressive perturbation mechanism: it tries to sample from a narrow dominant sector with a higher probability ($p_\varepsilon$), which also leads to a higher probability ($p_\varepsilon / \exp(\varepsilon)$) of non-dominant sectors due to the LDP constraint.

### D.2.7 Experimental Results for Extensions (Section 6.2.5)

#### Effect of Different Piecewise-based Mechanisms

Figure D.5 compares the performance of different piecewise-based mechanisms for TraCS-D. In addition to the mechanisms defined in Definition 9 and Definition 10, we name the redesigned SW for TraCS-D as TraCS-D (R-SW), as detailed in Appendix D.2.6. We can observe that TraCS-D consistently exhibits smaller errors than TraCS-D (R-SW) across all the $\varepsilon$ values. Statistically, the mean AE of TraCS-D is 52.6% of TraCS-D (R-SW) across all the $\varepsilon$ values. This difference comes from the MSE of their sampling distributions, where Definition 9 and Definition 10 have smaller MSE than R-SW.

#### TraCS-D for Circular Area

Figure D.6 illustrates an example of TraCS-D applied to a circular area $\mathcal{S} = [0, 2\pi) \otimes [0, 1)$. We set the sensitive location $\tau$ as $(\pi, 0.5)$ and the privacy parameter $\varepsilon = 5$, then collect 50 random samples of perturbed locations using TraCS-D. In this case, the dominant area is $[0.87\pi, 1.13\pi) \otimes [0.32, 0.67)$, as determined by the perturbation mechanisms in Definition 9 and Definition 10. We can observe that the perturbed locations are concentrated in the dominant area.

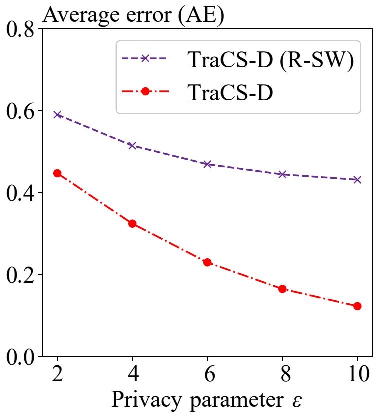


Figure D.5: Different piecewise-based mechanisms.

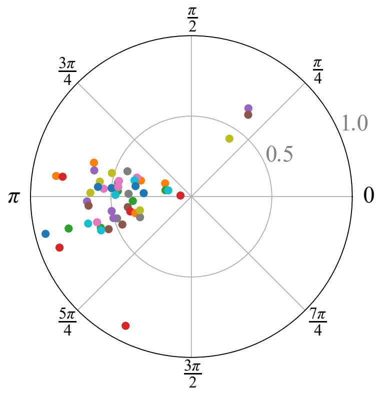


Figure D.6: Samples of TraCS-D at $\tau = (\pi, 0.5)$ for circular area.

### D.2.8 2D Laplace Mechanism with Truncation (Section 6.3.1)

For 2D location spaces equipped with the Euclidean distance, a standard Laplace-based approach is the Planar Laplace mechanism [7].[†] It was originally proposed for *geo-indistinguishability*, and it also provides an LDP guarantee with a calibrated sensitivity.

**Definition 23** (Planar Laplace mechanism). *Let $\mathcal{X} \subseteq \mathbb{R}^2$ be the input domain. The Planar Laplace mechanism $\mathcal{M}_{\mathrm{PL}} : \mathcal{X} \to \mathbb{R}^2$ is defined by*

$$\mathcal{M}_{\mathrm{PL}}(x) = x + \eta,$$

*where $\eta \in \mathbb{R}^2$ is a noise vector with density*

$$pdf[\eta] = \frac{\varepsilon^2}{2\pi} \exp\bigl(-\varepsilon \|\eta\|_2\bigr).$$

**Sensitivity calibration for an LDP guarantee.** By the triangle inequality, for any $x_1, x_2 \in \mathcal{X}$ and any $y \in \mathbb{R}^2$, we have

$$\frac{pdf[y - x_1]}{pdf[y - x_2]} = \exp\Bigl(\varepsilon\bigl(\|y - x_2\|_2 - \|y - x_1\|_2\bigr)\Bigr) \leq \exp\bigl(\varepsilon \|x_1 - x_2\|_2\bigr).$$

Therefore, over the domain $\mathcal{X}$, the mechanism satisfies $(\varepsilon \cdot \mathrm{diam}(\mathcal{X}))$-LDP, where $\mathrm{diam}(\mathcal{X})$ is the diameter of $\mathcal{X}$. Equivalently, to ensure $\varepsilon$-LDP for all pairs of inputs in $\mathcal{X}$, one can run the Planar Laplace mechanism with parameter $\varepsilon / \mathrm{diam}(\mathcal{X})$.

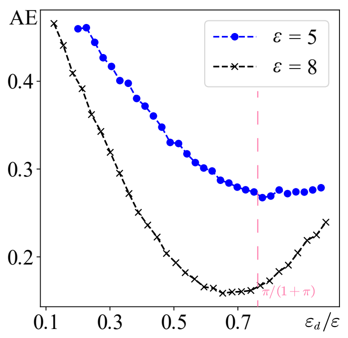
[†]We omit the standard (truncated) Laplace mechanism for $\mathbb{R}$, as it is designed for $L_1$ distance and has been shown to have worse data utility compared to piecewise-based mechanisms [184].

**Truncation for bounded location spaces.** The Planar Laplace mechanism outputs perturbed locations in $\mathbb{R}^2$, which may fall outside the bounded location space $\mathcal{S}$. A common practice is to truncate the output to $\mathcal{S}$, e.g. by projecting it to the nearest point in $\mathcal{S}$. This truncation is a post-processing step and does not affect the LDP guarantee.

### D.2.9 Privacy Parameter Assignment in TraCS-D (Section 6.3.1)

In TraCS-D, the overall perturbation error is jointly determined by the direction perturbation and the distance perturbation. These two components are controlled by $\varepsilon_d$ for $\mathcal{M}_\circ$ and $\varepsilon - \varepsilon_d$ for $\mathcal{M}_-$, respectively. As a result, the (theoretical) error bound of TraCS-D depends on both $\varepsilon$ and the privacy split $\varepsilon_d$. In principle, the best choice of $\varepsilon_d$ is the one that minimizes this bound.

However, the optimal split depends on $\varepsilon$ and on the shape of $\mathcal{S}$, which makes a universal closed-form choice impractical. Using the same experimental setup as in the Figure 6.4a, we empirically evaluate how $\varepsilon_d$ affects the error of TraCS-D; the results are shown in the figure on the right. We observe a consistent pattern: for each fixed $\varepsilon$, the error decreases initially and then increases as $\varepsilon_d$ varies from 0 to the full budget $\varepsilon$.

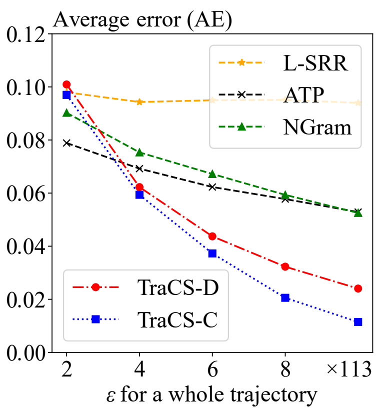


Across all tested $\varepsilon$ values, the minimizer $\varepsilon_d/\varepsilon$ is close to our heuristic split $\varepsilon_d = \pi/(1+\pi)$ (dashed pink line), which supports the use of this heuristic in practice.

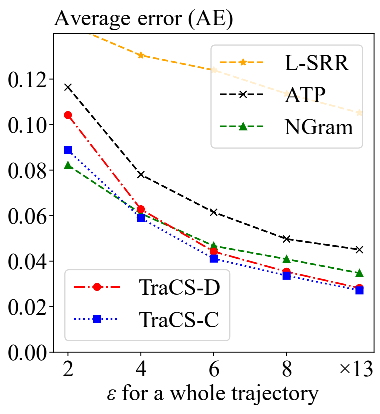
### D.2.10 Privacy Parameter Assignment for a Whole Trajectory (Section 6.3.2)

We compare TraCS with NGram, L-SRR, and ATP under a trajectory-level privacy parameter assignment on the TKY and CHI datasets. The average trajectory lengths are 113 (TKY) and 13 (CHI). Accordingly, we assign a trajectory-level budget of $\varepsilon \times 113$ for TKY and $\varepsilon \times 13$ for CHI. This scaling makes the trajectory-level setting more comparable to the location-level $\varepsilon$ assignment in Figure 6.9. The results are shown in Figure D.7.

We observe that TraCS has larger AEs than the discrete-space mechanisms when $\varepsilon$ is small, but its errors decrease rapidly as $\varepsilon$ increases. As a result, TraCS outperforms the discrete-space mechanisms in the large-$\varepsilon$ regime. Compared with assigning $\varepsilon$ at the location level in Figure 6.9, the overall trends remain similar across all mechanisms: TraCS starts to outperform all other discrete-space mechanisms when $\varepsilon \approx 4$ per location on both datasets. Among the discrete-space mechanisms, ATP

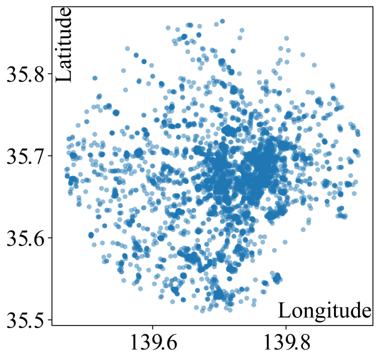


(a) TKY dataset

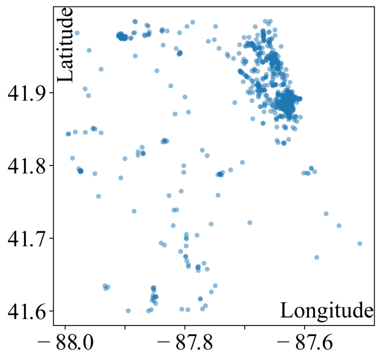


(b) CHI dataset

Figure D.7: Comparison with trajectory-level $\varepsilon$ assignment. The average trajectory length is 113 for TKY and 13 for CHI. TraCS's AE decreases fast as $\varepsilon$ increases, outperforming other discrete-space mechanisms when $\varepsilon$ is large.



(a) TKY location space



(b) CHI location space

Figure D.8: Location spaces for TKY and CHI datasets. TraCS operates over the whole rectangular area that encloses the discrete location space of each city, while other discrete-space mechanisms operate only on the discrete locations (blue points).

achieves the lowest AEs on TKY, whereas NGram (i.e. the Exponential mechanism with a reachable set) attains the lowest AEs on CHI, due to CHI's smaller location space than TKY.

As discussed in Section 6.3.2, the larger average errors (AEs) of TraCS in the small-$\varepsilon$ regime can be attributed to its enlarged effective location space. TraCS is designed for continuous spaces and perturbs locations over the rectangular area that encloses the discrete location space of each city, as illustrated in Figure D.8. The discrete location set in TKY is comparatively dense and evenly distributed, whereas the location set in CHI is much sparser. Consequently, the gap between the discrete set and its enclosing rectangle is larger for CHI, which amplifies the utility loss of TraCS and explains its weaker performance on CHI (relative to discrete-space mechanisms) compared with

TKY.

# Appendix E

# Appendix of Chapter 7

## E.1 Proofs

### E.1.1 Proof of Theorem 17

*Proof.* From the definition of $\mathcal{I}_\delta(\mathcal{M}(x))$, we have

$$\Pr[\mathcal{I}_\delta(\mathcal{M}(x)) = \mathcal{M}(x)] = 1 - \delta.$$

Since $\mathcal{M}(x)$ satisfies $\varepsilon$-LDP, it means the privacy loss $\mathcal{L}_{\mathcal{M},x_1,x_2}(\tilde{x}) \leq \varepsilon$ always holds. Thus, with probability at least $1 - \delta$, the $\mathcal{L}_{\mathcal{M},x_1,x_2}(\tilde{x})$ is bounded by $\varepsilon$, i.e.

$$\Pr[\mathcal{L}_{\mathcal{M},x_1,x_2}(\tilde{x}) \leq \varepsilon] \geq 1 - \delta,$$

which proves that $\mathcal{I}_\delta(\mathcal{M}(x))$ satisfies $(\varepsilon, \delta)$-PAC LDP. $\square$

### E.1.2 Proof of Theorem 18 and Discussion

#### Extended Gaussian Mechanism

The proof generally follows the structure of Dwork's proof of the Gaussian mechanism on page 261 [48]. Their proof assumes $\varepsilon \leq 1$ for the soundness of the $\ln(\varepsilon, \Delta f)$ function. We eliminate this limitation by refining the proof technique and providing a new noise bound.

*Proof.* For the Gaussian distribution and an original data $x_1 \in [0, 1]$, the probability of seeing $x_1 + \tilde{x}$ with $\tilde{x} \sim \mathcal{N}(0, \sigma^2)$ is given by the probability density function at $\tilde{x}$, i.e. $\mathrm{pdf}[\tilde{x}]$. The probabilities of seeing *this same value* when the original data is $x_2 \in [0,1]$ belongs to a set $\{\mathrm{pdf}[\tilde{x}+\Delta] : \Delta \in [-1,1]\}$. Thus, the privacy loss is

$$\mathcal{L}_{\mathcal{M},x_1,x_2}(\tilde{x}) = \ln\left(\frac{\mathrm{pdf}[\tilde{x}]}{\mathrm{pdf}[\tilde{x}+\Delta]}\right) = \ln\frac{\exp(-\frac{1}{2\sigma^2}\cdot\tilde{x}^2)}{\exp(-\frac{1}{2\sigma^2}\cdot(\tilde{x}+\Delta)^2)}$$
$$= \frac{1}{2\sigma^2}\cdot\left(2\tilde{x}\Delta+\Delta^2\right).$$

Since $\tilde{x} \sim \mathcal{N}(0,\sigma^2)$, the privacy loss is distributed as a Gaussian random variable with mean $\Delta^2/(2\sigma^2)$ and variance $\Delta^2/\sigma^2$. We can see that the Gaussian mechanism can not satisfy pure $\varepsilon$-LDP for any $\varepsilon$ because $\tilde{x} \in (-\infty,\infty)$, meaning the privacy loss is unbounded.

Nonetheless, the privacy loss random variable can be bounded by $\varepsilon$ with a certain probability $1-\delta$. Specifically, letting $Z \sim \mathcal{N}(0,1)$, the privacy loss can be rewritten as a random variable:

$$\mathcal{L}_{\mathcal{M},x_1,x_2}(\tilde{x}) \sim \frac{\Delta}{\sigma}\cdot Z + \frac{\Delta^2}{2\sigma^2}.$$

In the remainder of the proof, we will find a value of $\sigma$ that ensures $\mathcal{L}_{\mathcal{M},x_1,x_2}(\tilde{x}) \le \varepsilon$ with probability at least $1-\delta$.

**Step 1: Gaussian tail bound.** When $\mathcal{L}_{\mathcal{M},x_1,x_2}(\tilde{x}) \ge \varepsilon$, we have

$$\frac{\Delta}{\sigma}\cdot Z + \frac{\Delta^2}{2\sigma^2} \ge \varepsilon, \text{which means } |Z| \ge \frac{\varepsilon\sigma - \frac{\Delta^2}{2\sigma}}{\Delta}.$$

Since $\Delta \in [-1,1]$, we can set $\Delta = 1$ to minimize the right-hand side of the above inequality, yielding the worst-case $|Z|$, which gives $|Z| \ge \varepsilon\sigma - 1/(2\sigma)$. We need to bound the probability of this event by $\delta$:

$$\Pr\left[|Z| \ge \varepsilon\sigma - \frac{1}{2\sigma}\right] \le \delta, \text{implying } \Pr\left[Z \ge \varepsilon\sigma - \frac{1}{2\sigma}\right] \le \frac{\delta}{2}.$$

By the Gaussian tail bound (Chernoff bound):

$$\Pr[Z \ge t] \le \exp\left(\frac{-t^2}{2}\right),$$

we have

$$\exp\left(\frac{-\left(\varepsilon\sigma - \frac{1}{2\sigma}\right)^2}{2}\right) \le \frac{\delta}{2}.$$

**Step 2: Solving $\boldsymbol{\sigma}$.** Define $C = \sqrt{-2\ln(\delta/2)}$. We can rewrite the above inequality as

$$\left(\varepsilon\sigma - \frac{1}{2\sigma}\right)^2 \ge C^2.$$

Taking the square root of both sides and rearranging gives us two cases:

$$\varepsilon\sigma - \frac{1}{2\sigma} \ge C \quad \text{or} \quad \varepsilon\sigma - \frac{1}{2\sigma} \le -C.$$

Therefore, the best choice of $\sigma$ is the one that satisfies:

$$\varepsilon\sigma - \frac{1}{2\sigma} = \pm C.$$

Solving this quadratic equation gives us two solutions for $\sigma$:

$$\sigma = \frac{\pm C \pm \sqrt{C^2 + 2\varepsilon}}{2\varepsilon}.$$

Since $\sigma$ must be positive, we take the positive root, which gives us the final solution:

$$\sigma = \frac{\sqrt{-2\ln(\delta/2)} + \sqrt{-2\ln(\delta/2) + 2\varepsilon}}{2\varepsilon},$$

which is equivalent to the result in Theorem 18. □

**Comparison with Dwork's Proof.**

There are two main concerns with Dwork's proof of the Gaussian mechanism:

- The proof is hard to interpret within the $(\varepsilon, \delta)$-DP notion;
- The proof is based on a problematic Gaussian tail bound.

Specifically, (i) the proof is based on the assumption that the privacy loss can be probabilistically bounded by $1 - \delta$, which is consistent with the PAC privacy notion but hard to interpret in the original $(\varepsilon, \delta)$ privacy notion. The $(\varepsilon, \delta)$ privacy notion says that the distance between two distributions is *always* bounded by the given $(\varepsilon, \delta)$ pair, i.e. strictly cannot be unbounded. From this notion, it is hard to interpret the probabilistic bound of $1 - \delta$ in Dwork's proof. (ii) The proof is based on a tail bound

$$\Pr[Z \geq t] \leq \frac{\sigma}{\sqrt{2\pi}} \cdot \exp\left(\frac{-t^2}{2\sigma^2}\right),$$

where $Z \sim \mathcal{N}(0, \sigma^2)$. This bound is problematic. As a counterexample, if $t = 0$ and $\sigma = 1$, the bound gives $\Pr[Z \geq 0] \leq 1/\sqrt{2\pi} \approx 0.399$, which is incorrect since $\Pr[Z \geq 0] = 0.5$ for Gaussian distributions with mean 0.

**Comparison with the Analytic Gaussian.**

Another Gaussian mechanism is the analytic Gaussian mechanism [11]. The name originates from the analytic form of the Gaussian distribution in its result. We don't adopt this mechanism for two reasons: (i) It lacks analytical form of noise scale $\sigma$ due to the complexity of the given formula to satisfy their privacy definition. i.e. Theorem 8 in [11]. As stated in their paper: ”we propose to find $\sigma$ using a numerical algorithm... (p4, line 7)”, which complicates analytical utility quantification. (ii) It is based on an alternative privacy definition:

$$\Pr\left[\mathcal{L}_{\mathcal{M},x_1,x_2} \geq \varepsilon\right] - e^{\varepsilon} \cdot \Pr\left[\mathcal{L}_{\mathcal{M},x_1,x_2} \leq -\varepsilon\right] \leq \delta.$$

While they proved this definition satisfies $(\epsilon, \delta)$-DP, it is hard to interpret, because it bounds the *difference* between the probability of the privacy loss being too large ($\mathcal{L} \geq \epsilon$) and too small ($\mathcal{L} \leq \epsilon$), different as Dwork's bounding $\Pr[\mathcal{L} \geq \epsilon] \leq \delta$, which is more direct and interpretable.

### E.1.3 Proof of Theorem 16

*Proof.* The combination of $d$ independent $(\varepsilon, \delta)$-PAC LDP mechanisms satisfies pure $\varepsilon$-LDP only if all $d$ mechanisms satisfy pure $\varepsilon$-LDP. Otherwise, the failure probability is at least $\delta$. Therefore, the probability of no failure is at least $(1-\delta)^d$. Thus, the total failure probability of $d$ independent $(\varepsilon, \delta)$-PAC LDP mechanisms is $1-(1-\delta)^d$.

The combination result for $\varepsilon$ is straightforward, and it is the same as the combination of pure $\varepsilon$-LDP mechanisms. □

*Remark.* Compare with the original combination theorem for $d$ mechanisms, i.e. $(d\varepsilon, d\delta)$-LDP [48], the new result is guaranteed to be more precise, as $1-(1-\delta)^d \leq d\delta$ for any $\delta \in (0,1)$ and $d \geq 1$.

## E.2 Complementary Materials

### E.2.1 Deterministic Robustness of White-box Classifiers (Section 7.3.2)

When the classifier is a (public) white-box model, we can directly analyze its exact robustness radius from the model parameters. According to their representativeness, classifiers can be categorized into two types: closed-form classifiers and non-closed-form classifiers.

#### Closed-form Classifiers

A closed-form classifier has a mathematical expression that can be directly analyzed. Examples include the naive Bayes classifier [162], QDA [9, 63], certain variants of SVM [71], and clustering models like $k$-means [164] and Gaussian mixture models [70]. Closed-form expression means excellent interpretability, including robustness analysis. For these classifiers, decision boundaries can be analytically derived from the model parameters, allowing for analytical expression of robustness radius $\theta$ for any input variable $x_0$.

Formally, if the decision boundary of $h$ is a closed-form curve (or surface) $C(x) = 0$, then $\theta$ at an input variable $x_0$ is the distance from $x_0$ to $C(x)$. Its analytical expression w.r.t variable $x_0$ can be calculated by minimizing the distance between $x_0$ and $C(x)$.

**Example 14.** *Consider a 2D QDA classifier $h_{1,2}(x) : (x_u, x_v) \rightarrow \{1, 2\}$, where the discriminant function for the first class is $h_1(x) = -0.5(x_u^2 + x_v^2) + \ln(0.5)$, and for the second class is $h_2(x) = -0.5(0.5(x_u - 1)^2 + 0.5(x_v - 1)^2) - 0.5\ln(4) + \ln(0.5)$. The decision boundary $h_1(x) - h_2(x) = 0$ is a circle with radius $2\sqrt{\ln 2 + 1}$ centered at $(-1, -1)$. Thus, the $\ell_2$ robustness radius at $x_0 = (x_u, x_v)$ is $\theta = |2\sqrt{\ln 2 + 1} - \sqrt{(x_u + 1)^2 + (x_v + 1)^2}|$.*[*]

## Non-closed-form Classifiers

Non-closed-form classifiers often lack explicit mathematical expressions for their decision boundaries. Examples include neural network [163] and decision tree [22]. Their decision boundaries are analytically intractable, making it challenging to derive the robustness radius $\theta$ directly. Finding $\theta$ for these classifiers is known as the robustness verification problem [82, 112], i.e. verify whether $h(B_\theta(x)) = h(x)$ for a given $\theta$ at $x$. The main insight is to encode classifiers as a set of constraints, transforming the problem of verifying $h(B_\theta(x)) = h(x)$ into a satisfiability problem that can be solved by constraints solvers.

Formally, given a concrete sample $x_0$ and $\theta$, we have $B_\theta(x_0) = \{x : ||x - x_0||_\infty \leq \theta\}$ as a linear constraint on $x$. Denote $h_i(x)$ as the score function of the $i$-th class of $h(x)$, and let $c$ be the correct class of $x_0$. By encoding $B_\theta(x_0)$ and the classifier $h(x)$ into a satisfiability problem

$$x \in B_\theta(x_0) \ \wedge \ h_c(x) < h_i(x) \text{ for } i \neq c,$$

we can verify the problem using a constraint solver. If it is unsatisfiable, then $h(x)$ outputs the correct class within $B_\theta(x_0)$. The maximum $\theta$ that makes the problem unsatisfiable is the robustness radius $\theta$.

**Example 15.** *(To replicate the QDA classifier from the previous example, which is also the same classifier depicted in Figure 7.2.) Consider a 2D neural network classifier $h_{1,2}(x) : [0, 1]^2 \rightarrow \{1, 2\}$ with structure $a_2(\mathrm{ReLU}(a_1(x)))$, where $\mathrm{ReLU} = \max(0, x)$ and $a_1, a_2$ are affine functions of the*

[*]A trivial lower bound of the $\ell_\infty$ robustness radius can be obtained by using $\theta/\sqrt{2}$, based on the norm inequality.

*hidden layer and output layer that*

$$a_1(x) = \begin{bmatrix} -1.86 & -2.09 \\ 0.12 & -0.46 \end{bmatrix} x + \begin{bmatrix} 3.71 \\ -0.08 \end{bmatrix},$$
$$a_2(x) = \begin{bmatrix} -3.05 & 0.40 \\ 4.02 & -0.22 \end{bmatrix} x + \begin{bmatrix} 0.94 \\ -0.58 \end{bmatrix}.$$

*Denote $h_1(x)$ and $h_2(x)$ as the 1st and 2nd dimension of $h_{1,2}(x)$. Given $x_0 = (0.5, 0.5)$ and $\theta = 0.1$, we have $B_{0.1}(x_0) = ||x - x_0||_\infty \leq 0.1$ as a linear constraint on $x$. If the correct class is $h_{1,2}(x_0) = 1$, the robustness verification problem is*

$$x \in B_{0.1}(x_0) \ \wedge \ h_1(x) < h_2(x).$$

*If this problem is verified as unsatisfiable, it means $h_{1,2}(x)$ always outputs class $1$ within $B_{0.1}(x_0)$. Finally, finding $\arg\max_\theta h(B_\theta(x)) = h(x)$ is a sequence of decision problems on $\theta$.*

### E.2.2 Explanation of Definition 17

There are two randomness sources in Definition 17: (i) Denote the deterministic robustness $h(\tilde{x}) = h(x)$ an indicator $\phi(\tilde{x}) \in \{0, 1\}$. We can say a system is robust under random $\tilde{x}$ with $\Pr[\phi(\tilde{x}) = 1] > 1 - \omega$. In this setting, the robustness boundary is assumed to be known, and the randomness comes from $\tilde{x}$. (ii) When the robustness boundary is unknown, $\phi(\tilde{x})$ must be estimated and thus takes values in $[0, 1]$ instead of $\{0, 1\}$. In this case, the condition $\phi(\tilde{x}) > 1 - \tau$ becomes a random event, where the randomness comes from the estimation of $\phi$. This leads to two-level probabilistic guarantee $\Pr[1 - \Pr[\phi(\tilde{x}) \leq \tau]] \geq 1 - \omega$. Existing definitions [144, 180] use this two-level probability but lack interpretations.

### E.2.3 LDP Mechanisms in Figure 7.3

We provide concrete instantiations of the LDP mechanisms in Figure 7.3 and discuss their $\rho(\varepsilon, \theta)$ curves.

#### Piecewise-based Mechanism

The original PM mechanism [150] is defined on $[-1, 1] \to [-C_\varepsilon, C_\varepsilon]$, where $C_\varepsilon$ is a variable dependent on $\varepsilon$. Such design ensures unbiasedness, as it was originally designed for mean estimation. The original SW mechanism [95] is defined on $[0, 1] \to [-b_\varepsilon, 1 + b_\varepsilon]$. Unlike PM, SW is biased and designed for distribution estimation.

Although defined on different domains, both PM and SW mechanisms can be "normalized" to the same domain $[0,1] \to [0,1]$ with the same privacy parameter $\varepsilon$. We give their formulation as the following two definitions.

**Definition 24** (The PM mechanism). *The PM mechanism takes an input $x \in [0,1]$ and outputs a random variable $\tilde{x} \in [0,1]$ as follows:*

$$\mathrm{pdf}[\mathcal{M}(x) = \tilde{x}] = \begin{cases} p_\varepsilon & \text{if } \tilde{x} \in [l_{x,\varepsilon}, r_{x,\varepsilon}], \\ p_\varepsilon / e^\varepsilon & \tilde{x} \in [0,1] \setminus [l_{x,\varepsilon}, r_{x,\varepsilon}], \end{cases}$$

*where $p_\varepsilon = e^{\varepsilon/2}$,*

$$[l_{x,\varepsilon}, r_{x,\varepsilon}] = \begin{cases} [0, 2C) & \text{if } x \in [0, C), \\ x + [-C, C] & \text{if } x \in [C, 1-C], \\ (1-2C, 1] & \text{otherwise}, \end{cases}$$

*with $C = (e^{\varepsilon/2} - 1)/(2e^\varepsilon - 2)$.*

The SW mechanism is similar but uses different instantiations for $p_\varepsilon$ and $[l_{x,\varepsilon}, r_{x,\varepsilon})$.

**Definition 25** (The SW mechanism). *The SW mechanism takes an input $x \in [0,1]$ and outputs a random variable $\tilde{x} \in [0,1]$ as the same formulation as the PM mechanism, but with*

$$p_\varepsilon = \frac{e^\varepsilon - 1}{\varepsilon}, \quad C = \frac{e^\varepsilon(\varepsilon - 1) + 1}{2(e^\varepsilon - 1)^2}.$$

Both mechanisms guarantee $\varepsilon$-LDP through their piecewise design, as the probability of seeing the same value $\tilde{x}$ when the original data is $x_1$ and $x_2$ is bounded by $e^\varepsilon$.

**$\boldsymbol{\rho(\varepsilon, \theta)}$ curves.** The concentration analysis $F(x+\theta) - F(x-\theta)$ of the PM and SW mechanisms results in a linear curve with two segments. When $\theta$ is small, the concentration is dominated by the $[l_{x,\varepsilon}, r_{x,\varepsilon})$ segment, which gives a slope proportional to $p_\varepsilon$. When $\theta$ is large, the concentration is dominated by the $[0,1) \setminus [l_{x,\varepsilon}, r_{x,\varepsilon})$ segment, which gives a slope proportional to $p_\varepsilon / e^\varepsilon$. Thus, the $\rho(\varepsilon, \theta)$ curves for the PM and SW mechanisms exhibit a linear pattern with two distinct slopes.

### Discrete Mechanism

By discretizing the $[0,1]$ domain into bins, LDP mechanisms defined on discrete domains can also be applied. In Figure 7.3, we fixed the number of bins to 100, i.e. the domain is $\mathcal{X} := \{0, 0.01, 0.02, \ldots, 1\}$. The according Exponential mechanism and $k$-RR mechanism is defined as follows.

**Definition 26** (The Exponential mechanism)**.** *Given score function $d(x, \tilde{x})$, the Exponential mechanism defined on $\mathcal{X}$ takes an input $x \in \mathcal{X}$ and outputs a random variable $\tilde{x} \in \mathcal{X}$ as follows:*

$$\Pr[\mathcal{M}_{\exp}(x) = \tilde{x}] = \frac{\exp\left(\frac{\varepsilon d(x,\tilde{x})}{2\Delta d}\right)}{\sum_{y' \in \mathcal{Y}} \exp\left(\frac{\varepsilon d(x,\tilde{x}')}{2\Delta d}\right)},$$

*where $\Delta d = \max_{x,\tilde{x},\tilde{x}' \in \mathcal{X}} |d(x, \tilde{x}) - d(x, \tilde{x}')|$ is the sensitivity of the score function $d$.*

Here we choose the score function as the negative absolute distance, i.e. $d(x, \tilde{x}) := -|x - \tilde{x}|$, so that outputs closer to $x$ are assigned higher probabilities. Accordingly, the sensitivity is $\Delta d = 1$.

**Definition 27** (The $k$-RR mechanism)**.** *$k$-RR is a sampling mechanism $\mathcal{M} : \mathcal{X} \to \mathcal{X}$ that, given input $x \in \mathcal{X}$, outputs $\tilde{x} \in \mathcal{X}$ according to*

$$\Pr[\mathcal{M}(x) = \tilde{x}] = \begin{cases} \dfrac{e^{\varepsilon}}{|\mathcal{X}| - 1 + e^{\varepsilon}} & \text{if } \tilde{x} = x, \\ \dfrac{1}{|\mathcal{X}| - 1 + e^{\varepsilon}} & \text{otherwise.} \end{cases}$$

**$\boldsymbol{\rho(\varepsilon, \theta)}$ curves.** The concentration analysis $F(x + \theta) - F(x - \theta)$ of the Exponential mechanism results in a smooth curve, as the probability of seeing value $\tilde{x}$ further away from $x$ decreases smoothly w.r.t $\theta$. In contrast, the concentration analysis of the $k$-RR mechanism exhibits a pattern similar to the PM and SW mechanisms. When $\theta$ is small, the probability is dominated by the bin containing $x$, results in a slope $\propto e^{\varepsilon}/(|\mathcal{X} - 1 + e^{\varepsilon}|)$. When $\theta$ becomes larger, the probability is dominated by the other bins, which gives a slope $\propto 1/(|\mathcal{X} - 1 + e^{\varepsilon}|)$.

### E.2.4 PAC LDP vs $(\varepsilon, \delta)$-LDP (Section 7.4.2)

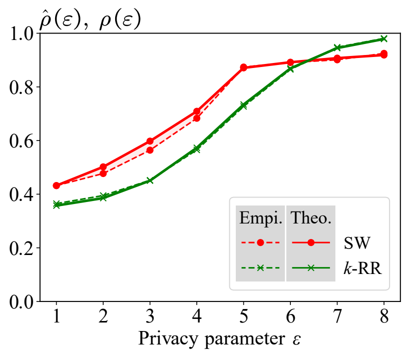
This section discusses the difference between PAC LDP, $(\varepsilon, \delta)$-LDP, and other related privacy notions.

**Definition 28** ($(\varepsilon, \delta)$-LDP [48])**.** *A randomized mechanism $\mathcal{M}$ satisfies $(\varepsilon, \delta)$-LDP if for all $x_1, x_2 \in \mathcal{X}$ and all $S \subseteq \mathcal{Y}$,*

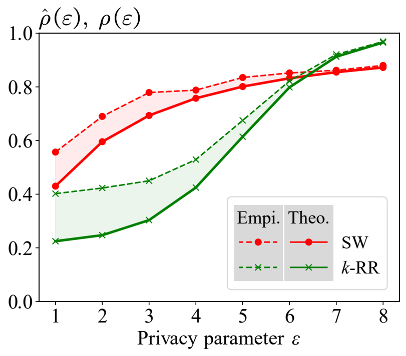
$$\Pr[\mathcal{M}(x_1) \in S] \le e^{\varepsilon} \cdot \Pr[\mathcal{M}(x_2) \in S] + \delta.$$

This definition says that the distance between the distributions of $\mathcal{M}(x_1)$ and $\mathcal{M}(x_2)$ is bounded by a $(\varepsilon, \delta)$ pair. More specifically, it can be rewritten as:

$$\frac{\Pr[\mathcal{M}(x_1) \in S]}{\Pr[\mathcal{M}(x_2) \in S] + \delta/e^{\varepsilon}} \le e^{\varepsilon}.$$



(a) Logistic Regression.



(b) Random Forest.

Figure E.1: Empirical and theoretical utility for two classifiers trained on the Stroke Prediction dataset.

Given concrete values of pair $(\varepsilon, \delta)$,[†] the term $\delta/e^{\varepsilon}$ is a constant. Thus, it means that the distance between the distributions of $\mathcal{M}(x_1)$ and $\mathcal{M}(x_2)$ must be strictly bounded, i.e. similar to pure $\varepsilon$-LDP. However, this definition is often interpreted as: it is $\varepsilon$-LDP with a failure probability of $\delta$. Meanwhile, the Gaussian mechanism relies on this interpretation.

Other relaxed privacy notions, including the Concentrated privacy from Dwork [44], Renyi privacy [114], and PAC privacy [167], provide patches to $(\varepsilon, \delta)$-LDP. The Gaussian mechanism has natural results under the Concentrated privacy and Renyi privacy notions, while we provide a PAC privacy result in Appendix E.1.2. Among these privacy notions, $(\varepsilon, \delta)$-PAC LDP provides the same meaning as the common interpretation of $(\varepsilon, \delta)$-LDP, i.e. it is $\varepsilon$-LDP with a failure probability of $\delta$.

### E.2.5 More Case Studies for the Stroke Prediction Dataset (Section 7.6.2)

This section provides a detailed utility analysis of the two classifiers trained on the Stroke Prediction dataset.

#### Other Records and Mechanisms

We also focus on the first record in the dataset, i.e. the record "Age: 67, BMI: 36.6, Hypertension: 0, . . . ", to evaluate the performance of other two LDP mechanisms: the SW mechanism and the $k$-RR mechanism.

[†]In fact, $\delta$ can be defined in a more intricate manner, such as incorporating the noise scale $\sigma$ as in [20]. However, this approach introduces a recursive dependency thus uncontrollable in practical applications, as $\sigma$ itself should be defined in terms of $\delta$.

Figure E.1 shows the theoretical and empirical utility of the two classifiers under the SW and $k$-RR mechanisms. The theoretical utility of both mechanisms is almost the same as the empirical utility for the Logistic Regression classifier. For the Random Forest classifier, the theoretical utility is a lower bound for the empirical utility, especially when $\varepsilon$ is small. Meanwhile, we can see that the SW mechanism does not always outperform the $k$-RR mechanism. When $\varepsilon$ exceeds 6, $k$-RR outperforms SW in both classifiers.

## Closed Form CDF of the PM Mechanism

This section provides the closed form expression for $\rho(\varepsilon, \theta)$ when using the PM mechanism.

The input "Age: 79" corresponds to the normalized value $x = 0.79$ in domain $[0, 1]$. Then according to Definition 24 of the PM mechanism, we have

$$F_{\mathrm{PM}_{\varepsilon,\mathrm{Age}}}(0.63) = \int_0^{0.63} pdf[\mathcal{M}(0.79) = \tilde{x}]\mathrm{d}\tilde{x}.$$

For piecewise-based mechanisms, we need to consider the range of $[l_{0.79,\varepsilon}, r_{0.79,\varepsilon}]$ relative to 0.63. For $\varepsilon$ that makes $0.63 \notin [l_{0.79,\varepsilon}, r_{0.79,\varepsilon}]$, the above integral is

$$F_{\mathrm{PM}_{\varepsilon,\mathrm{Age}}}(0.63) = 0.63 \cdot e^{-\varepsilon/2}.$$

For $\varepsilon$ that makes $0.63 \in [l_{0.79,\varepsilon}, r_{0.79,\varepsilon}]$, the above integral is

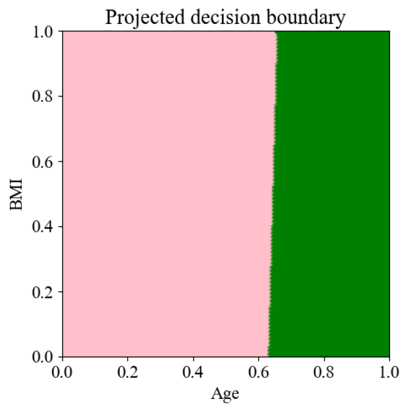
$$F_{\mathrm{PM}_{\varepsilon,\mathrm{Age}}}(0.63) = (0.79 - C) \cdot e^{-\varepsilon/2} + (C - 0.16) \cdot e^{\varepsilon/2},$$

where $C = (e^{\varepsilon/2} - 1)/(2e^{\varepsilon} - 2)$, as define in Definition 24.

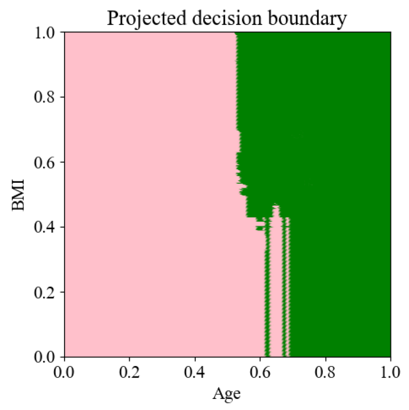
## Projected Decision Boundary

The gap between the theoretical and empirical utility for the Random Forest classifier comes from the complexity of its decision boundary.

Figure E.2 illustrates the projected decision boundaries of the two classifiers on the "Age" and "BMI" features. We can see that the decision boundary of the Random Forest classifier is significantly more complex than that of the Logistic Regression classifier, i.e. a combination of many hyperrectangles, making it hard to be approximated by one hyperrectangle. Consequently, the theoretical utility for the Random Forest classifier is conservative compared to its actual utility. In contrast, the simpler decision boundary of the Logistic Regression classifier can be well approximated by hyperrectangles, leading to a precise theoretical utility.



(a) Logistic Regression (LR).



(b) Random Forest (RF).

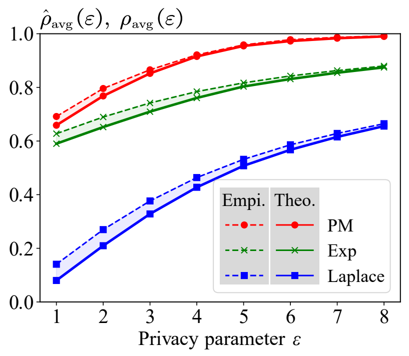
Figure E.2: Projected decision boundary of the two classifiers on the stroke dataset.

### Average-case and Worst-case Utility

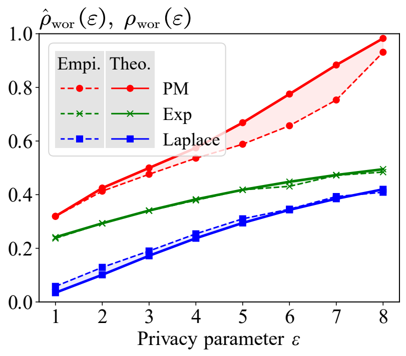
Figure E.3 shows the average-case and worst-case utility of the Logistic Regression classifier under three LDP mechanisms. For the average-case utility, the trends closely mirror those observed in the point-wise utility quantification presented in the main text. For the worst-case utility, the curves are less regular but still exhibit the same overall ordering among mechanisms: the PM mechanism consistently achieves the highest utility, followed by the Exponential mechanism, and then the Laplace mechanism. In the worst-case scenario, we observe that the theoretical utility is not always lower than the empirical utility (though they are close), especially for the PM mechanism when $\varepsilon$ is large. This occurs because the worst-case robustness hyperrectangle is typically two-dimensional and small, which amplifies the impact of sampling errors in empirical evaluation. In such cases, the actual utility is often higher than the empirical utility.

Figure E.4 shows the average-case and worst-case utility of the Random Forest classifier. The worst-case utility for the Random Forest classifier is significantly smaller than that of the Logistic Regression classifier. This is due to the complex decision boundary of the Random Forest classifier, as shown in Figure E.2b. Around Age, BMI $\approx \{0.65, 0.3\}$, the robustness hyperrectangle is significantly smaller than that of the Logistic Regression classifier, leading to both small theoretical and empirical utilities.

**Conclusion on Robustness.** From the results of average-case and worst-case utility analysis of the Logistic Regression and Random Forest classifiers, we can conclude that the Logistic Regression classifier is more robust under LDP perturbations than the Random Forest classifier.

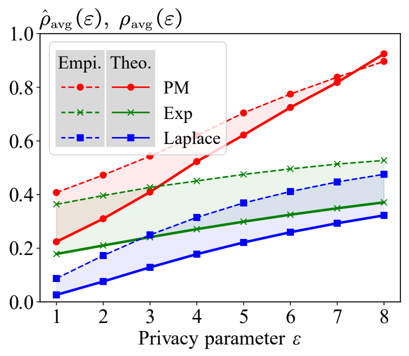


(a) LR: Average-case utility.

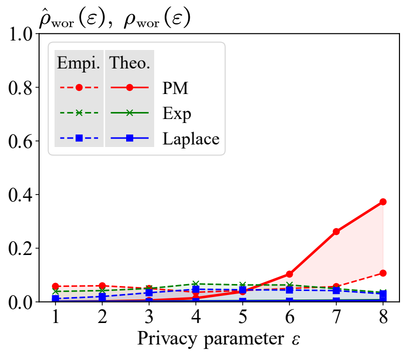


(b) LR: Worst-case utility.

Figure E.3: Average-case and worst-case utility for the Logistic Regression classifier trained on the Stroke Prediction dataset.



(a) RF: Average-case utility.



(b) RF: Worst-case utility.

Figure E.4: Average-case and worst-case utility for the Random Forest classifier trained on the Stroke Prediction dataset.

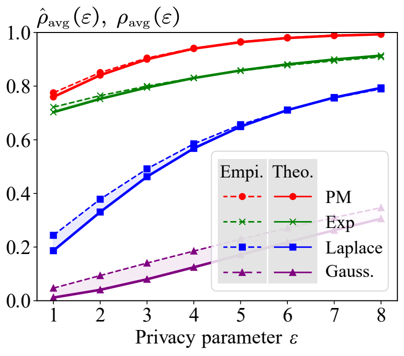
### E.2.6 More Case Studies for the Bank Customer Attrition Dataset (Section 7.6.2)

#### Detailed Quantification

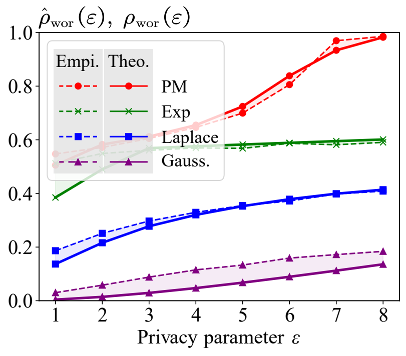
We take the Logistic Regression classifier as an example to show the detailed utility quantification under the PAC LDP.

The robustness hyperrectangle at this record is $\theta_\diamond = [0, 0.72] \times [0, 1]$. Using this information, we can provide the quantification of the utility at the record under the PM mechanism.

*For this trained Logistic Regression classifier on the Bank Customer Attrition dataset, at the record "Age: 22, Estimated Salary: 101 348, Credit Score: 619, ...", with probability at least $\rho(\varepsilon, \theta_\diamond)$, the classifier preserves the correct prediction under the privacy indicator $\mathcal{I}_{0.1}$ and the PM mechanism*



(a) LR: Average-case utility.



(b) LR: Worst-case utility.

Figure E.5: Average-case and worst-case utility for the Logistic Regression classifier trained on the Bank Customer Attrition dataset.

*applied to "Age" and "Estimated Salary" (i.e.* $(2\varepsilon, 0.1)$*- PAC LDP), where*

$$\begin{aligned}\rho(\varepsilon,\theta) =&[F_{\text{PM}_{\varepsilon,\text{Age}}}(0.72) - F_{\text{PM}_{\varepsilon,\text{Age}}}(0)][F_{\text{PM}_{\varepsilon,\text{Sal}}}(1) - F_{\text{PM}_{\varepsilon,\text{Sal}}}(0)]\\ =&F_{\text{PM}_{\varepsilon,\text{Age}}}(0.72).\end{aligned}$$

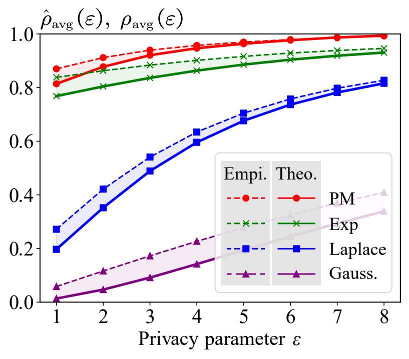
For the Gaussian mechanism, the utility quantification is similar but uses the Gaussian CDF.

*For this trained Logistic Regression classifier on the Bank Customer Attrition dataset, at the record "Age: 22, Estimated Salary: 101 348, Credit Score: 619, ...", with probability at least* $\rho(\varepsilon, \theta_\diamond)$*, the classifier preserves the correct prediction under the Gaussian mechanism (Theorem 18) applied to "Age" and "Estimated Salary" (i.e.* $(2\varepsilon, 0.1)$*- PAC LDP), where*

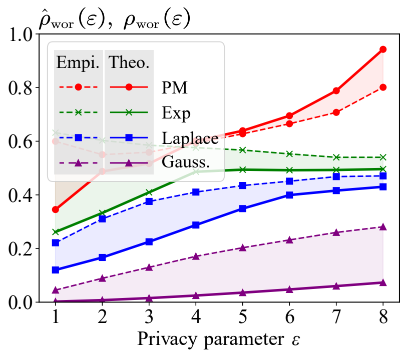
$$\begin{aligned}\rho(\varepsilon,\theta) =&[F_{\text{Gau}_{\varepsilon,\text{Age}}}(0.72) - F_{\text{Gau}_{\varepsilon,\text{Age}}}(0)][F_{\text{Gau}_{\varepsilon,\text{Sal}}}(1) - F_{\text{Gau}_{\varepsilon,\text{Sal}}}(0)]\\ =&F_{\text{Gau}_{\varepsilon,\text{Age}}}(0.72) - F_{\text{Gau}_{\varepsilon,\text{Age}}}(0).\end{aligned}$$

### Average-case and Worst-case Utility

Figure E.5 and Figure E.6 summarize the average-case and worst-case utility for classifiers trained on the Bank Customer Attrition dataset. The observed trends are consistent with those from the Stroke Prediction dataset: (i) The theoretical utility closely matches the empirical utility, especially in the average-case scenario. (ii) The PM mechanism consistently achieves the highest utility, followed by the Exponential mechanism, then the Laplace mechanism, and finally the Gaussian mechanism.



(a) RF: Average-case utility.



(b) RF: Worst-case utility.

Figure E.6: Average-case and worst-case utility for the Random Forest classifier trained on the Bank Customer Attrition dataset.

### E.2.7 More Details for the MNIST Classifier (Section 7.6.2)

#### Monte Carlo Estimation of the Utility

We emphasize the conceptual connection between LDP and robustness while adopting robustness notions that largely align with the standard *local robustness* used in prior work. Although these notions are clean and well established, they can yield conservative utility estimates in high-dimensional settings.

To obtain tighter utility quantification in high dimensions, one can consider more flexible robustness notions, e.g. robustness regions given by irregular sets rather than axis-aligned hyperrectangles (as induced by complex decision boundaries). Such regions can be estimated via Monte Carlo sampling in high-dimensional spaces. This estimation is performed once; afterward, for any given $\varepsilon$, the utility can be theoretically quantified by approximately integrating the $d$-dimensional LDP mechanism's probability distribution over the estimated robustness region.

Formally, let $h : [0,1]^d \to \{1,\dots,K\}$ be a $d$-dimensional classifier and let $x \in [0,1]^d$ be an input with predicted (and correct) label $h(x)$. Draw $\tilde{x}_1,\dots,\tilde{x}_N$ i.i.d. uniformly from $[0,1]^d$, and for each sample check whether $h(\tilde{x}_i) = h(x)$ holds. Assume that $m$ of these samples satisfy the check, and let $\mathcal{S} \subseteq [0,1]^d$ denote an estimated robustness region that contains the accepted samples. Under a

$d$-dimensional (continuous) LDP mechanism,[‡] the utility can then be quantified as

$$\begin{aligned}\rho(\varepsilon, \mathcal{S}) = &\int_{\mathcal{S}} pdf[\mathcal{M}_\varepsilon(x) = \tilde{x}]\mathrm{d}\tilde{x} \\ \approx &\mathrm{Vol}(\mathcal{S}) \cdot \frac{1}{m}\sum_{i=1}^{m} pdf[\mathcal{M}_\varepsilon(x) = \tilde{x}_i],\end{aligned}$$

where $\mathrm{Vol}(\mathcal{S})$ is the volume of the region $\mathcal{S}$, which can be estimated by $\mathrm{Vol}([0,1]^d) \cdot m/N$. It is still a theoretical measure, since samples are fixed and $pdf[\mathcal{M}_\varepsilon(x) = \tilde{x}_i]$ is determined analytically by the LDP mechanism and can be evaluated for any $\varepsilon$.

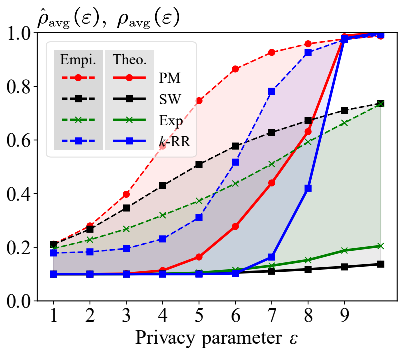
This approach follows the connection between concentration analysis and robustness analysis established in this paper, with the concentration analysis now performed over the estimated robustness region $\mathcal{S}$ rather than the hyperrectangle $\theta_\diamond$. The error is now governed by the Monte Carlo estimation error (i.e. $\mathcal{O}(1/\sqrt{N})$) and volume estimation error, instead of Hoeffding bound in Theorem 15.

We experimented with this approach on the MNIST-7×7 dataset. While it is theoretically sound, in such a high-dimensional space the samples that pass the uniform-sampling check are sparse. As a result, for most accepted samples $\tilde{x}_i$, the density $pdf[\mathcal{M}_\varepsilon(x) = \tilde{x}_i]$ is close to zero, which drives the estimated utility to an unrealistically small value. Moreover, we cannot directly apply importance sampling using $pdf[\mathcal{M}_\varepsilon(x)]$ as the proposal distribution, since this proposal distribution depends on $\varepsilon$ and would essentially reduce the procedure to empirical utility estimation.

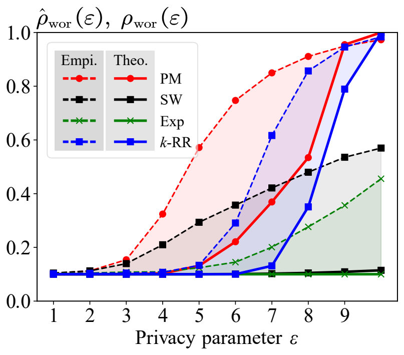
**Average-case and Worst-case Utility**

We use the first 50 images from the MNIST-7×7 dataset as representative samples for analyzing the average-case and worst-case utility under LDP-perturbed inputs. Figure E.7 presents the results of this analysis. Both the average-case and worst-case utility show a similar trend to the point-wise utility quantification, with the PM mechanism consistently achieving the highest utility.

[‡]For discrete LDP mechanisms, the utility can be computed directly by summing the probabilities that the mechanism outputs samples in $\mathcal{S}$, eliminating the need for volume estimation. We omit this discussion for simplicity.



(a) NN: Average-case utility.



(b) NN: Worst-case utility.

Figure E.7: Average-case and worst-case utility for the Neural Network classifier trained on the MNIST-7×7 dataset.