PRACTICE-INSPIRED TRUST MODELS AND MECHANISMS FOR DIFFERENTIAL PRIVACY

by

Brendan Avent

A Dissertation Presented to the
FACULTY OF THE USC GRADUATE SCHOOL
UNIVERSITY OF SOUTHERN CALIFORNIA
In Partial Fulfillment of the
Requirements for the Degree
DOCTOR OF PHILOSOPHY
(COMPUTER SCIENCE)

August 2023

Acknowledgements

I could not have comprehended the path that I would eventually take through graduate school, and I attribute the successful completion of my Ph.D. to the support of those around me.

First and foremost, I would like to thank my parents, Mitchell Avent and Loretta Rubenstein, for your unwavering love and support throughout my many years of schooling.

I thank my advisor, Aleksandra Korolova, for your guidance and patience throughout these years. Your devotion to finding the most meaningful and impactful problems, and then precisely and rigorously formalizing them, has completely reshaped how I think about research.

I also thank my qualifying exam and thesis committee members Salman Avestimehr, Leana Golubchik, David Kempe, and Cyrus Shahabi for listening to my many presentations, reading several of my papers and this dissertation, and providing helpful feedback at each juncture. I am especially grateful to David for his detailed comments on the final chapter that not only dramatically improved its presentation, but improved its contents as well.

From my brief summer internships, I would like to thank my collaborators and mentors there. From my time at Microsoft, I thank Ben Livshits, and from my time at Amazon, I thank Borja Balle, Tom Diethe, Javier González, and Andrei Paleyes.

For my close friends and loved ones: there are far too many of you to name, and there is no way I could justify naming some of you while leaving others out. Instead, I would like to say that I am personally indebted to all of you for your support and companionship over all these years, and that I could not have done it without each and every one of you.

I am grateful to have received support for my work directly and indirectly from the following sources: the National Science Foundation from grants #1755992, #1916153, #1943584, and #1956435, Meta from their Privacy Enhancing Technologies Award, Amazon Research Cambridge, a VMWare fellowship, and a gift from Google. Additionally, part of my work was done at UC Berkeley's Simons Institute during their Spring 2019 "Data Privacy: Foundations and Applications" semester, which I am grateful to have been a part of.

Table of Contents

List of Tables

List of Figures

Abstract

Now more than ever, organizations such as companies, governments, and researchers collect and analyze people's personal data to drive decisions and fuel innovation. Differential privacy has become the gold standard for protecting privacy in computer science, particularly for privacy-preserving data analyses and machine learning. Differential privacy is a mathematical definition of privacy that provides quantifiable protections for inferences that can be made when contributing one's data to an analysis by adversaries with access to arbitrary auxiliary information. Significant research effort has been devoted to designing and analyzing mechanisms that satisfy differential privacy. However, far less research to date has studied the pragmatic considerations of differential privacy, i.e., how its trust models and mechanisms can be adapted and applied for real-world uses.

In this thesis, we focus on making differential privacy useful for real-world applications by removing barriers that hinder its adoption in practice. In the first part of the thesis, we address the utility gap between the more and less desirable trust models of differential privacy. Towards this, we define a new "hybrid" differential privacy trust model and design and analyze high-utility mechanisms for multiple applications within it. In the second part of the thesis, we address the lack of tools for analyzing the utility of complex differentially private mechanisms. We do so by developing a new method for quantifying the privacy–utility trade-off of complex, hyperparameterized, differentially private mechanisms. Assessing our new method across a multitude of private machine learning tasks, we find that it is highly effective at quantifying such mechanisms' privacy–utility trade-offs. In the third and final part of the thesis, we address the open question of

how to improve the utility of large-scale query-answering differentially private mechanisms. We extend the state-of-the-art differentially private mechanism for this problem and, in two different settings, find that it can efficiently and effectively answer a massive number of queries.

Chapter 1

Introduction

Data is the new oil. Like oil, data is valuable, but if unrefined it cannot really be used.

- Clive Humby, 2006

Mathematician Clive Humby's famous quote has proven increasingly accurate in the years that followed it. Today's internet is entirely powered by the data of its users, and in turn, influences nearly every facet of our lives. More than that, everyday decisions by businesses, governments, and researchers are driven by data. Just as there are risks involved in drilling, refining, and using oil, so too are there risks involved in collecting, processing, and using people's data. The risk we focus on in this thesis is that of *privacy leakage*, which is the ability for an unwanted party to learn private information about an individual based on their data. Privacy leakage can be trivially eliminated by simply never collecting or using one's data — however, such an approach would mean that many important research studies could not be performed, useful data-driven algorithms could not be used, and critical government and business analyses could not be conducted. Thus, akin to never refining oil, never using one's data renders it worthless.

To preserve individuals' privacy while maintaining the usefulness of their data, the classic approach was to anonymize the data using techniques such as data redaction, data swapping [DR82], k-anonymity [SS98], l-diversity [Mac+07], and others. However, such ad hoc data anonymization techniques do not provide strong and mathematically rigorous guarantees against privacy

leakage and are susceptible to *linkage* or *background knowledge* attacks. These are attacks where an adversary links some auxiliary information (which the anonymization procedure could not account for) with the anonymized data to partially or wholly deanonymize it. Real-world high-profile privacy leaks from anonymized data sources have demonstrated that data anonymization is insufficient for providing individuals with even a moderate level of privacy protection. These privacy leaks included medical records for Massachusetts state employees [Swe97], search histories of AOL users [BZH06], movie ratings of Netflix viewers [NS08], genome sequences in GenBank [Gym+13], and US Census responses [Bur22; Dic+22]. These examples, along with many others, motivate the need for a formal definition of privacy that enables strong and mathematically rigorous guarantees against privacy leakage, even in the face of adversaries who can acquire arbitrary auxiliary information.

1.1 Differential Privacy

Grounded in part by the motivation to overcome the problems posed by the potential availability of auxiliary information, the definition of *differential privacy* [Dwo+06b] (DP) was introduced in 2006. Differential privacy overcomes these problems by turning the focus away from judging whether the data itself is anonymized and, instead, toward the mechanism (i.e., algorithm) that processes the data in order to avoid privacy leakage. Concretely, differential privacy introduces a constraint on the amount of information that a mechanism's output can reveal about any individual whose data was used in its input compared to what the mechanism's output can reveal when that individual's data is not used. Informally, for a mechanism to satisfy the DP constraint, the mechanism's output distribution must be approximately the same when any individual's data used in its input is changed. This is accomplished by carefully incorporating randomness into the mechanism, enabling a certain plausible deniability for anyone whose data might have been used by the mechanism.

The approximate sameness of output distributions is controlled by a parameter ϵ, which serves as a knob that trades off privacy and utility. That is, a smaller value of ϵ implies more similar output distributions, which necessitates more randomness in the mechanism and thus yields a higher level of privacy. On the other hand, a larger value of ϵ implies that output distributions can be more distinct; therefore, less randomness in the mechanism is needed, making its output more useful (i.e., higher utility). The effect of this definition is that for any adversary analyzing the output of a DP mechanism, ϵ controls the confidence with which the adversary can determine any additional information about those whose data was used by the mechanism relative to those whose data was not. Concretely, a larger ϵ allows the adversary to infer more about any individual's data, while a smaller ϵ allows the adversary to infer less. The power of DP primarily stems from the fact that it is purely a property of the mechanism that accesses data, which means its proven guarantees against privacy leakage remain intact regardless of any auxiliary information an adversary may acquire. Therefore, the output of a DP mechanism with a small ϵ will never provide a significant amount of information about any individual's data *even if* an adversary obtains arbitrary auxiliary information, including information about the individual in question or about the other individuals whose data was used.

Due to the power of the differential privacy framework and through a significant undertaking by the research community since its introduction in 2006, DP has become the de facto standard for privacy-preserving data analysis and machine learning in computer science literature. Industry and government entities have followed suit, with a select number of large-scale, real-world deployments of DP by Google [EPK14; Pap19; Akt+20; Bav+20; Bav+21], Apple [Tea17], Snap [Pih+22], Microsoft [DKY17; Kop21], and the U.S. Census Bureau [Daj+17]. Most recent efforts include open-source DP libraries by IBM [Hol+19], OpenDP [GHV20], and Tumult Labs [Ber+22].

In the remainder of this section, we briefly introduce the technical details of differential privacy that are foundational for this thesis[1]. We first state the formal definition of differential privacy, then describe important properties of DP that follow directly from its definition. We then detail select fundamental DP mechanisms, which we use as building blocks for the more advanced mechanisms in this thesis.

1.1.1 Defining Differential Privacy

As previously described, differential privacy limits the impact that any individual's data can have on the output of a mechanism. That is, it is a constraint on a mechanism $\mathcal{M}$ that takes as input a potentially sensitive dataset D and outputs $\mathcal{M}(D)$. Informally, the constraint specifies that the output $\mathcal{M}(D)$ must be approximately indistinguishable from the output of the mechanism $\mathcal{M}(D')$ on any similar dataset D'. Formally, this notion is captured in the definition of DP.

Definition 1.1.1 (Differential Privacy [Dwo+06b])**.** A randomized mechanism $\mathcal{M}$ is (ϵ, δ)-differentially private if and only if for all neighboring input datasets D and D' that differ in precisely one individual's data, the following inequality is satisfied for all possible sets of outputs $Y \subseteq \mathit{Range}(\mathcal{M})$:

$$\Pr[\mathcal{M}(D) \in Y] \leq e^{\epsilon} \Pr[\mathcal{M}(D') \in Y] + \delta.$$

The setting where $\delta = 0$ is referred to as *pure* differential privacy and the mechanism is said to satisfy ϵ-DP. The setting where $\delta > 0$ is referred to as *approximate* differential privacy.

Importantly, the probabilities in this definition's inequality refer only to the mechanism $\mathcal{M}$'s internal randomness, not any probabilistic properties that the dataset may have. Furthermore, for the privacy guarantee to be meaningful, it is assumed that the mechanism does not reveal the concrete realizations of its internal randomness. The non-negative real values ϵ and δ are often referred to as the *privacy level* or *privacy cost*, and a maximum bound on the privacy cost

[1]For a more detailed treatment on this material and other DP topics, refer to Dwork and Roth's textbook on differential privacy [DR+14].

is referred to as the *privacy budget*. Increasing values of either ϵ or δ imply less privacy but may enable DP mechanisms to achieve greater utility. The additive δ term can be interpreted as a bound on the probability that the mechanism fails to satisfy ϵ-DP, potentially resulting in significant privacy leakage for any user whose data is in the dataset D. Because of this, it is common practice to choose δ to be a relatively small value; i.e., $\delta \ll 1/|D|$.

Three fundamental properties of DP mechanisms that we use throughout this thesis follow directly from the DP definition.

- **Post-processing**: When a differentially private mechanism processes data, any further data-independent analysis of the mechanism's output will not weaken the differential privacy guarantee. Intuitively, this means that even with the help of outside data sources, an adversary cannot weaken one's privacy by "thinking hard" about the mechanism's output.

- **Sequential Composition:** One's data can be used repeatedly in multiple differentially private computations. However, the differential privacy guarantee degrades with each such computation. The *basic composition* theorem ([Dwo+06a], Theorem 1) states that if mechanisms $\mathcal{M}_1, \ldots, \mathcal{M}_k$ each satisfy (ϵ_1, δ_1)-DP, $\ldots$, (ϵ_k, δ_k)-DP, then the mechanism that composes them sequentially $\mathcal{M}_{[k]}(D) = (\mathcal{M}_1(D), \ldots, \mathcal{M}_k(D))$ satisfies $(\sum_{i=1}^k \epsilon_i, \sum_{i=1}^k \delta_i)$-DP.

- **Parallel Composition:** A dataset can be partitioned and separately used in independent differentially private mechanisms without accumulating a privacy cost for each mechanism. Formally, if k mechanisms $\mathcal{M}_1, \ldots, \mathcal{M}_k$ each satisfy (ϵ_1, δ_1)-DP, $\ldots$, (ϵ_k, δ_k)-DP and an input dataset D is partitioned into k disjoint subsets as $D_1, \ldots, D_k$, then the mechanism that composes them in parallel $\mathcal{M}_{[k]}(D) = (\mathcal{M}_1(D_1), \ldots, \mathcal{M}_k(D_k))$ satisfies $(\max_{i \in [k]} \epsilon_i, \max_{i \in [k]} \delta_i)$-DP ([McS09], Theorem 4).

Together, these properties are extremely powerful for two reasons. The first is that they allow reasoning about how an individual's privacy diminishes as their data is used in various analyses. The second is that they allow designing a complex DP mechanism by combining multiple DP and

non-DP mechanisms, all without necessitating a novel privacy analysis for the newly designed mechanism.

1.1.2 Achieving Differential Privacy

The definition of differential privacy only tells us the constraint that a mechanism must have, not what the mechanism is or how to design the DP mechanism to solve a problem. We now describe the fundamental DP mechanisms and corresponding concepts that we build on in the thesis.

Randomized Response Mechanism

The most basic non-trivial DP mechanism is the *Randomized Response* mechanism [War65; Dwo11], which interestingly predates differential privacy by half a century. The purpose of the Randomized Response mechanism is simple: given a binary value as input, it releases a privatized version of it as a single binary value. Despite this simplicity, it is widely used as a fundamental building block in designing and analyzing more advanced DP mechanisms. In fact, we significantly extend the Randomized Response mechanism in Chapter 2. It is formally defined as follows.

Definition 1.1.2 (Randomized Response mechanism)**.** Let x be an input dataset consisting of a single bit. Given $p \in [1/2, 1]$, the Randomized Response mechanism $\mathcal{M}_{\text{RR}}$ is defined as:

$$\mathcal{M}_{\text{RR}}(x) = \begin{cases} x & \text{with probability } p \\ 1 - x & \text{otherwise.} \end{cases} \tag{1.1}$$

Theorem 1.1.3. The Randomized Response mechanism $\mathcal{M}_{\text{RR}}$ satisfies (ϵ, δ)-DP when $p = \frac{e^\epsilon + \delta}{e^\epsilon + 1}$.

Proof. For $\mathcal{M}_{\text{RR}}$ to satisfy differential privacy, both of the following inequalities must hold:

$$\Pr[\mathcal{M}_{\text{RR}}(x) = x] \leq e^\epsilon \Pr[\mathcal{M}_{\text{RR}}(1 - x) = x] + \delta, \text{ and}$$
$$\Pr[\mathcal{M}_{\text{RR}}(1 - x) = x] \leq e^\epsilon \Pr[\mathcal{M}_{\text{RR}}(x) = x] + \delta.$$

With the given parameter p, these inequalities are respectively equivalent to

$$p \leq e^{\epsilon}(1-p) + \delta, \text{ and}$$
$$1 - p \leq e^{\epsilon} p + \delta.$$

The former reduces to $p \leq \frac{e^{\epsilon}+\delta}{e^{\epsilon}+1}$, while the latter is trivially satisfied (predicated on $p \geq 1/2$). Therefore, setting $p = \frac{e^{\epsilon}+\delta}{e^{\epsilon}+1}$ ensures that the differential privacy constraints hold on arbitrary input x, thus concluding the proof. □

Recall that the high-level purpose of performing a private data analysis is to make use of the data *while simultaneously* ensuring privacy. So far, we have only discussed the privacy of the mechanism. However, the conclusion of this proof invites our first opportunity to formally consider the *utility* of a DP mechanism, as well as the *privacy–utility trade-off* of DP mechanisms in general. Specifically, one may wonder why we choose $p = \frac{e^{\epsilon}+\delta}{e^{\epsilon}+1}$ given that any $p \in \left[1/2, \frac{e^{\epsilon}+\delta}{e^{\epsilon}+1}\right]$ is sufficient to achieve (ϵ, δ)-DP. Since p represents the probability that the bit is truthfully reported (where $1-p$ represents the probability that flipped bit is reported instead), intuition suggests that privacy should increase as p decreases towards $1/2$. In fact, this intuition is correct — any value of $p < \frac{e^{\epsilon}+\delta}{e^{\epsilon}+1}$ would allow the mechanism to satisfy (ϵ, δ)-DP, *and it would also* satisfy (ϵ', δ')-DP for some $\epsilon' < \epsilon$ and $\delta' < \delta$. However, once we consider the utility of a mechanism — which, in the simple case of the Randomized Response mechanism, we can informally consider to be measured as the probability of truthfully reporting the bit (p) — we see that privacy and utility are directly at odds. Thus, the reason we choose $p = \frac{e^{\epsilon}+\delta}{e^{\epsilon}+1}$ is because this choice induces the greatest possible utility for the Randomized Response mechanism while still ensuring that the mechanism satisfies (ϵ, δ)-DP. To attain higher utility with this mechanism, p must be increased, which decreases privacy; conversely, to attain higher privacy, p must be decreased, which decreases utility. This is a concrete illustration of a central concept in differential privacy, the *privacy–utility trade-off*, a problem we address for more complex mechanisms in depth in Chapter 3.

Additive Noise Mechanisms

The most common approach to achieving differential privacy is through the use of *additive noise* mechanisms. Informally, these mechanisms compute some function's true (non-private) value, then perturb the value using carefully calibrated noise from a random distribution (typically with mean 0). Examples of distributions that have been used to ensure differential privacy include the Laplace [Dwo+06b], Gaussian [Dwo+06a], binomial [Dwo+06a], geometric [GRS12], and discrete Gaussian [CKS20] distributions. Here, we introduce the two most popular corresponding mechanisms which we utilize extensively throughout this thesis: the Laplace mechanism $\mathcal{M}_{\text{Lap}}$ and the Gaussian mechanism $\mathcal{M}_{\text{Gauss}}$.

We begin with the Laplace mechanism, defining it and stating its privacy guarantee. Informally, the Laplace mechanism adds random noise to a function's true value from the Laplace distribution calibrated to the function's ℓ_1 sensitivity (i.e., how much any hypothetical individual could influence the function's value).

Definition 1.1.4 (Laplace mechanism)**.** Let $D \in X$ be the input dataset, $f : X \to \mathbb{R}^k$ be an arbitrary function, and b be a non-negative real value. Define $\Delta_1(f) = \sup_{D', D''} ||f(D') - f(D'')||_1$, where the supremum is over all possible neighboring datasets $D', D'' \in X$ that differ in precisely one individual's data. The Laplace mechanism is defined as:

$$\mathcal{M}_{\text{Lap}}(D) = f(D) + (Y_1, \ldots, Y_k),$$

where each Y_i is independently drawn from the 0-mean Laplace(b) distribution.

Theorem 1.1.5 ([Bal+20])**.** The Laplace mechanism $\mathcal{M}_{\text{Lap}}$ satisfies (ϵ, δ)-DP when $b = \frac{\Delta_1(f)}{\epsilon - 2\ln(1-\delta)}$.

Now we define the Gaussian mechanism, then state two different variants of its privacy guarantee. Informally, the Gaussian mechanism adds random noise to a function's true value from the Gaussian distribution calibrated to the function's ℓ_2 sensitivity.

Definition 1.1.6 (Gaussian mechanism)**.** Let $D \in X$ be the input dataset, function $f : X \to \mathbb{R}^k$ be an arbitrary function, and σ be a non-negative real value. Define $\Delta_2(f) = \sup_{D',D''} ||f(D') - f(D'')||_2$, where the supremum is over all possible neighboring datasets $D', D'' \in X$ that differ in precisely one individual's data. The Gaussian mechanism is defined as:

$$\mathcal{M}_{\text{Gauss}}(D) = f(D) + (Y_1, \ldots, Y_k),$$

where each Y_i is independently drawn from the Normal$(0, \sigma^2)$ distribution.

The advantage of the Gaussian mechanism over the Laplace mechanism is that when ℓ_2 sensitivity is significantly lower than ℓ_1 sensitivity, the Gaussian mechanism can add significantly less noise to the function's true value. In other words, the Gaussian mechanism can attain higher utility. However, unlike the Laplace mechanism, the Gaussian mechanism is never able to satisfy pure DP; i.e., it cannot satisfy $(\epsilon, 0)$-DP for any ϵ.

Theorem 1.1.7 (Standard analysis, [DR+14])**.** For $\epsilon, \delta \in (0, 1)$, the Gaussian mechanism $\mathcal{M}_{\text{Gauss}}$ satisfies (ϵ, δ)-DP when $\sigma > \sqrt{2\ln(1.25/\delta)}\Delta_2(f)/\epsilon$.

Theorem 1.1.8 (Optimal analysis, [BW18])**.** Let $\Phi(t)$ denote the cumulative distribution function of the standard normal distribution Norm$(0, 1)$ at a point t. The Gaussian mechanism $\mathcal{M}_{\text{Gauss}}$ satisfies (ϵ, δ)-DP if and only if

$$\delta \geq \Phi\left(\frac{\Delta_2(f)}{2\sigma} - \frac{\epsilon\sigma}{\Delta_2(f)}\right) - e^{\epsilon}\Phi\left(-\frac{\Delta_2(f)}{2\sigma} - \frac{\epsilon\sigma}{\Delta_2(f)}\right).$$

The former theorem regarding the Gaussian mechanism's privacy is the weaker of the two, meaning it requires a larger magnitude of noise from the Gaussian distribution to achieve the same DP guarantee. On the other hand, the latter theorem is optimal. I.e., for fixed ϵ, δ and Δ_2, the minimal σ satisfying Theorem 1.1.8's inequality is the minimum possible σ that the Gaussian mechanism can use while still satisfying (ϵ, δ)-DP. Despite this optimality, the former is more

often leveraged in analyses and implementations involving the Gaussian mechanism due to its simple, closed-form expression. We have included both theorems to illustrate two points: (1) different analyses of the same differentially private mechanism can yield different differential privacy guarantees, and (2) although a mechanism or its analysis may be optimal, practitioners may opt for a suboptimal alternative if it better suits their needs.

1.2 Overview and Contributions

Despite the aforementioned real-world deployments of DP, significant barriers still hinder the wider adoption of differential privacy by companies, governments, and individual practitioners. One primary barrier is that attempting to solve a task with DP at a desired privacy level often results in inadequate utility. The sources of this barrier, however, are distinct and span the entire conceptual stack of differential privacy. We have categorized these sources into three distinct challenges:

1. The utility gap between the more and less desirable trust models of differential privacy.
2. The lack of tools for analyzing the complex, hyperparameterized DP mechanisms' utilities.
3. The open question of how to improve DP large-scale query-answering mechanisms' effectiveness and efficiency.

> *In this thesis, we address each of these challenges in order to make differential privacy more useful for real-world applications by removing barriers that hinder its adoption.*

Individually in each chapter, we address these challenges in the following ways.

Chapter 2: The Hybrid Model of Differential Privacy

In the first part of the thesis, we address the first identified challenge: There is a significant utility gap between the more and less desirable trust models of differential privacy. We start by describing the traditional trust models of DP, each with its own strengths and weaknesses. We then detail scenarios where these traditional trust models are not the best match for individuals' privacy expectations or data curators' privacy and utility goals. We suggest that in such scenarios, one can instead consider more relaxed trust models. Towards this, we define a "hybrid" DP trust model, which allows a combination of the traditional trust models.

Within this hybrid trust model, we thoroughly explore several important topics to understand the model's strength and to answer the high-level question of the chapter:

Within the hybrid model, how can we design DP mechanisms
that achieve high utility for problems of practical interest?

The specific topics that we explore are:

- How the utility of a mechanism in the hybrid model can be best understood and contextualized relative to the classic trust models.
- How mechanisms in the hybrid model can be designed for problems of practical interest.
- How the privacy and utility of mechanisms in the hybrid model depend on the computations performed and on assumptions regarding the individuals' data in their respective traditional trust models.

To address these topics, we design and analyze high-utility DP mechanisms in the hybrid model for two practical applications: heavy hitter discovery and estimation as well as mean estimation. In these applications, we theoretically and empirically show that the hybrid model can be more powerful than the classic trust models it is built upon.

Chapter 3: Quantifying the Privacy–Utility Trade-off

In the second part of the thesis, we address the second identified challenge: There is a lack of tools available for analyzing complex DP mechanisms' utilities. We specifically focus on the problem of quantifying the trade-off between privacy and utility of complex, hyperparameterized DP mechanisms. Unlike for simple DP mechanisms, where one can use analytical tools to reason about this trade-off, such tools are often unavailable for more complex DP mechanisms (e.g., those used for machine learning tasks). Moreover, DP mechanisms for these tasks often have many hyperparameters that affect not only their utility (as is typical in non-DP mechanisms) but also their privacy. This makes it difficult even to define what the mechanism's privacy–utility trade-off is, let alone quantify it. Thus, the high-level question that we answer in this chapter is:

> *How can we rigorously define a hyperparameterized DP mechanism's privacy–utility trade-off, and then how can we design a practical method for quantifying it?*

To answer this question, we begin by establishing a rigorous definition for the privacy–utility trade-off of hyperparameterized DP mechanisms. Based on this definition, we leverage multi-objective Bayesian optimization tools to develop a method that efficiently quantifies a DP mechanism's privacy–utility trade-off using only empirical measurements. We then thoroughly evaluate our new method, finding that it is highly effective for quantifying this trade-off on practical machine learning tasks.

Chapter 4: Pushing the Boundaries of Private, Large-Scale Query Answering

In the final part of the thesis, we address the third identified challenge: How to improve differentially private mechanisms' effectiveness and efficiency for the foundational problem of accurately releasing answers to a large number of queries. We address this in two different settings:

the classic setting in DP where all the queries to be answered are specified to the mechanism in advance, and a new setting we define where only partial knowledge of the queries is provided to the mechanism in advance. Within both settings, the high-level question that we answer is:

To what extent are differentially private mechanisms able to answer a large number of queries efficiently and with low error?

We ground our work in the state-of-the-art DP mechanism for answering prespecified sets of queries [Ayd+21]. We first perform a thorough reproducibility study on the mechanism's original analysis. In doing so, we improve its implementation, enabling it to answer a significantly larger number of queries. We then extend the mechanism's capabilities to answer r-of-k threshold queries — a more powerful, general class of queries than previously considered in non-theoretical works. In both settings, we thoroughly evaluate the extended mechanism, finding that it is able to efficiently and effectively answer extremely large sets of queries.

Taken together, our work advances the state of the art in differential privacy, making it easier to adopt for real-world uses and improving its practical applicability.

Chapter 2

The Hybrid Model of Differential Privacy

To address the first high-level challenge of this thesis (Section 1.2) that hinders DP's adoption in practice — the significant utility gap between the more and less desirable trust models of DP — we introduce the *hybrid model* of DP[2]. We initiate our study of the hybrid model by describing its motivation: how in some practical scenarios, the traditional DP trust models are not the best match for the individuals' and the data curators' privacy and utility desiderata. We then define our hybrid DP trust model as a flexible combination of the traditional DP trust models. Under this hybrid model, we design mechanisms for two important problems in data science: heavy hitter discovery and estimation as well as mean estimation. For both problems, we show that our respective mechanisms achieve high utility relative to mechanisms in the traditional trust models, thus demonstrating the power of the hybrid model.

2.1 Overview

Prior to our work, only two trust models were primarily considered in DP literature: the trusted-curator model (TCM) and the local model (LM). In the TCM, the organization (or data curator) first receives the individuals' true data, then takes on the responsibility of ensuring that any analysis performed on that data is differentially private. In the LM, the individuals first privatize

[2]This chapter is based on work in our publications [Ave+17; Ave+19; ADK20].

their own data to ensure DP, then the curator receives this already privatized data. Importantly, differential privacy is ensured in both models — the only distinction is the timing of when DP is ensured.

In practice, the LM may be a better match for curators' and individuals' privacy goals. Particularly, in the LM, individuals' privacy is assured even when they do not trust the curator, and the curator limits its liability in the face of data leaks. However, it is well understood theoretically and empirically that there is a utility gap between the models. Specifically, utility is far worse in the LM than in the TCM [BNO08; Kas+11; BS15; Shi+17].

Based on our observation that the TCM and LM can coexist, we develop a new trust model to enable the design of DP mechanisms which may bridge this utility gap while being amenable to practical use. Our *hybrid model* is a slight relaxation of the LM in which the majority of the individuals desire privacy in the LM, but where a small fraction of individuals choose to opt in to contributing their data under the TCM [Ave+17; Ave+19]. We often refer to individuals who opt in to using the TCM as "opt-in users" and those using the LM as "client users". The two groups each use their data under their respective trust models to compute a portion of the overall task. Each group has the optional ability to interact with the other group to share information that could potentially help the other group compute their portion of the task. Once both groups have computed their individual tasks, the curator aggregates their results and processes them into a final output. This hybrid model system is illustrated in Figure 2.1.

With this new model defined, the question we address in this chapter of the thesis is:

Within the hybrid model, how can we design DP mechanisms
that achieve high utility for problems of practical interest?

We answer this high-level question by decomposing it into the following three concrete questions, which we subsequently address.

1. *How can a hybrid mechanism's utility be best understood and contextualized relative to the classic trust models?*

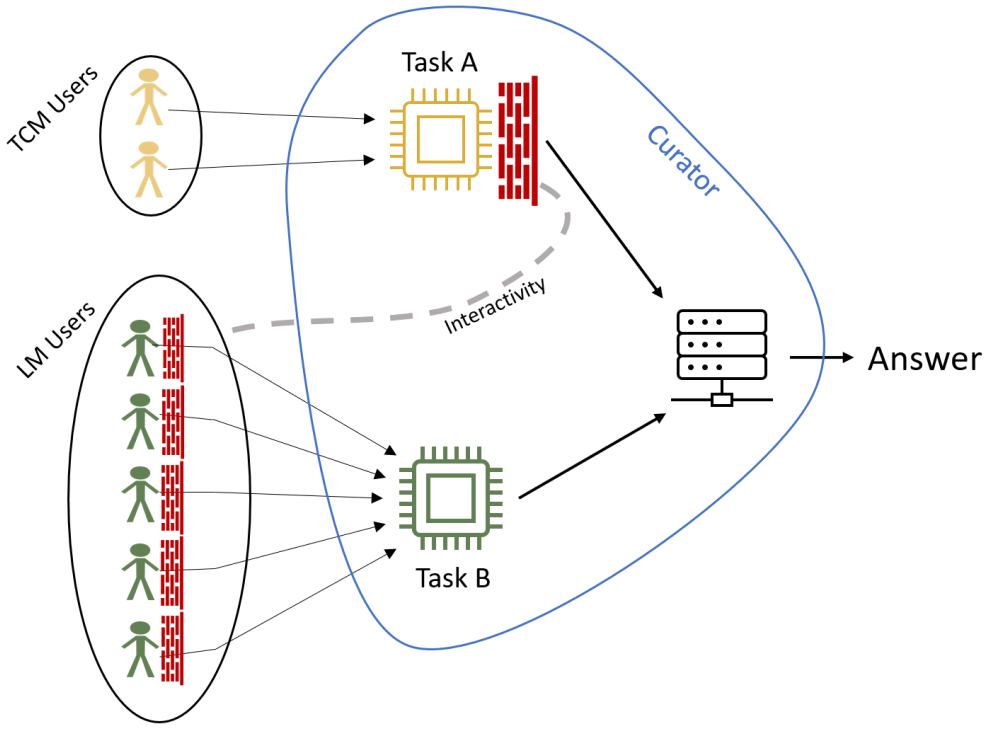


Figure 2.1: Overview of our hybrid differential privacy model's components.

2. *How can hybrid mechanisms be designed for problems of practical interest, and how does their utility compare to the utility of mechanisms in the traditional trust models?*

3. *How does the privacy and utility of a hybrid mechanism vary depending on the computations performed, and on assumptions regarding the two groups of individuals' data?*

Understanding Utility

To understand the utility of a hybrid mechanism, we contextualize it by comparing against baseline mechanisms in the classic trust models.

- TCM Baseline: Any TCM baseline mechanism must operate *only on the data of those individuals who opted in* to the TCM, in order to not violate their trust preferences.

- LM Baseline: Since the LM requires strictly less trust than the TCM, any LM baseline mechanism *can be applied to the data of all individuals* without violating anyone's trust.

Thus, we compare a hybrid mechanism's utility against: (1) the utility of a TCM baseline mechanism operating *only* on the data of the TCM individuals, and (2) the utility of an LM baseline mechanism operating on *all* individuals' data.

To choose the concrete baseline mechanisms in each classic trust model, we consider three natural strategies. The first candidate strategy is to use the theoretically optimal analogous mechanisms in the TCM and LM as baselines. However, optimal differentially private mechanisms are only known for the simplest of problems; e.g., single-bit queries or linear queries [Li13; GV14; KOV14; Bas19]. The second candidate strategy is to use the state-of-the-art analogous mechanisms in the TCM and LM as baselines. In practice however, the state-of-the-art mechanisms may not be widely deployed due to issues with computational overhead, communication cost, implementation complexity, etc. Thus, the third strategy is to use the most popular analogous mechanisms in the TCM and LM. In this chapter, we utilize both the second and third strategies to select baseline mechanisms.

Designing Hybrid Mechanisms

We propose two high-level approaches to leveraging the hybrid model when designing mechanisms.

The first is what we informally call a specialization-based approach, determined by the utilities that mechanisms in each classic trust model can achieve for different aspects of a given problem. Specifically, we: (1) split the problem into disjoint tasks, (2) determine mechanisms for each task in both the TCM and LM, and then (3) assign tasks to the TCM and LM groups based on their mechanisms' relative utility differences (i.e., specialization). We study whether a hybrid mechanism designed using this specialization-based approach can achieve higher utility

than both baselines, or if its utility will necessarily lie between (or even below) the two. Concretely, we use this approach to design a hybrid mechanism to address the problem of heavy hitter discovery and estimation for the particular application of local search (Section 2.2.1). Through a thorough empirical evaluation, we conclude that in certain scenarios, it is possible for such a hybrid mechanism to achieve higher utility than both baselines.

The specialization-based approach to designing hybrid mechanisms may not be applicable if the problem cannot readily be split into separate tasks. For such problems, a simpler approach is to: (1) apply the baseline TCM mechanism on the TCM individuals' data, (2) independently apply the baseline LM mechanism on the LM individuals' data, and then (3) interpolate their results. We informally call this the direct-combination approach, and study whether a hybrid mechanism designed using the direct-combination approach would have utility that is necessarily an interpolation of the baseline mechanisms' utilities, or if it could achieve higher utility than both baselines via strategically combining them. Concretely, we study the problem of mean estimation under the hybrid model by designing hybrid mean estimators that are a direct combination of baseline estimators in the classic trust models (Section 2.3.2). We then analytically evaluate the utilities of our hybrid estimators, and compare them against the baselines. We find and characterize scenarios where a strategic, direct combination of the two baselines results in a hybrid mechanism whose utility exceeds both baselines simultaneously.

Group Properties and Interaction

Because the hybrid model is composed of two distinct groups of individuals, the groups' behavior and interaction within the hybrid mechanism may influence the mechanism's privacy and utility.

One way these groups may influence a hybrid mechanism's utility is through selection bias stemming from each individual's decision of whether or not to opt in to the TCM. Towards this, we study what impact the two groups being dissimilar has on a hybrid mechanism's utility (Section 2.3.5). For the hybrid mean estimation problem, we find and analytically characterize the

scenarios in which heterogeneity between the groups has no negative impact on the hybrid mechanism's utility. Correspondingly, we find and characterize other scenarios where heterogeneity significantly reduces the mechanism's utility. For problems other than mean estimation, the impact of group heterogeneity on a hybrid mechanism's utility remains an open question.

Another way the groups may influence a hybrid mechanism's utility is through their intergroup interaction; i.e., through the information they share with each other. An ongoing line of work in the LM on interactive mechanisms studies how allowing interaction between individual LM individuals can enable the design of higher utility mechanisms [Kas+11; STU17; SU17; DF19; Jos+19b]. The difference in our work is that instead of studying how interaction between *individuals* can affect utility, we study how interaction *across the two groups* of individuals can affect utility. For instance, a hybrid mechanism may have one group first compute a partial solution for the task (e.g., if this group specializes in that portion of the task), and then share this partial result with the other group to aid their computation of the final solution. We study whether such interaction is necessary for hybrid mechanisms to achieve utility greater than the utility of analogous baseline mechanisms in the traditional trust models (Section 2.3). Our hybrid mechanism for the heavy hitter discovery and estimation problem utilizes intergroup interaction, whereas our hybrid mechanism for the mean estimation problem does not. However, both mechanisms are able to achieve utility greater than their baselines in certain scenarios. This demonstrates that such intergroup interaction is not always necessary to attain good utility.

Aside from its impact on utility, intergroup interaction may influence a hybrid mechanism's privacy. For instance, on one end of the spectrum, it may be that communicating a partial solution to a task from one group to another could weaken the first group's privacy. On the other end of the spectrum, it may be that the additional randomness induced downstream by the second group's application of a DP mechanism may further obfuscate the underlying data of the individuals in the first group, thus improving the first group's privacy against an adversary that only sees the output of the final computation (and not its intermediate steps). We investigate the extent that intergroup

interaction (or lack thereof) impacts the privacy guarantee for individuals in each group relative to their respective privacy guarantees under their chosen trust models (Section 2.4). For both the heavy hitter problem and the mean estimation problem, we determine which individuals will experience improved privacy guarantees, and we precisely quantify the improvement. We find that depending on the structure of the hybrid mechanism and on the precise method used to ensure DP, the improvement to users' privacy guarantees can be significant.

2.2 Heavy Hitter Discovery and Estimation

The first problem that we study under the hybrid model is that of heavy hitter discovery and estimation. Specifically, in this section of the thesis, the central problem we consider is:

> *How can we design a high utility hybrid mechanism for the problem of heavy hitter discovery and estimation?*

Heavy hitter discovery and estimation is the problem of finding the most popular items in a set and then estimating those items' frequencies. This is a classic and well studied problem in the context of information retrieval [AS+94; SON95; Toi+96; HPY00; CCFC02; Cor+03]. Because using these algorithms on sensitive personal data poses privacy risks, there has been significant research in the DP community on developing privacy-preserving heavy hitter discovery and estimation mechanisms. By now, heavy hitter discovery and estimation while preserving DP has become one of the canonical problems in DP literature under both the TCM [Bha+10; Li+12; Kor12] and the LM [EPK14; FPE16a; Qin+16].

The state-of-the-art mechanisms in the TCM and LM at the time of this work were that of Korolova [Kor12] and Qin et al. [Qin+16] respectively. While very different in their implementations, their high-level designs were roughly the same: first use a portion of the privacy budget to estimate which items are most likely to be heavy hitters, then use the remaining privacy budget to estimate the frequencies of those particular items. Qin et al. measured the utility of their

LM mechanism by computing its *normalized discounted cumulative gain* (NDCG)[3] on real-world datasets of search queries. The TCM mechanism of Korolova was specifically designed to operate on search log datasets (i.e., datasets of search queries, along with the corresponding URLs that individuals had clicked on) in order to generate a query click graph[4]. Korolova measured the utility of their mechanism by performing a side-by-side comparison against the non-private query click graphs of real-world search logs. Due to its high utility and usability, the principles of Korolova's mechanism have been deployed for real-world use in Google's general-purpose SQL library [Wil+20].

Our hybrid approach to this problem is motivated by the utility challenges of the mechanisms in the LM. Although the LM mechanism of Qin et al. achieved significantly higher utility than prior approaches in the LM, for many practical parameter regimes, its utility was extremely poor. For instance, computing the top-10 heavy hitters required $\epsilon > 10$ for their mechanism to achieve 90% NDCG on real-world datasets. Similarly, using a practical privacy budget ($\epsilon = 3$) on real-world datasets, their mechanism was unable to compute even the top-1 heavy hitter with NDCG exceeding 85%. The reason for this poor utility is due to the heavy hitter discovery portion of the task, which is significantly more challenging under the LM than under the TCM [BS15; FPE16a; Bas+20b]. This utility disparity between mechanisms in the two models is what motivates our hybrid approach: use the TCM individuals for what TCM mechanisms can compute with relatively high utility (heavy hitter discovery), then use the LM individuals for the portion of the task that LM mechanisms can compute with acceptable utility (frequency estimation).

Local Search Application

We study a real-world application of heavy hitter discovery and estimation: estimating the head list of a search log dataset to enable local search. The *head list* of a search log comprises a search

[3]NDCG is a standard measure of ranking quality in practice [JK02; Val+09] which takes into account both the presence/absence of items and their ordering.

[4]A query click graph is a graph where vertices correspond to both queries and URLs, with an edge connecting a query to a URL with weight equal to the number of individuals who posed the query then clicked that URL.

engine's most popular queries and their corresponding most clicked URLs. Storing a head list locally on individuals' devices makes *local search* possible: if an individual makes a query from the head list, results can be returned instantaneously, avoiding the need to communicate with a server. Such local caching of the most frequent search queries has a disproportionately positive impact on the expected query latency [Sil10; BY+07], as search engine queries follow a power-law distribution [BY+08]. Local search can also provide additional benefits, including smoothing temporary network disruptions or enabling entirely new features in web browsers.

Our Contributions

In practice, the primary challenge to solving local search is estimating the head list while simultaneously satisfying the following desiderata: (1) guaranteeing DP for each individual's data, (2) respecting each individual's trust preference, and (3) ensuring high practical utility for the problem of local search. In this section, we address this challenge by designing and evaluating a DP head list estimation mechanism within the hybrid model.

First, we thoroughly detail the design of BLENDER, a complex DP mechanism that we design using a specialization-based approach (Section 2.1) to achieve high-utility differentially private local search. Specifically, we define its general structure and the roles that the two groups of users have within it. We then thoroughly detail the sub-mechanisms that comprise BLENDER, and prove their differential privacy guarantees.

With BLENDER defined, we then specify the utility measures and state-of-the-art baseline mechanisms in the classic trust models that we use to evaluate the performance of BLENDER.

Finally, we perform a comprehensive empirical evaluation in order to quantify BLENDER's utility. The results of the evaluation not only show that BLENDER is able to achieve utility levels that are useful in practice, but they also answer a fundamental question about the hybrid model. In particular, BLENDER's utility is always at least as good as one of the baseline mechanisms. More importantly, in some scenarios, BLENDER is able to achieve utility greater than *both* baseline

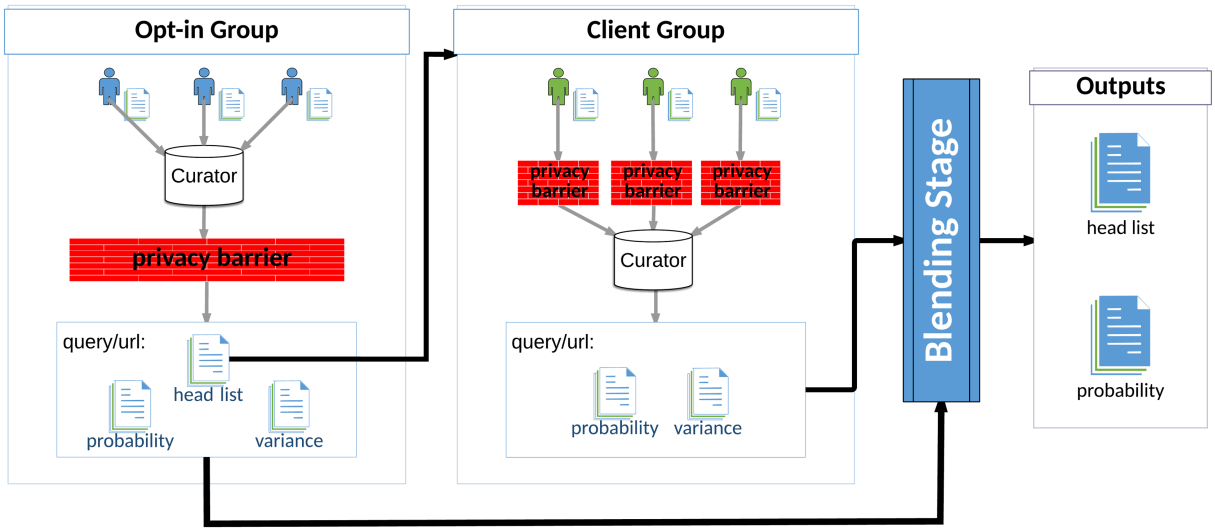


Figure 2.2: Architecture diagram of the BLENDER mechanism.

mechanisms simultaneously. This is the first demonstration that hybrid trust models can lead to non-trivial improvements in utility, and can, in fact, be strictly more powerful than either of the classic trust models.

2.2.1 Designing BLENDER

To address the DP head list estimation problem — and more generally to address the first two open questions of this chapter (Section 2.1) in order to begin overcoming the utility challenges of the LM and trust challenges of the TCM — we design the BLENDER mechanism that operates in the hybrid model. We provide an informal overview of the system first, and then dive into the details of its design and formal privacy properties.

2.2.1.1 Informal System Overview

A high-level overview of BLENDER is illustrated in Figure 2.2. The core idea behind BLENDER is to utilize a complex interaction strategy to take advantage of the aforementioned utility disparity between the models in the heavy hitter discovery portion of the task.

Towards this, based on the work of Korolova [Kor12], it first uses most individuals from the opt-in group (i.e., TCM group) to estimate which queries and corresponding URLs are most popular. This determines which records (i.e., query/URL pairs $\langle q, u \rangle$) are in the head list. In addition to these records, a single "wildcard" record $\langle \star, \star \rangle$ is included to represent all records in the population that were *not* included in the estimated head list. The small number of remaining reserved opt-in users (who were not used to estimate which records are in the head list) are then used to determine the head list records' order by estimating the empirical frequencies and variances of those head list records. Finally, the head list is optionally trimmed down to a final desired size.

Next, this estimated head list is sent to each user in the client group (i.e, LM group), bypassing the need for them to perform the "discovery" portion of the task. The clients then simply use the head list in conjunction with their own data to independently determine the head list records' ordering. They do so by also estimating the empirical frequencies and variances of the head list records. Each client then reports their individual frequency and variance estimates back to BLENDER.

Finally, BLENDER carefully combines both groups' frequency estimates (using their corresponding variance estimates) to generate a final ordering of the head list's queries and URLs.

2.2.1.2 Formal System Overview

We now proceed with the formal algorithmic definition of the BLENDER mechanism. First, we detail the core mechanism and each high-level stage, including the key parameters. We then detail the mechanisms that comprise each stage of BLENDER, including their differential privacy guarantees.

BLENDER Core Algorithm 2.1 presents the precise algorithmic overview of each step, including key parameters. Lines 1–3 of BLENDER describe the treatment of data from opt-in users, line 4 describes the treatment of data reported by clients, and line 5 describes the process for combining the probability estimates computed from the two groups. The only distinction between opt-in

users and clients in terms of privacy guarantees provided is the trust model; other than that, users from both groups are assured the same (ϵ, δ)-DP guarantee.

Algorithm 2.1 `BLENDER`

Input

- O, C: The set of opt-in users and clients.
- m_O, m_C: The maximum number of records to collect from each opt-in and client user.
- f_O: The fraction of the opt-in users to use for head list discovery (with the remaining used for head list estimation).
- f_C: the fraction of the clients' privacy budget to allocate to queries (as opposed to URLs).
- M: The maximum size of the finalized head list.
- ϵ, δ: The differential privacy parameters.

Body

1: Arbitrarily partition O into S and $T = O \setminus S$, such that $|S| = f_O|O|$ and $|T| = (1 - f_O)|O|$.
2: Let $HL_S = \texttt{DiscoverHeadList}(S, m_O, \epsilon, \delta)$ be the initial head list of records computed based on opt-in users' data.
3: Let $HL, \hat{p}_O, \hat{\sigma}^2_O = \texttt{EstimateOptinProbabilities}(T, m_O, HL_S, M, \epsilon, \delta)$ be the refined head list of records, their estimated probabilities, and estimated variances based on opt-in users' data.
4: Let $\hat{p}_C, \hat{\sigma}^2_C = \texttt{EstimateClientProbabilities}(C, m_C, f_C, HL, \epsilon, \delta)$ be the estimated record probabilities and estimated variances based on client reports.
5: Let $\hat{p} = \texttt{BlendProbabilities}(\hat{p}_O, \hat{\sigma}^2_O, \hat{p}_C, \hat{\sigma}^2_C, HL)$ be the combined estimate of record probabilities.
6: **Return:** $HL, \hat{p}$.

A key feature of `BLENDER` is that its privacy properties do not depend too strictly on the specific choices of these sub-mechanisms. That is, the post-processing property of differential privacy (Section 1.1.1) guarantees that if `DiscoverHeadList`, `EstimateOptinProbabilities`, and `EstimateClientProbabilities` each satisfy (ϵ, δ)-DP in their respective trust models (which we later prove they do), then `BLENDER` also satisfies (ϵ, δ)-DP. This allows changing the sub-mechanisms if better versions (utility-wise or implementation-wise) are discovered in the future.

Among the parameters of `BLENDER`, the privacy parameters and the sets of opt-in and client users can be viewed as externally set. On the other hand, the number of records collected from each user, the opt-in user group split, and the privacy budget split can all be viewed as knobs

that the implementer of BLENDER is at liberty to tweak in order to improve the overall utility of BLENDER's results.

Opt-in Group Mechanisms for Head List Discovery and Estimation We now detail the two mechanisms that are executed on the opt-in users' data. First, the opt-in users are partitioned into two sets — S, whose data will be used for initial head list discovery, and T, whose data will be used to estimate the probabilities and variances of records from the formed initial head list.

The initial head list discovery mechanism, described in Algorithm 2.2, constructs the list in a differentially private manner using search record data from group S. The mechanism follows the strategy first introduced by Korolova et al. [Kor+09] by aggregating the records of the opt-in users from S, and including those records whose noisy count exceeds a threshold in the head list. The Laplace noise added to the true counts and the threshold are calibrated to ensure DP. The goal of the mechanism is to approximate the true set of most frequently searched and clicked search records as closely as possible, while ensuring differential privacy. The DP guarantee of this mechanism is given in the following lemma.

Lemma 2.2.1. ([Kor12]) `DiscoverHeadList` satisfies (ϵ, δ)-differential privacy if $m_O = 1, \epsilon > \ln(2)$, and $\tau \geq 1$.

Our mechanism differs from previous work in two ways: 1) it replaces the collection and thresholding of queries with the collection and thresholding of records (i.e., query/URL pairs) and 2) its definition of neighboring databases is that of databases differing in one user's record values, rather than in the removal of one user's data. These distinctions necessitate the choice of $m_O = 1$ as well as higher values for noise and threshold than Korolova's mechanism [Kor12].

For those records included in the initial head list, the mechanism described in Algorithm 2.3 uses the remaining opt-in users' data (from set T) to differentially privately estimate all head list records' probabilities $\hat{p}_O$. The M most frequent records in $\hat{p}_O$ are retained to form the final head list. As a post-processing step, variance estimates for each of the probabilities are computed to

Algorithm 2.2 `DiscoverHeadList`

Input

- S: A set of opt-in users.
- m_O: The maximum number of records to collect from each opt-in user.
- ϵ, δ: The differential privacy parameters.

Body

for each user $i \in S$ **do**
  Let $D_{S,i}$ be the database aggregating at most m_O arbitrary records from i.
end for
Let D_S be the concatenation of all $D_{S,i}$ databases.
Let $N(r, D_S)$ denote the number of times any record r appears in D_S.
Let HL_S be an empty map.
Set $b_S = \frac{2m_O}{\epsilon}$.
Set $\tau = \max\{b_s \cdot \left(\ln(\exp(\frac{\epsilon}{2}) + m_O - 1) - \ln(\delta)\right), 1\}$.
for each distinct $\langle q, u\rangle \in D_S$ **do**
  Let Y be an independent sample from Laplace(b_S).
  if $N(\langle q, u\rangle, D_S) + Y > \tau$ **then**
    Add q to HL_S if $q \notin HL_S$.
    Append u to $HL_S[q]$.
  end if
end for
Add $\langle \star, \star \rangle$ to HL_S.
Return: HL_S.

be used in BLENDER's final blending stage. We formalize both the mechanism's DP guarantee and the unbiasedness of its variance computation as follows.

Algorithm 2.3 `EstimateOptinProbabilities`

Input

- T: A set of opt-in users.
- m_O: The max number of records to collect from each opt-in user.
- HL_S: The initial head list of records whose probabilities are to be estimated.
- M: The maximum size of the finalized head list.
- ϵ, δ: The differential privacy parameters.

Body

1: **for** each user $i \in T$ **do**
2: Let $D_{T,i}$ be the database aggregating at most m_O arbitrary records from i.
3: **end for**
4: Let $D_{T,i}$ be the database aggregating at most m_O arbitrary records from i.
5: Let $N(r, D_T)$ denote the number of times an arbitrary record r appears in D_T.
6: Transform any record $\langle q, u\rangle \in D_T$ that does not appear in HL_S into $\langle \star, \star\rangle$.
7: Let $\hat{p}_O$ be a vector indexed by records in HL_S containing the respective probability estimates.
8: Let $\hat{\sigma}^2_O$ be a vector indexed by records in HL_S containing variance estimates of the respective probability estimate.
9: Let $|D_T|$ denote the total number of records in D_T.
10: Set $b_T = \frac{2m_O}{\epsilon - 2\ln(1-\delta)}$.
11: **for** each $\langle q, u\rangle \in HL_S$ **do**
12: Let Y be an independent sample from Laplace(b_T).
13: Set $\hat{p}_{O,\langle q,u\rangle} = \frac{1}{|D_T|}(N(\langle q, u\rangle, D_T) + Y)$.
14: Set $\hat{\sigma}^2_{O,\langle q,u\rangle} = \frac{\hat{p}_{O,\langle q,u\rangle}(1-\hat{p}_{O,\langle q,u\rangle})}{|D_T|-1} + \frac{2b_T^2}{|D_T|\cdot(|D_T|-1)}$.
15: **end for**
16: Let HL map the M queries with the highest estimated marginal probabilities (according to $\hat{p}_O$) to their respective sets of URLs.
17: For the records not retained in HL, accumulate their estimated probabilities into $\hat{p}_{O,\langle\star,\star\rangle}$ and update $\hat{\sigma}^2_{O,\langle\star,\star\rangle}$ as in line 14.
18: **Return:** $HL, \hat{p}_O, \hat{\sigma}^2_O$.

Lemma 2.2.2. ([Bal+20]) `EstimateOptinProbabilities` satisfies (ϵ, δ)-differential privacy if $m_O = 1$.

Lemma 2.2.3. $\hat{\sigma}^2_{O,\langle q,u\rangle}$ is an unbiased variance estimate for the opt-in group's record probabilities if $m_O = 1$.

Proof. See end-of-chapter Appendix 2.A.

Finally, the head list is passed to the client group, and the head list and its probability and variance estimates are passed to the `BlendProbabilities` step of `BLENDER`. The choice of how to split opt-in users into the sub-groups of S and T and the choice of M are unrelated to privacy constraints, and can be made by `BLENDER`'s implementer to optimize utility goals. This is discussed further in Section 2.2.3.

Client Group Mechanisms for Head List Estimation The mechanism that each client user executes locally is defined in Algorithm 2.4, and the results are reported back to `BLENDER`. Here, records are no longer treated as a single entity, but rather in a two-stage process: first privatizing the query, then privatizing the URL. This helps optimize utility in the setting where the number of queries is significantly larger than the number of URLs associated with each query. To achieve differential privacy in each stage, we design a new DP mechanism in order to utilize the head list obtained from the opt-in group. This new DP mechanism is a two-fold generalization of the basic Randomized Response mechanism (Section 1.1.2). Specifically, our mechanism first reports the true query with a carefully calibrated probability, and otherwise reports a uniformly random query from all the other queries in the head list. It then follows a similar procedure to privately report the corresponding URL.

Lemma 2.2.4. `LocalReport` is (ϵ, δ)-differentially private.

Proof. We prove this statement by proving that each iteration of the `for` loop in line 8 of `LocalReport` is (ϵ', δ')-differentially private, where $\epsilon' = \epsilon/m_C$ and $\delta' = \delta/m_C$. If this claim holds, then because there are at most m_C iterations of this loop for each client, the sequential composition property of DP mechanisms (Section 1.1.1) guarantees that `LocalReport` ensures (ϵ, δ)-DP for each client.

Let L denote each iteration of the `for` loop in line 8 of `LocalReport`. L takes as input a record $\langle q, u\rangle \in D$, and returns a record $L(\langle q, u\rangle)$. If q is not in HL or u is not in $HL[q]$, then they immediately get transformed into a default value $(\star)$ that is in the head list. Since L outputs only

Algorithm 2.4 `LocalReport`

Input

- m_C: The maximum number of records to collect from the client.
- f_C: The fraction of the privacy budget to allocate to reporting queries.
- *HL*: The head list, represented as a map keyed by queries $\{q_1, \ldots, q_k, \star\}$. The value for each $q \in HL$ is defined as $HL[q] = \{u_1, \ldots, u_l, \star\}$, representing all URLs in the head list associated with q.
- ϵ, δ: The differential privacy parameters.

Body

Let $D_{C,i}$ be the database aggregating at most m_C records from current client i.
Let $\epsilon' = \epsilon / m_C$, and $\delta' = \delta / m_C$.
Let $\epsilon'_Q = f_C \epsilon', \epsilon'_U = \epsilon' - \epsilon'_Q$ and $\delta'_Q = f_C \delta', \delta'_U = \delta' - \delta'_Q$.
$k = |HL|$, and $t = \frac{\exp(\epsilon'_Q) + (\delta'_Q/2)(k-1)}{\exp(\epsilon'_Q) + k - 1}$.
for each $q \in HL$ **do**
    Set $k_q = |HL[q]|$, and $t_q = \frac{\exp(\epsilon'_U) + (\delta'_U/2)(k_q - 1)}{\exp(\epsilon'_U) + k_q - 1}$.
end for
for each $\langle q, u \rangle \in D_{C,i}$ **do**
    if $q \notin HL$ **then**
        Set $q = \star$.
    end if
    if $u \notin HL[q]$ **then**
        Set $u = \star$.
    end if
    With probability $(1 - t)$,
        Let q' be a uniformly random query from $HL \setminus q$.
        Let u' be a uniformly random URL from $HL[q']$.
        report $\langle q', u' \rangle$.
        continue
    With probability $(1 - t_q)$,
        Let u' be a uniformly random URL from $HL[q] \setminus u$.
        report $\langle q, u' \rangle$.
        continue
    report $\langle q, u \rangle$.
end for
Return: HL_S.

values that exist in the head list, we need to prove that for any arbitrary neighboring datasets $\langle q, u\rangle$ and $\langle q', u'\rangle$, $\Pr\big[L(\langle q, u\rangle) \in Y\big] \leq e^{\epsilon'} \Pr\big[L(\langle q', u'\rangle) \in Y\big] + \delta'$ holds for all sets of head list records Y.

Whenever $k = 1$ or $k_q = 1$, the only query (or URL for a specific query) is $\star$, which will be output with probability 1. Thus, the DP constraint trivially holds, since the reported values then do not rely on the client's data. Therefore, we assume $k \geq 2$ and $k_q \geq 2$. Because there is a single decision point where it is determined whether q will be reported truthfully or not, we can split the privacy analysis into two parts: 1) usage of the f_C fraction of the privacy budget to report a query, and 2) usage of the remainder of the privacy budget to report a URL (given the reported query). This decomposes a simultaneous two-item (ϵ', δ') reporting problem into two single-item reporting problems with (ϵ'_Q, δ'_Q) and (ϵ'_U, δ'_U) respectively, where $\epsilon'_Q = f\epsilon'$, $\delta'_Q = f\delta'$, $\epsilon'_U = (1-f_C)\epsilon'$, and $\delta'_U = (1 - f_C)\delta'$.

1. Privacy of query reporting: Consider the query reporting case first. Overloading our use of L, let $L(q)$ be the portion of L that makes use of q. We first ensure that

$$\Pr[L(q) = q_{HL}] \leq \exp(\epsilon'_Q) \Pr[L(q') = q_{HL}] + \frac{\delta'_Q}{2} \tag{2.1}$$

holds for all q, q', and $q_{HL} \in HL$. This trivially holds when $q_{HL} = q = q'$ or $q_{HL} \notin \{q, q'\}$. The remaining scenarios to consider are: 1) $q \neq q_{HL}, q' = q_{HL}$ and 2) $q = q_{HL}, q' \neq q_{HL}$. By the design of the mechanism, $\Pr[L(q_{HL}) = q_{HL}] = t$ and $\Pr[L(\bar{q}_{HL}) = q_{HL}] = (1 - t)(\frac{1}{k-1})$, where $\bar{q}_{HL}$ represents any query not equal to q_{HL}. With $t = \frac{\exp(\epsilon'_Q)+(\delta'_Q/2)(k-1)}{\exp(\epsilon'_Q)+k-1}$, it is straightforward to verify that inequality (2.1) holds.

Consider an arbitrary set of head list queries Y.

$$
\begin{aligned}
\Pr[L(q) \in Y] &= \sum_{q_{HL} \in Y} \Pr[L(q) = q_{HL}] \\
&= \sum_{q_{HL} \in Y \setminus \{q,q'\}} \Pr[L(q) = q_{HL}] + \sum_{q_{HL} \in Y \cap \{q,q'\}} Pr[L(q) = q_{HL}] \\
&= \sum_{q_{HL} \in Y \setminus \{q,q'\}} \Pr[L(q') = q_{HL}] + \sum_{q_{HL} \in Y \cap \{q,q'\}} \Pr[L(q) = q_{HL}] \qquad (2.2) \\
&\leq \sum_{q_{HL} \in Y \setminus \{q,q'\}} \Pr[L(q') = q_{HL}] + \sum_{q_{HL} \in Y \cap \{q,q'\}} \left(e^{\epsilon'_Q} \Pr[L(q') = q_{HL}] + \frac{\delta'_Q}{2}\right) \qquad (2.3) \\
&\leq e^{\epsilon'_Q} \sum_{q_{HL} \in Y} \Pr[L(q') = q_{HL}] + 2 \cdot \frac{\delta'_Q}{2} \\
&= e^{\epsilon'_Q} \Pr[L(q') \in Y] + \delta'_Q,
\end{aligned}
$$

Equality (2.2) stems from the fact that the probability of reporting a false query is independent of the user's true query. Inequality (2.3) is a direct application of inequality (2.1). Thus, L is (ϵ'_Q, δ'_Q)-differentially private for query reporting.

2. Privacy of URL reporting: With t_q defined as $t_q = \frac{\exp(\epsilon'_U)+(\delta/2)'_U(k_q-1)}{\exp(\epsilon'_U)+k_q-1}$, an analogous argument shows that the (ϵ'_U, δ'_U)-differential privacy constraints hold if the original q is kept. On the other hand, if it is replaced with a random query, then they trivially hold as the mechanism reports a random element in the URL list of the reported query, without taking into consideration the client's true URL u.

By the sequential composition property, each of the at most m_C iterations of L is $(\epsilon'_Q + \epsilon'_U, \delta'_Q + \delta'_U) = (\epsilon', \delta')$-differentially private. □

The fact that the head list (approximating the set of the most frequent records) is available to each client plays a crucial role in improving the utility of the data produced by this DP mechanism compared to the previously known mechanism operating in the local model. This allows use of the entire privacy budget to report the true value, rather than having to allocate some of it for estimating an analogue of the head list, as done in Fanti et al. [FPE16b] and Qin et al. [Qin+16].

The choices of m_C and f_C are not related to privacy constraints, and can be chosen by BLENDER's implementer to optimize utility goals, as will be discussed in Section 2.2.3.

In the EstimateClientProbabilities mechanism of BLENDER, defined in Algorithm 2.5, all client users' local reports are aggregated and processed. Because EstimateClientProbabilities only processes the clients' reports (never their private data directly), this mechanism trivially satisfies differential privacy by the post-processing property of DP. From a utility perspective, the LocalReport mechanism (i.e., using a randomization procedure that can report any record with some probability) induces a predictable bias to the distribution of reported records. To account for this, EstimateClientProbabilities performs a debiasing procedure in order to compute proper probability and variance estimates of the records.

Lemma 2.2.5. $\hat{p}_{C,\langle q,u\rangle}$ is an unbiased estimate of the clients' record probabilities.

Proof. See end-of-chapter Appendix 2.A.

Lemma 2.2.6. $\hat{\sigma}^2_{C,\langle q,u\rangle}$ is an unbiased variance estimate of the clients' record probabilities if $m_C = 1$.

Proof. See end-of-chapter Appendix 2.A.

The probability and variance estimates computed by EstimateClientProbabilities, $\hat{p}_C$ and $\hat{\sigma}^2_C$, are then passed to the BlendProbabilities stage of BLENDER.

Mechanism for Blending Estimates The blending portion of the BLENDER combines the independent estimates produced by the opt-in and client probability estimation mechanisms by taking into account the sizes of the groups and the amount of random noise each group's mechanism respectively added. This produces blended probability estimates $\hat{p}$ which, in expectation, are more accurate than either group produced individually. The procedure for blending is not subject to privacy constraints, as it operates on the data whose privacy has already been ensured by previous steps of BLENDER.

Algorithm 2.5 `EstimateClientProbabilities`

Input

- C: The set of clients.
- m_C: The maximum number of records to collect from each client.
- f_C: The fraction of the clients' privacy budget to allocate to queries.
- HL: A map from each query to its corresponding set of URLs.
- ϵ, δ: The differential privacy parameters.

Body

Append query $q = \star$ to HL.
for each query $q \in HL$ **do**
  Append URL $u = \star$ to $HL[q]$.
end for
for each client $i \in C$ **do**
  Let $D_{C,i} = \texttt{LocalReport}(m_C, f_C, HL, \epsilon, \delta)$ be the reports from client i's local execution of `LocalReport`.
end for
Let D_C be the concatenation of all reported client datasets, $D_{C,i}$.
Let $|D_C|$ denote the total number of records in D_C.
Let variables $\epsilon'_Q, \epsilon'_U, \delta'_Q, \delta'_U, k, t, k_q, t_q (\forall q \in HL)$ be defined as in lines 2–4 of `LocalReport`.
Let $\hat{r}_C, \hat{p}_C, \hat{\sigma}^2_C$ be vectors indexed by records in HL (which, overloading its use, can also be indexed by queries).
for $q \in HL$ **do**
  Let $\hat{r}_{C,q}$ be the fraction of queries q in D_C.
  Set $\hat{p}_{C,q} = \frac{\hat{r}_{C,q} - \frac{1-t}{k-1}}{t - \frac{1-t}{k-1}}$.
  Set $\hat{\sigma}^2_{C,q} = \frac{1}{\left(t - \frac{1-t}{k-1}\right)^2} \frac{\hat{r}_{C,q}(1-\hat{r}_{C,q})}{|D_C|-1}$
  for $u \in HL[q]$ **do**
    Let $\hat{r}_{C,\langle q,u\rangle}$ be the fraction of records which are $\langle q, u\rangle$ in D_C.
    Set $\hat{p}_{C,\langle q,u\rangle} = \frac{\hat{r}_{C,\langle q,u\rangle} - \left(t\frac{1-t_q}{k_q-1}\hat{p}_{C,q} + \frac{1-t}{k-1}\frac{1}{k_q}(1-\hat{p}_{C,q})\right)}{t(t_q - \frac{1-t_q}{k_q-1})}$
    Set

$$\hat{\sigma}^2_{C,\langle q,u\rangle} = \frac{|D_C|}{t^2\left(t_q - \frac{1-t_q}{k_q-1}\right)^2(|D_C|-1)} \cdot \left(\frac{\hat{r}_{C,\langle q,u\rangle}(1-\hat{r}_{C,\langle q,u\rangle})}{|D_C|} + \left(\frac{1-t}{(k-1)k_q} - t\frac{1-t_q}{k_q-1}\right)^2 \hat{\sigma}^2_{C,q} + 2\left(\frac{1-t}{(k-1)k_q} - t\frac{1-t_q}{k_q-1}\right)\left(\frac{\hat{r}_{C,\langle q,u\rangle}(1-\hat{r}_{C,q})}{|D_C|(t-\frac{1-t}{k-1})}\right)\right)$$

  end for
end for
Return: $\hat{p}_C, \hat{\sigma}^2_C$.

Algorithm 2.6 `BlendProbabilities`

Input

- $\hat{p}_O, \hat{p}_C$: The probability estimates from the opt-in and client mechanisms.
- $\hat{\sigma}_O, \hat{\sigma}_C$: The variance estimates from the opt-in and client mechanisms.
- *HL*: The head list of records.

Body

1: Let $\hat{p}$ be a vector indexed by records in *HL*.
2: **for** $\langle q, u \rangle \in HL$ **do**
3: $\quad w_{\langle q,u \rangle} = \frac{\hat{\sigma}^2_{C,\langle q,u \rangle}}{\hat{\sigma}^2_{O,\langle q,u \rangle} + \hat{\sigma}^2_{C,\langle q,u \rangle}}$.
4: $\quad \hat{p}_{\langle q,u \rangle} = w_{\langle q,u \rangle} \cdot \hat{p}_{O,\langle q,u \rangle} + (1 - w_{\langle q,u \rangle}) \cdot \hat{p}_{C,\langle q,u \rangle}$.
5: **end for**
6: Project $\hat{p}$ onto probability simplex (e.g., see [WCP13]).
7: **Return:** $\hat{p}$.

The motivation for this `BlendProbabilities` mechanism is born from the question of how to best combine the independent estimates of both groups. A standard measure of an estimator's quality is its variance. Although it may seem natural to choose the estimate with lower variance as the final estimate $\hat{p}$, it is possible to compute a better estimate by jointly utilizing the information provided by both estimates. This is because the errors in these estimates come from different sources. The error in the estimates obtained from `EstimateOptinProbabilities` stems from the Laplace mechanism applied to a small number of users' data, whereas the error in the estimates obtained from `EstimateClientProbabilities` stems from our generalization of the Randomized Response mechanism applied to almost all of the users. In the following theorem, we create a joint estimate that accounts for these different sources and scales of error.

Theorem 2.2.7. If $\hat{\sigma}^2_{O,\langle q,u \rangle}$ and $\hat{\sigma}^2_{C,\langle q,u \rangle}$ are sample variances of $\hat{p}_{O,\langle q,u \rangle}$ and $\hat{p}_{C,\langle q,u \rangle}$ respectively, and the blended estimate is the convex combination $\hat{p}_{\langle q,u \rangle} = w_{\langle q,u \rangle} \cdot \hat{p}_{O,\langle q,u \rangle} + (1 - w_{\langle q,u \rangle}) \cdot \hat{p}_{C,\langle q,u \rangle}$, then the sample variance optimal weighting is given by $w_{\langle q,u \rangle} = \frac{\hat{\sigma}^2_{C,\langle q,u \rangle}}{\hat{\sigma}^2_{O,\langle q,u \rangle} + \hat{\sigma}^2_{C,\langle q,u \rangle}}$.

Proof. Let the unbiased probability and variance estimates for each group's records be computed as in Lemmas 2.2.3, 2.2.5, and 2.2.6. The unbiased blended estimate of $p_{\langle q,u \rangle}$ is then defined as the convex combination of both groups' estimates: $\hat{p}_{\langle q,u \rangle} = w_{\langle q,u \rangle} \cdot \hat{p}_{O,\langle q,u \rangle} + (1 - w_{\langle q,u \rangle}) \cdot \hat{p}_{C,\langle q,u \rangle}$. The

sample variance of $\hat{p}_{\langle q,u \rangle}$ is given by $\hat{\sigma}^2_{\langle q,u \rangle} = w^2_{\langle q,u \rangle} \cdot \hat{\sigma}^2_{O,\langle q,u \rangle} + (1 - w_{\langle q,u \rangle})^2 \cdot \hat{\sigma}^2_{C,\langle q,u \rangle}$. Minimizing $\hat{\sigma}^2_{\langle q,u \rangle}$ with respect to $w_{\langle q,u \rangle}$ yields the stated result. □

With all sub-mechanisms of BLENDER fully defined, and with their differential privacy guarantees proven, we conclude by stating BLENDER's formal DP guarantee.

Theorem 2.2.8. BLENDER satisfies (ϵ, δ)-DP for all users.

Proof. Let O be the full set of opt-in users, and C be the full set of clients. Let S, T be the partition of opt-in users such that the users in S are assigned to the `DiscoverHeadList` sub-mechanism, and the users in T are assigned to the `EstimateOptinProbabilities` sub-mechanism. Because both sub-mechanisms satisfy (ϵ, δ)-DP for their respective disjoint subsets of users (Lemmas 2.2.1 and 2.2.2) and those users' raw data are never subsequently processed, by parallel composition (Section 1.1.1) we conclude that BLENDER ensures (ϵ, δ)-DP for the opt-in users O. Similarly, each client's local execution of `LocalReport` satisfies (ϵ, δ)-DP (Lemma 2.2.4). Since the clients' raw data are never subsequently processed, we conclude that BLENDER ensures (ϵ, δ)-DP for all clients C. Taken together, the privacy guarantees for both groups imply that BLENDER satisfies (ϵ, δ)-DP for all users. □

2.2.2 Measuring Utility

The practical measures of utility that we consider for head list estimation mechanisms in any trust model are ℓ_1 error (smaller is better) as well as the industry-standard NDCG (larger is better). For both measures, rather than theoretically analyzing BLENDER's (or any other DP mechanism's) worst-case utility, we evaluate utility experimentally using real-world search log datasets. We briefly describe both utility measures here, as well as the baseline mechanisms that we use to contextualize BLENDER's utility.

ℓ_1 Error: The ℓ_1 error is the Manhattan distance between the estimated and true probability vectors; i.e., $\sum_q |\hat{p}_q - p_q|$. We use this metric to specifically evaluate how well BLENDER estimates

the probabilities of queries (rather than entire records) in the head list. For any query q not in BLENDER's estimated head list, we assume that BLENDER implicitly estimates its probability as $\hat{p}_q = 0$.

NDCG: NDCG is a standard measure of search quality [JK02; Val+09] that takes into account the order of queries by performing *discounting*. In particular, most popular queries at the *head* of the search have a higher weight, whereas the relative significance of the less popular queries is reduced. The relevance, or gain, of an item at position i in the ranked list is measured using a graded relevance score defined as $rel_i = \frac{n_i}{\sum_j n_j}$, where n_j is the number of occurrences of the item in position j in the given dataset. The closer i's estimated rank is to its true rank, the larger the gain. For a *head* of k top elements, the estimated rank list is computed as $DCG_k = \sum_{i=1}^{k} \frac{2^{rel_i}-1}{\log_2(i+1)}$.

Here, the discounting happens because of the $\log_2(i)$ factor that diminishes the effect of later queries. This value is normalized by the Ideal DCG ($IDCG_k$), in which the estimated and the actual ranking are exactly the same, to obtain a value that ranges between 0 and 1.

Since we operate on records rather than just queries, we utilize a generalization of the traditional NDCG. Here, we compute the NDCG of each query's URL list, $NDCG^q$, as specified above, and then compute the DCG of the queries as $DCG_k^Q = \sum_{i=1}^{k} \frac{2^{rel_i}-1}{\log_2(i+1)} \cdot NDCG^i$.

The final NDCG computation is then DCG_k^Q normalized by the analogous Ideal DCG ($IDCG_k^Q$). In a way, our computation considers an NDCG of NDCGs, which makes it even harder for mechanisms to maintain consistently high NDCG values when compared to the query-only case. This formulation takes the true ranking and frequencies from the dataset into account, not the frequency estimates that BLENDER outputs. Since changes to the true frequencies may not result in ranking changes, ℓ_1 error is an even less forgiving measure than NDCG.

Since the purpose of BLENDER is to estimate probabilities of the top records, we discard the artificially added $\star$ queries and URLs and rescale rel_i prior to ℓ_1 and NDCG computations. However, since we use the probability projection method [WCP13] in `BlendProbabilities`, the probability estimates involving $\star$ have a minor implicit effect on the ℓ_1 error and NDCG.

	AOL	Yandex
Dataset on disk	1.75 GB	16 GB
Unique queries	4,811,646	13,171,961
Unique clients	519,371	4,970,073
Unique URLs	1,620,064	12,702,350

Table 2.1: Search log dataset statistics.

Baseline Mechanisms: To put the utility of the BLENDER mechanism into context, we use the following baseline mechanisms (previously described in Section 2.2) in each of the classic trust models. For the TCM baseline, we adapt Korolova's mechanism [Kor12] to the task of head list discovery and estimation applied *only* to the TCM users. Since the opt-in group's DiscoverHeadList and EstimateOptinProbabilities mechanisms were based on this adapted mechanism, the results of this baseline are equivalent to if BLENDER had no client group. For the LM baseline, we utilize Qin et al.'s mechanism [Qin+16] applied to *all individuals*.

2.2.3 Evaluating BLENDER

With BLENDER defined, we perform a comprehensive empirical evaluation in order to quantify its utility, finding that BLENDER is able to achieve practically useful levels of utility. We precede BLENDER's empirical evaluation with an example analysis to concretely illustrate how its components function individually and together. We then describe the parameter choices that must be made and justify how we choose them. Finally, we describe the various experiments that we perform on BLENDER and detail their results.

The datasets that we use for all experiments are the AOL search logs released in 2006 and the Yandex search dataset[5] released in 2013. Table 2.1 describes their characteristics.

[5] https://www.kaggle.com/c/yandex-personalized-web-search-challenge/data

Query	**AOL dataset** p_q	**BLENDER estimate** $\hat{p}_q$	**Opt-in estimate** $\sum_u \hat{p}_{O,\langle q,u\rangle}$	**Client estimate** $\hat{p}_{C,q}$	**Client estimate** $\sum_u \hat{p}_{C,\langle q,u\rangle}$
$\star$	0.9121	0.9144	0.9148	0.9143	0.9143
google	0.0211	0.0211	0.0220	0.0210	0.0210
yahoo	0.0067	0.0081	0.0061	0.0088	0.0088
google.com	0.0066	0.0075	**0.0083**	0.0073	0.0073
myspace.com	0.0055	0.0046	0.0034	0.0052	0.0052
mapquest	0.0055	**0.0062**	**0.0051**	**0.0066**	**0.0066**
yahoo.com	0.0048	0.0047	**0.0057**	0.0043	0.0043
www.google.com	0.0044	0.0038	0.0043	0.0035	0.0035
myspace	0.0034	0.0030	0.0031	0.0030	0.0030
ebay	0.0030	0.0030	0.0030	0.0029	0.0029

Table 2.2: Comparison of probability estimates for top-10 most popular AOL queries. Parameter choices are shown in Table 2.3, with $\epsilon = 3$ here.

2.2.3.1 Illustrative Analysis

To illustrate the approach we take for assessing result quality, Table 2.2 shows the top-10 most frequent queries in the AOL dataset alongside the probability estimates given by BLENDER's various sub-mechanisms.

The table is sorted by column 2, which contains the non-private, empirical probabilities from the AOL dataset with 1 random record sampled from each user. Column 3 contains the final query probability estimates output by BLENDER. Each sub-mechanism computes probability estimates over the *records* in the head list. To obtain *query* probability estimates from these record estimates, we simply aggregate the probabilities associated with each URL for a given query (columns 4 and 6). The sample variance of each aggregated probability, used for blending, is naively computed as in Theorem 2.2.3. Column 5 is the EstimateClientProbabilities' estimate of the query probabilities, which it directly computes. Column 6 contains the same information, but is computed by aggregating the estimated probabilities of EstimateClientProbabilities' records corresponding to a specific query. BLENDER uses columns 4, 5, and 6 when it comes to blending the records. Regressions — i.e., estimates that appear out of order relative to column 2 — are shown in red.

The biggest takeaway is that the numbers in columns 2 and 3 are similar to each other, with BLENDER's usage resulting in only one regression. This is an example of BLENDER compensating for the weaknesses of the opt-in and the client estimates. Specifically, despite the opt-in group having several regressions in this particular instance, combining the opt-in and client data compensates for that, resulting in only one regression.

2.2.3.2 Parameter Choices

BLENDER has a handful of parameters, some of which can be viewed as given externally (e.g, by business interests, legal requirements, etc.), and others whose choice is purely up to the entity deploying BLENDER. We now describe and motivate our choices for these parameters' values.

Privacy parameters (ϵ, δ). We view ϵ and δ as externally given privacy parameters (e.g., by what is a common practice for differentially private mechanisms in the industry [Tan+17; Tea17; EPK14]). We use a δ that is larger for the AOL dataset than for the Yandex dataset to reflect that the Yandex dataset contains data of more users. However, we ensure that for a fixed dataset, we use the same ϵ and δ values for the opt-in and client users. From a behavioral perspective, this reduces a user's opt-in decision down to one purely of trust towards the curator.

Opt-in and client group sizes $(|O|, |C|)$. The relative sizes of the opt-in and client groups, $|O|$ and $|C|$ respectively, can be viewed as exogenous variables which are dictated by the trust that users place in the search engine[6]. We choose 5% for AOL and 3% for Yandex for the fraction of opt-in users because they are both reasonably small while still allowing us to effectively demonstrate BLENDER's utility benefits.

[6]In the future, as differential privacy gains widespread adoption, it is conceivable that the values of the privacy parameters may affect their relative sizes; for example, the smaller the ϵ, the more users are willing to "opt in". However, this relationship is out of the scope of this work.

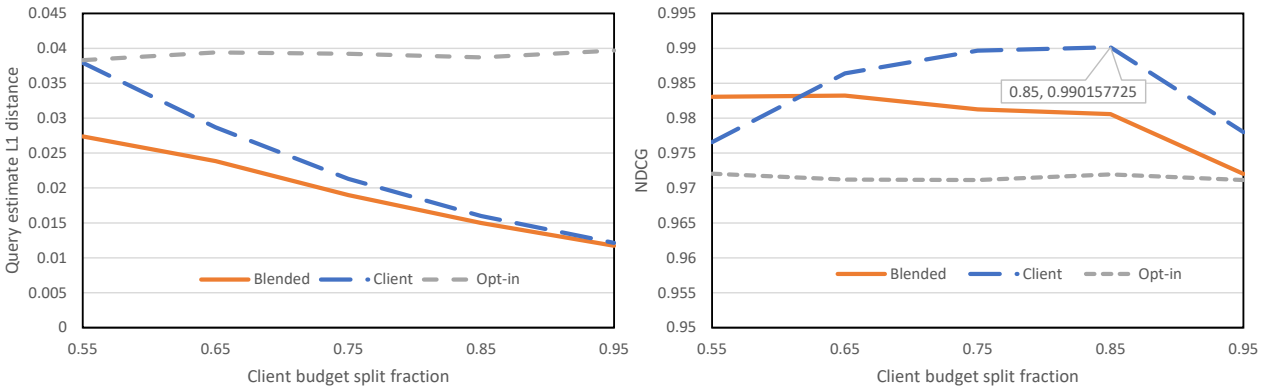
Number of records to collect from each opt-in user ($m_O = 1$). This is a choice necessitated by the privacy constraints of the `DiscoverHeadList` mechanism.

The choices for remaining the parameters, m_C, f_C, f_O, and M, are driven purely by utility considerations.

Number of records to collect from each client ($m_C = 1$). Across a range of experimental values, we found that collecting 1 record per user always yielded the greatest utility, thus justifying this parameter choice. Two factors account for this. The first factor is that the privacy budget must be split across a client's reports, reducing each individual report's utility. The second factor is that the variance estimates used in the blending stage assume that reports are uncorrelated — this assumption likely does not hold in practice within a given user's set of records.

Privacy budget split for clients ($f_C = 0.85$). Figure 2.3 shows the effects of the budget split on both the ℓ_1 and NDCG metrics. Unsurprisingly, Figure 2.3a shows that the larger a client's privacy budget fraction dedicated to query estimation is (as opposed to URL estimation), the better the ℓ_1 error for the client and `BLENDER` results. The NDCG metric in Figure 2.3b shows a trade-off that emerges as we assign more budget to the queries, de-emphasizing the URLs. The client mechanism's NDCG value peaks at a budget split of 0.85; choosing a split above this point induces a significant drop in the blended NDCG values. Note that the grey opt-in line in Figure 2.3b is constant, as the opt-in group is not affected by the client group's budget split.

Fraction of opt-in data dedicated to head list discovery ($f_O = 0.95$). We choose $f_O = 0.95$ because our goal is to build a large candidate head list. Unless we allocate most of the opt-in user data to building such a head list, `BLENDER`'s subsequent results may be accurate, but they will apply only to a small number of records. In order for `BLENDER` to be effective for the local search application in practice, it needs to amass a head list of at least double or triple digits in size. Even without looking at experimental data, this choice makes intuitive sense: the opt-in group size is



(a) L1

(b) NDCG

Figure 2.3: Comparing AOL dataset results across a range of budget splits for client, opt-in, and blended results.

small relative to the client group size, and it is difficult to generate a head list in the clients' local privacy model — thus, it makes sense to utilize most of the opt-in group's data for the task that is most difficult in the LM.

Final head list size (M). The choice of M is influenced by competing considerations. The larger the head list for which BLENDER provides probability estimates, the more effective the local search application (subject to those probability estimates being accurate). However, as the desired head list size increases, the accuracy of BLENDER's estimates drops (most notably due to client data privatization). We want to strike a balance that allows BLENDER to get a sensibly large record set with reasonably accurate probability estimates for it. We choose $M = 50$ and $M = 500$ for the AOL and Yandex datasets, to reflect their respective size differences.

Subsequently, we use the parameters shown in Table 2.3 in all experiments unless explicitly stated otherwise.

2.2.3.3 Utility Comparison Against Baseline Mechanisms

We now evaluate BLENDER's utility compared to the baseline mechanisms in both classic trust models.

Parameter	AOL	Yandex
ϵ	4	4
δ	10^{-5}	10^{-7}
$\frac{\lvert O\rvert}{\lvert O\rvert+\lvert C\rvert}$	5%	3%
m_O	1	1
m_C	1	1
f_O	0.95	0.95
f_C	0.85	0.85
M	50	500

Table 2.3: Default parameters used in BLENDER experiments.

As previously described in Section 2.2.2, the baseline mechanisms that we use in the TCM and LM are that of Korolova [Kor12] and Qin et al. [Qin+16] respectively. Qin et al. evaluates the NDCG of their state-of-the-art mechanism on the AOL dataset for the head list size of 10 across a range of ϵ values. Thus, we evaluate the NDCG of BLENDER and of our adapted variant of Korolova's mechanism on the same head list size and ϵ values. The outcome of this comparison is displayed in Figure 2.4. Across the full range of the ϵ privacy parameter considered, BLENDER achieves NDCG values above 95%. The TCM baseline performs fairly well, exceeding 95% beginning at $\epsilon = 4$. However, the LM baseline mechanism only attains NDCG values of 30% at its peak. We believe that given the intense focus on search optimization in the field of information retrieval, NDCG values as low as those of Qin et al. are generally unusable in practice. Overall, and particularly at the higher levels of privacy ($\epsilon \leq 3$), BLENDER significantly outperforms the closest related state-of-the-art mechanisms.

We do provide one remark on the difference in utility computation between our work and that of Qin et al.; specifically, we use a slightly different scoring function in our NDCG computation. Qin et al. use a relevance score based purely on the rank of queries in the original AOL dataset. This results in penalizing misranked queries regardless of their underlying probabilities. BLENDER's relevance score only relies on the underlying probabilities, so misranked items with similar underlying probabilities only have a small negative impact on the overall NDCG score. We believe this is a more natural scoring method, thus justifying our choice. Although this choice

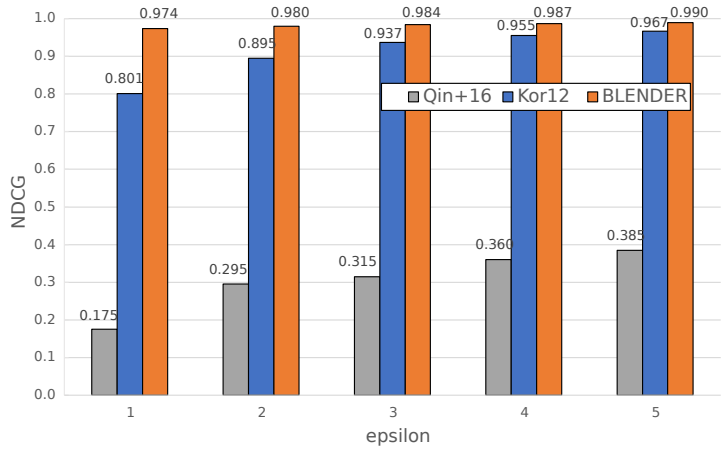
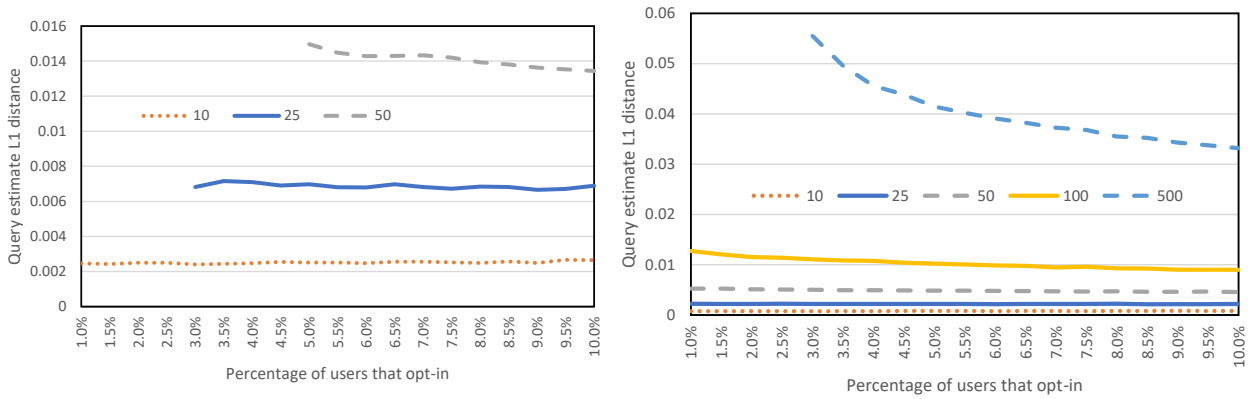


Figure 2.4: Comparing BLENDER's utility to TCM and LM baseline mechanisms across a range of ϵ values at a head list size of 10.

yields a slightly increased NDCG, this increase is outweighed by the fact that BLENDER operates on records (rather than queries, as in Qin et al.). Because of this, the "NDCG of NDCGs" score (Section 2.2.2) used to evaluate BLENDER is a strictly less forgiving metric than the traditional NDCG score. Thus, although simultaneously compensating for both differences would yield the ideal comparison, the comparison in Figure 2.4 is reasonable.

2.2.3.4 Evaluating How Parameter Choices Affect Utility

We now evaluate how BLENDER's utility is affected by the various parameter choices — primarily, the size of the opt-in group and the ϵ privacy parameter. We first evaluate BLENDER's utility for the simpler problem of discovering and estimating the top *queries* using the ℓ_1 error metric. We then evaluate BLENDER's utility on the more challenging problem of discovering and estimating the top *queries and URLs* using NDCG. Next, we analyze how each group of users contributes to BLENDER's final utility. Finally, we examine how BLENDER's utility is affected when the opt-in group is extremely small.



(a) AOL

(b) Yandex

Figure 2.5: BLENDER's ℓ_1 error as a function of the opt-in percentage.

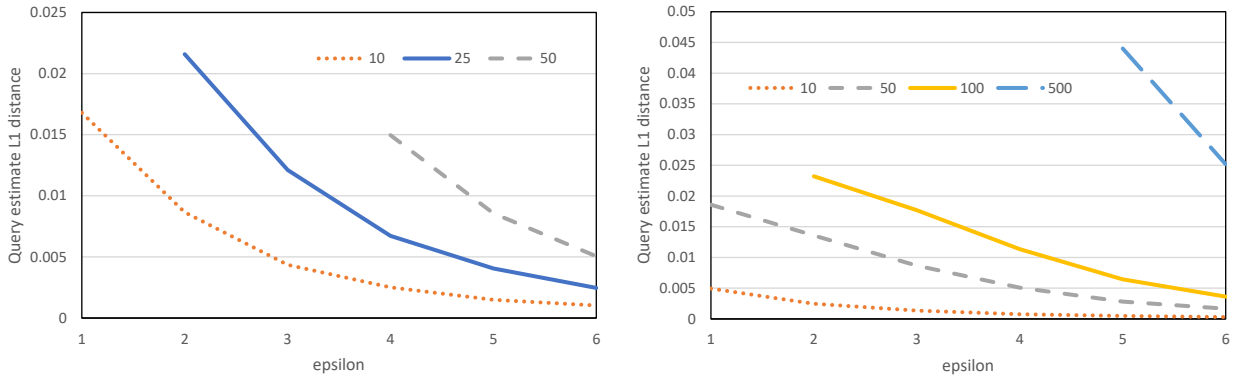
ℓ_1 evaluation of BLENDER's utility. Evaluating BLENDER's ℓ_1 error on the AOL and Yandex datasets, Figure 2.5[7] shows the results across opt-in user percentages ranging between 1% and 10%[8].

We observe slight differences in the two datasets and across the various head list sizes. Some differences may be due to the fact that given the relatively small size of the AOL dataset, BLENDER needs a higher percentage of opt-in users to achieve reasonably sized head lists and ℓ_1 error values. In fact, when we increase the opt-in percentage to 10% for the AOL dataset, we see a slight decline in ℓ_1 error for the largest head list size similar to what is observed in Figure 2.5b for the Yandex dataset. If the goal is to have head lists of size 500+, we see that with the larger Yandex dataset, an opt-in percentage as small as 3% is sufficient. The main takeaway from this is that when the opt-in group is large enough to attain the desired head list size, the estimated query probabilities will generally be high quality in terms of their ℓ_1 error.

Figure 2.6 shows BLENDER's ℓ_1 error as a function of ϵ, ranging from 1 to 6. For both datasets, BLENDER's ℓ_1 error steadily declines, achieving values under 0.01 for all but the smallest ϵ values and largest head list sizes.

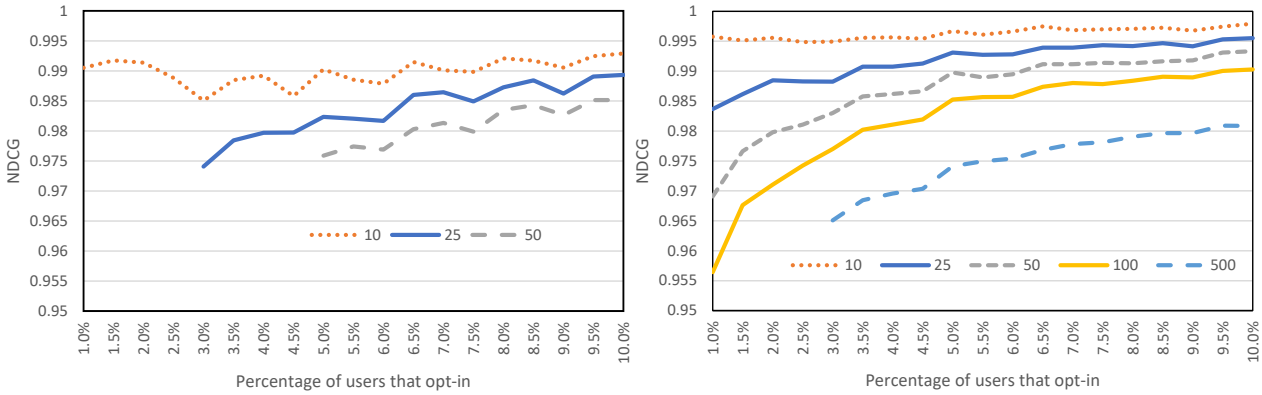
[7]Portions of lines do not appear on figures if the desired head list size was not reached (e.g., in Figure 2.5a, the line representing results for a head list of size 50 does not begin until 5% because a head list of size 50 could not be generated with a lower opt-in percentage).

[8]We believe that requiring opt-in percentages in excess of 10% is likely to put undue strain on the system in terms of recruitment; simply put, finding enough opt-in users may prove difficult or impossible in the long run.



(a) AOL (b) Yandex

Figure 2.6: BLENDER's ℓ_1 error on the AOL and Yandex datasets at various head list sizes across a range of ϵ values.

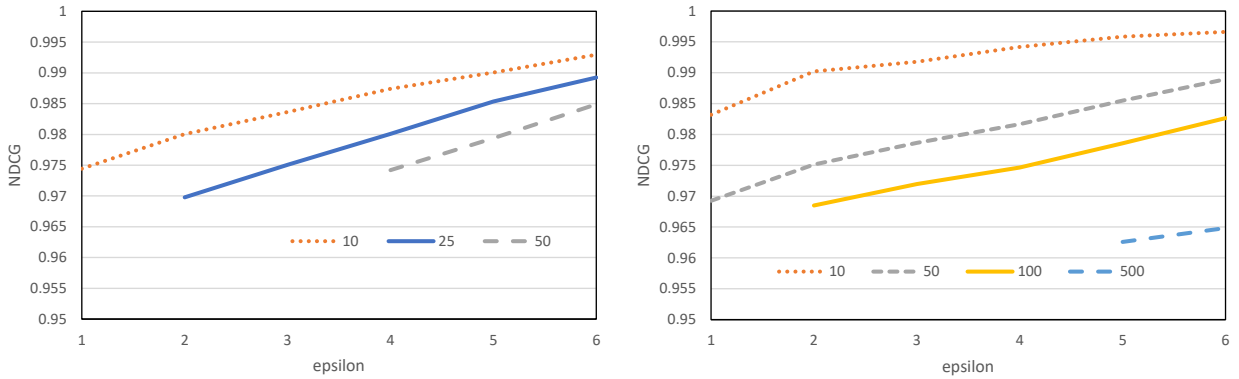


(a) AOL (b) Yandex

Figure 2.7: BLENDER's NDCG as a function of the opt-in percentage.

NDCG evaluation of BLENDER's utility. We now measure BLENDER's NDCG as a function of the opt-in percentage ranging between 1% and 10%. Figure 2.7 shows the corresponding results.

Despite query/URL record estimation being a more challenging problem than query estimation alone, the results here are quite encouraging. For the smaller AOL dataset, BLENDER achieves an NDCG in excess of 95% when the percentage of users in the opt-in group is at least 5%, which we regard as acceptable. However, for the larger Yandex dataset, BLENDER achieves that same NDCG level significantly sooner. For an opt-in group composed of only 1% of the total users, BLENDER's NDCG is above 95% for all but the largest head list size.



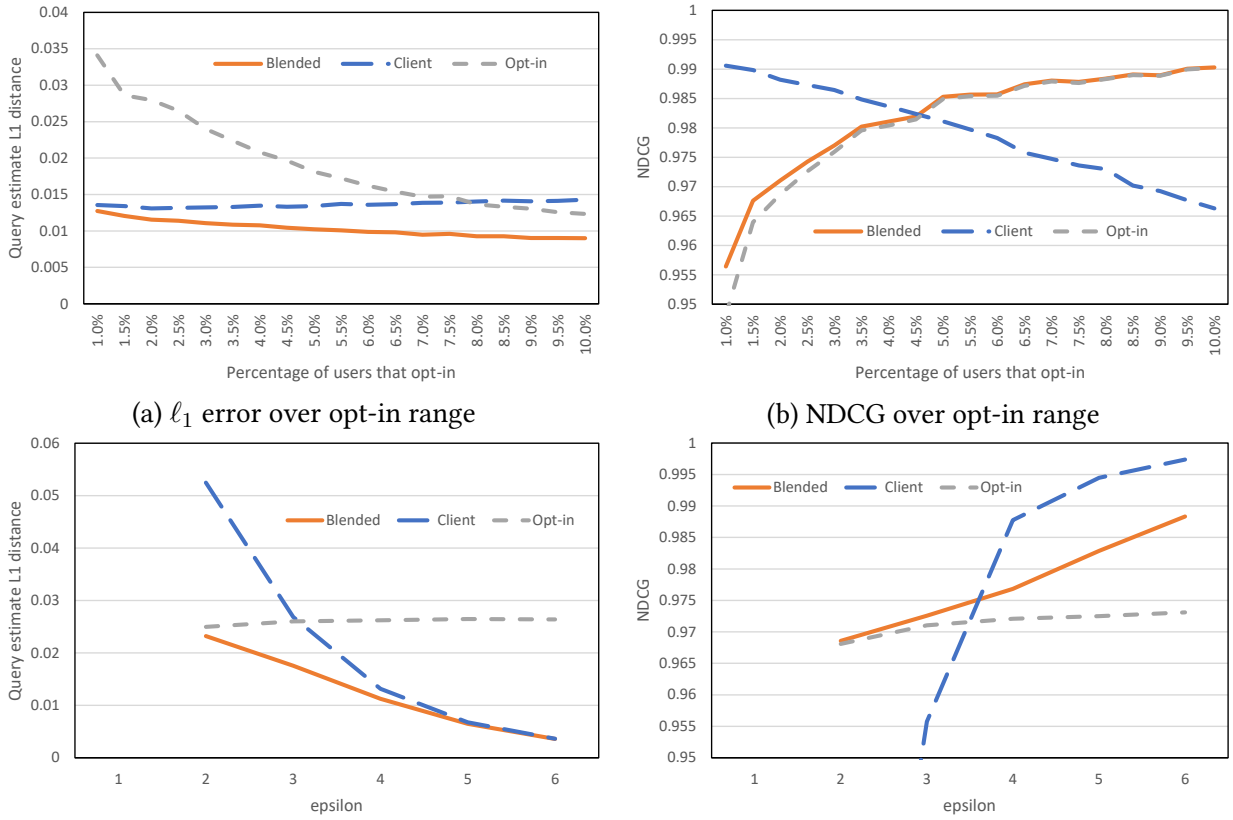
(a) AOL (b) Yandex

Figure 2.8: BLENDER's NDCG on the AOL and Yandex datasets at various head list sizes across a range of ϵ values.

Figure 2.8 shows BLENDER's NDCG on both datasets at various head list sizes and across a range of ϵ values. There is a clear trend toward higher NDCG values for Yandex, which is not surprising given the sheer volume of data. For the Yandex dataset, even with ϵ as low as 1, BLENDER still achieves NDCG values of 95% and above for all but the two largest head list sizes. For those two desired head list sizes, BLENDER necessitates a larger ϵ in order to discover the full head list from the opt-in users.

Each group's effect on BLENDER's final result. Thus far, we have determined that BLENDER is capable of achieving high-utility results. However, it is unclear how each group's estimates are contributing to BLENDER's final result. Specifically, we now address the question of whether the small number of samples with low noise from the opt-in group dominates (or is dominated by) the large number of samples with high noise from the client group.

Targeting a head list of size 100 on the Yandex dataset, we examine this question in Figure 2.9 for a range of opt-in percentages and ϵ values. These graphs show a complex relationship between the two groups' utility with regards to the final blended result. In all cases, the blended result is better than the worse of either the opt-in or client results. With regards to ℓ_1 error, the blended result is better than *both* groups' individual results when varying either the opt-in user percentage or the ϵ value.



(a) ℓ_1 error over opt-in range

(b) NDCG over opt-in range

(c) ℓ_1 error over ϵ range

(d) NDCG results over ϵ range

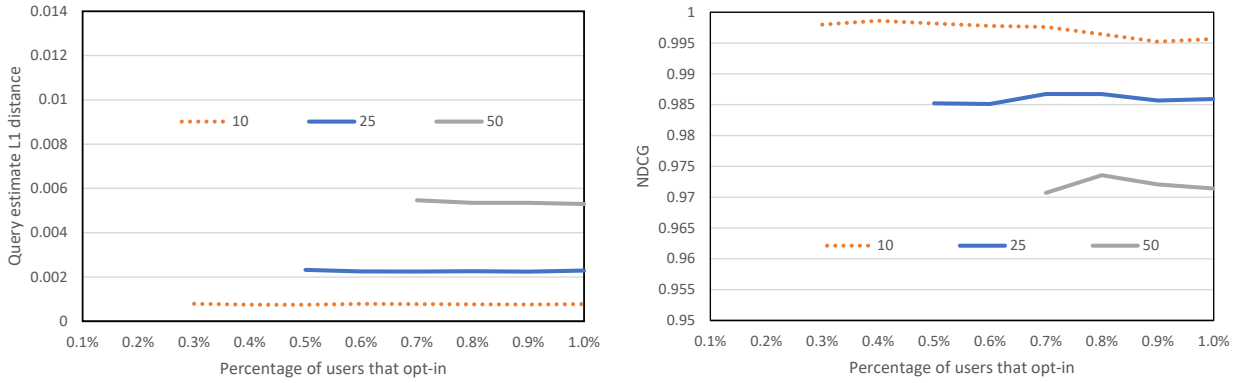
Figure 2.9: BLENDER's ℓ_1 error and NDCG results broken out between the different groups' results on the Yandex dataset with head list size 100 across a range of opt-in percentages (a,b) and a range of ϵ values at 3% opt-in (c,d).

When increasing the opt-in user percentage, the two groups' results behave as expected; the opt-in group's results improve as it gains more users, and the client group's results gradually deteriorate as it loses users. Interestingly, Figures 2.9a and b show that the ℓ_1 error of the client group's query estimates deteriorate quite slowly as their group size decreases, whereas their NDCG results deteriorate more quickly. To understand this behavior, first observe that there are significantly fewer queries (of which ℓ_1 measures the utility) than there are query/URL pairs (of which NDCG measures the utility). Also note that the utility of the generalized Randomized Response component of `LocalReport` degrades as the set of items under consideration increases. Taken together, these two facts explain the difference in the deterioration rates of the client group's utility between Figure 2.9a and b.

The NDCG of the blended result mainly tracks the NDCG of the opt-in group's results even in the case where the client result is clearly better (from 1% up to 3%). This would support the idea that the opt-in estimates may be dominating the client estimates during the blending process. However, this trend does not appear to hold when increasing ϵ, as the blended estimate's utility rapidly improves alongside the client estimate's utility, while the opt-in estimate's utility remains relatively flat. Interestingly, as ϵ is increased, the opt-in estimate's ℓ_1 error remains relatively constant and its NDCG only slightly improves. This is caused by the large amount of noise that is inherent in the opt-in group due to its relatively small size; i.e., a 3% sized opt-in group induces a certain level of sampling error such that the noise introduced for privacy is negligible by comparison.

The takeaway is that there is no single group that clearly dominates in its contribution to the final blended result. In fact, both groups appear to contribute across the ranges of parameters considered.

Utility impact of a tiny opt-in group. In the real world, it may be the case that a 5% or even a 3% sized opt-in group is still too large to be considered feasible. As mentioned in the above evaluation of query discovery and estimation, `BLENDER`'s utility is generally high *conditioned on*



(a) ℓ_1 results

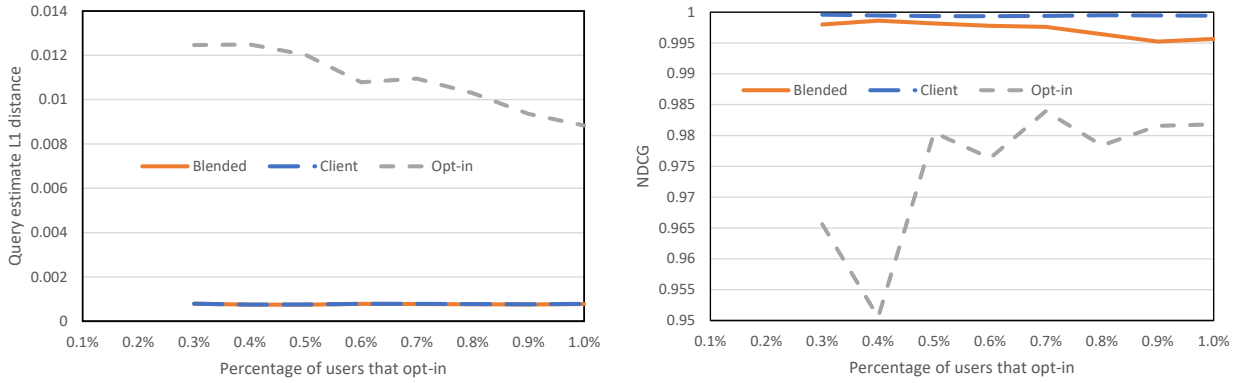
(b) NDCG results

Figure 2.10: BLENDER's ℓ_1 error and NDCG on the Yandex dataset at various head list sizes across a range of tiny opt-in percentages.

the desired head list size being achieved. When the opt-in group becomes too small, it becomes a significantly greater challenge for BLENDER to achieve large head list sizes. For the head list sizes that BLENDER *can* achieve at smaller opt-in percentages, what sort of utility results can we expect from BLENDER? We answer this question here.

Figure 2.10 shows BLENDER's utility on the Yandex dataset targeting smaller head list sizes across a range of opt-in group sizes from 0.1% up to 1%. These results confirm our previous conclusion that once BLENDER can attain a particular head list, it becomes fairly easy for BLENDER to achieve high utility probability estimates.

At these tiny opt-in percentages, with 95% of the opt-in group being assigned to head list discovery, only 0.005% to 0.05% of the users' data are used to estimate the probabilities under the TCM. In this setting, one may question the extent to which the opt-in users are contributing to the high-utility blended results. Figure 2.11 shows the ℓ_1 error and NDCG values for the opt-in group's results, the client group's, and the final blended results across these tiny opt-in sizes for a head list of size 10 on the Yandex dataset. As suspected, the estimates from the opt-in group have much lower utility relative to the client group. BLENDER's blending stage is able to automatically take advantage of the opt-in group's high variance results (stemming from the tiny number of samples provided by this group to estimate the record probabilities) and weigh the blending much more heavily towards the client group's results.



(a) ℓ_1 results

(b) NDCG results

Figure 2.11: BLENDER's ℓ_1 error and NDCG statistics broken out per group on the Yandex dataset at head list size 10 across a range of tiny opt-in percentages.

2.3 Mean Estimation

In the prior section on heavy hitter discovery and estimation, we gained invaluable insights into the hybrid model. However, the complexity of both the problem (having two stages, first discovering the heavy hitters and then estimating their frequencies) as well as the primary utility metric (NDCG) necessitated a fully empirical analysis of our designed hybrid mechanism. We gain deeper insights into the power of the hybrid model by analytically studying the more focused problem of mean estimation. Inspired by the frequency estimation portion of the heavy hitter problem, the central problem that we consider in this section of the thesis is:

> *How can we design a high utility hybrid mechanism for the problem of mean estimation?*

Estimating the mean μ of a distribution $\mathcal{D}$ from a dataset is a foundational problem in statistics that has been considered in a variety of different settings. The particular setting that we study is: estimating the mean μ of a distribution $\mathcal{D}$ with bounded support $[0, m]$ from a dataset D of n users, where each point x_i in the dataset is the data of a single user drawn i.i.d. from $\mathcal{D}$. This setting is well studied in statistics and in the DP literature, due to its prevalence as a fundamental building block in solutions to more complex tasks. As a result, a large number of DP mechanisms

in both the TCM and LM have been designed to address this problem [Dwo+06b; Dwo+06a; GRS12; CKS20], including two of the most widely used mechanisms in practice which we have already introduced in Section 1.1.2: the Laplace and Gaussian mechanisms.

Hybrid Model Mean Estimation

We extend this foundational mean estimation problem into the hybrid setting, taking into consideration that users' different trust model preferences may be correlated with their underlying behavior. Such correlation would violate the above assumption that all users' data are drawn i.i.d. from $\mathcal{D}$. To accommodate this potential distributional difference between the two groups, we generalize the mean estimation problem as follows:

Let c be the fraction of users who opted in to the TCM, and let $\mathcal{D}_T$ and $\mathcal{D}_L$ be the distributions that the TCM and LM users' data are drawn i.i.d. from, respectively. The input dataset is then decomposed as $D = D_T \cup D_L$, where D_T is the data of the cn TCM users drawn i.i.d. from $\mathcal{D}_T$, and D_L is the data of the $(1-c)n$ LM users drawn i.i.d. from $\mathcal{D}_L$. We seek to estimate the joint mean of the two groups; i.e., the mean of their mixture distribution $\mathcal{D} = c\mathcal{D}_T + (1-c)\mathcal{D}_L$[9]. This hybrid mean setting is illustrated in Figure 2.12. Thus, the differentially private mechanisms that we design and analyze in this section are statistical estimators, and we refer to them as such throughout the remainder of the section.

Because this section is notationally dense, Table 2.4 serves as a reference table for the various symbols that we define.

Our Contributions

We initiate our study of the mean estimation problem under the hybrid model by concretely defining how we measure utility. Along with this, we define baseline estimators in the TCM and LM, and analytically characterize their utilities in order to subsequently contextualize the

[9]This generalization reduces to the basic mean estimation problem when the two groups have the same underlying distribution ($\mathcal{D}_T = \mathcal{D}_L$).

Symbol	Usage
ϵ, δ	Differential privacy parameters.
n	Total number of users.
c	Fraction of users who opt in to TCM.
T, L	Set of users who opted in to TCM and set of users who are using LM, respectively.
$\mathcal{D}$	Mixture distribution of both groups' data.
μ, σ^2, m	Mean, variance, and maximum support of $\mathcal{D}$.
$\mathcal{D}_T$	Distribution of TCM groups' data.
μ_T, σ_T^2, m_T	Mean, variance, and maximum support of $\mathcal{D}_T$.
$\mathcal{D}_L$	Distribution of LM groups' data.
μ_L, σ_L^2, m_L	Mean, variance, and maximum support of $\mathcal{D}_L$.
x_i	User i's private data drawn i.i.d. from its group's distribution.
$\hat{\mu}, \hat{\mu}_T, \hat{\mu}_L$	Empirical mean estimates with all users, with only the TCM users, and with only the LM users, respectively.
$\mathcal{E}$	MSE of an estimator with respect to $\hat{\mu}$.
$\tilde{\mu}_T, \mathcal{E}_T$	`TCM-Only` estimator and its MSE.
$\tilde{\mu}_F, \mathcal{E}_F$	`Full-LM` estimator and its MSE.
$\tilde{\mu}_L, \mathcal{E}_L$	`LM-Only` estimator and its MSE.
$\tilde{\mu}_H(w), \mathcal{E}_H(w)$	Hybrid estimator with weight w and its MSE.
Y_T, s_T^2	`TCM-Only` estimator's privacy random variable and its variance.
$Y_{L,i}, s_L^2$	User i's local privacy random variable and its variance.
n_{crit}, c_{crit}	n and c values that partition where $\mathcal{E}_T \leq \mathcal{E}_F$.
$R(\mathcal{E}), r(\mathcal{E})$	Relative improvement of estimator with MSE $\mathcal{E}$ over the best and worst non-hybrid baselines, respectively.

Table 2.4: Comprehensive list of notation for mean estimation in the hybrid model.

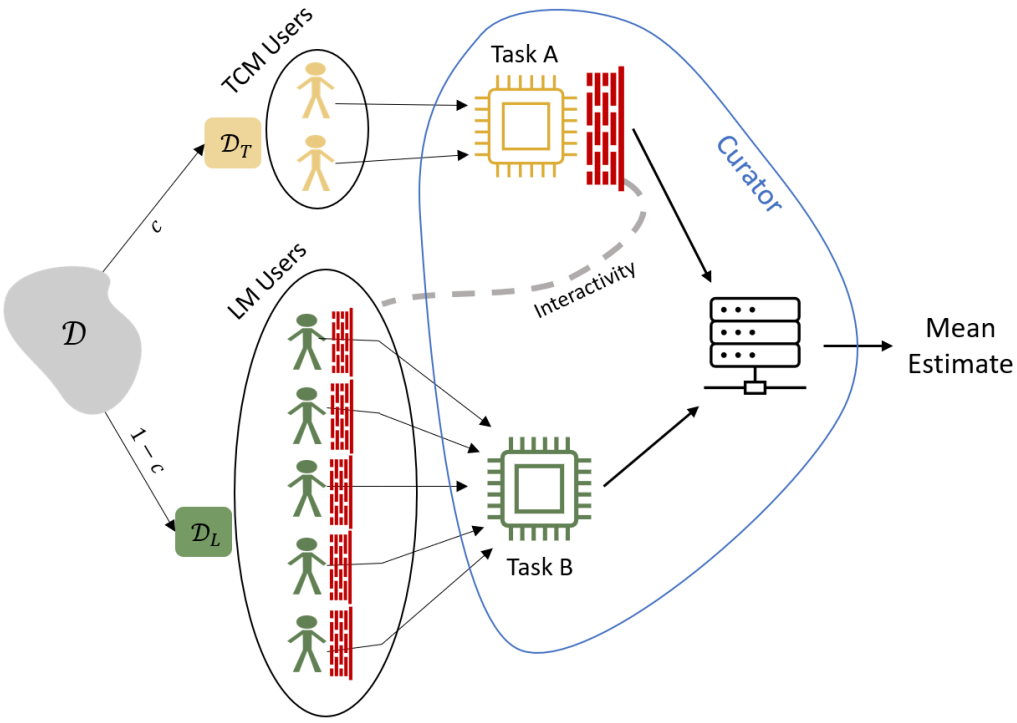


Figure 2.12: Overview of the mean estimation problem through the lens of our hybrid differential privacy model.

utility of any hybrid estimator. We then define a family of hybrid estimators based not on a specialization-based approach like with BLENDER, but rather by utilizing a direct-combination approach (Section 2.1). Specifically, estimators in this family independently compute private estimates from the TCM and LM groups, and then straightforwardly combine them in a weighted manner.

Analyzing the hybrid estimator family enables us to address the questions posed in Section 2.1, particularly on understanding the utility of hybrid mechanisms as well as on how properties of the two groups of users affect hybrid mechanisms' privacy and utility. In this analysis, we first analytically determine the utility of the hybrid estimators, and compare it against baseline estimators in the classic trust models. We then quantify how the hybrid estimators' utilities are affected when the two groups' data are drawn from different distributions. Finally, we analyze how a mechanism's privacy and utility are impacted by the manner in which the groups of users interact.

For our first contribution, by deriving and analyzing concrete estimators from the hybrid estimator family, we determine that the hybrid estimator family can attain high utility. We achieve this by deriving specific estimators from our hybrid family in two statistical settings: when the two groups' data are drawn from the same distribution $\mathcal{D}$ and its variance is known to the estimator, and when the two groups' data are drawn from the same distribution but its variance is not known to the estimator. We refer to the setting where the two groups' data are drawn from the same distribution as the *homogeneous* setting.

- *Homogeneous, known-variance:* We utilize the knowledge of the variance to derive a hybrid estimator that achieves an optimal MSE, and refer to the corresponding estimator as the known-variance hybrid estimator. Computing the known-variance hybrid estimator's relative improvement against the baseline estimators, we find that it *provably* outperforms both baseline estimators *simultaneously in all parameter regimes*. Moreover, we bound this estimator's maximum relative improvement in realistic parameter regimes and find that its improvement peaks at a factor of approximately 2.3x over the best baseline.

- *Homogeneous, unknown-variance:* We derive a hybrid estimator that heuristically attempts to minimize the MSE, and refer to the corresponding estimator as the unknown-variance hybrid estimator. Computing the unknown-variance hybrid estimator's relative improvement against the baseline estimators, we find that it outperforms both baseline estimators simultaneously in *some* parameter regimes, and we analytically characterize these parameter regimes. In the parameter regimes where the unknown-variance hybrid estimator does not outperform both simultaneously, we show that it always outperforms one of the baselines. Moreover, we find that it often achieves utility comparable to the known-variance hybrid estimator in realistic parameter regimes.

We additionally evaluate both hybrid estimators' utilities in practice by simulating them on realistic distributions and parameters, finding that they both typically achieve a constant factor improvement over the baselines.

Deriving analytical utility results enables us to rigorously address the question of how the hybrid estimators' utilities are affected when users' data are drawn from distributions dependent on their trust model preference. We refer to this as the *heterogeneous* setting. The derived utility expressions reveal that the means and variances are the only aspects of the TCM and LM groups' distributions that affect the utility of both the hybrid and baseline estimators. Thus, to examine the utility impact of the two groups' distributions diverging, we separately consider the scenarios where the means diverge and where the variances diverge. We find that when the distributions' means diverge, our hybrid estimators become increasingly biased, and utility drops sharply relative to the best baseline. However, when the distributions' variances diverge, we find that our hybrid estimators maintain high utility. This implies that in practice, if the underlying means of the two groups are expected to be approximately the same, but their behavior is otherwise different (manifesting as different variances), then we can expect our hybrid estimators to achieve high utility.

Finally, we demonstrate how hybrid DP mechanisms can be designed from non-hybrid DP mechanisms by using our hybrid estimators as a drop-in mean estimation primitive. To accomplish this, we convert a classic DP mechanism for the K-means problem in the TCM into a mechanism in the hybrid model by using our hybrid estimator. We then empirically evaluate the effectiveness of this new hybrid mechanism, finding that it achieves high utility.

2.3.1 Measuring Utility

We initiate our study of mean estimation in the hybrid model by defining, from an absolute perspective, how we measure the utility of any estimator. We then define what the baseline estimators are in the classic trust models which we use to contextualize the utility of hybrid estimators that we design. For both baseline estimators, we analytically characterize their utility. Finally, we specify how we concretely measure a hybrid estimator's utility against the baseline estimators.

2.3.1.1 Absolute Measure of Utility

Our goal is to design accurate estimators of the mean μ of the mixture distribution $\mathcal{D}$. For measuring utility of any private estimator, we use the non-private empirical mean estimator as a benchmark.

Definition 2.3.1. The non-private empirical mean estimator is:

$$\hat{\mu} = \frac{1}{n}\sum_{i\in[n]} x_i = c\hat{\mu}_T + (1-c)\hat{\mu}_L.$$

Comparing a private estimator against the non-private empirical estimator in this way reflects our interest in the excess error introduced by the privatization scheme, beyond the inherent error induced by a finite sample size [Dwo+06b; Dwo+06a; GRS12; BW18; CKS20].

To concretely measure the utility of any private mean estimator $\tilde{\mu}$ in the classic trust models or the hybrid trust model, we measure its error with respect to the non-private empirical benchmark estimator in terms of its mean squared error (MSE).

Definition 2.3.2. The MSE between a private estimator $\tilde{\mu}$ and the non-private empirical mean $\hat{\mu}$ is:

$$\mathcal{E} = \text{MSE}(\tilde{\mu}, \hat{\mu}) = \mathbb{E}[(\tilde{\mu} - \hat{\mu})^2].$$

For brevity, since the non-private empirical benchmark estimator is used to measure the MSEs of all private estimators in this section, we simply refer to it as the MSE of the private estimator.

2.3.1.2 Baseline Estimators

We now motivate how we design all private estimators, including the baseline estimators. We then formally define baseline estimators in the TCM and LM. For each baseline estimator, we analytically characterize its utility against the non-private empirical benchmark estimator.

To design any private estimator, hybrid or otherwise, we leverage the broad and powerful class of "additive noise mechanisms" (Section 1.1.2). In the classic trust models, the state-of-the-art mechanisms for DP mean estimation fall under this class. Additive noise mechanisms ensure DP for real-valued functions by adding randomness directly to the function's output, where the randomness is drawn from a carefully constructed distribution (typically with mean 0). For mean estimation across a variety of distributional settings in both the TCM and LM, several specific additive noise mechanisms have been proven optimal or near-optimal. Examples include the Geometric mechanism [GRS12], the Staircase mechanism [KOV14; GV15], and the Truncated Laplacian mechanism [Gen+20]. The class of additive noise mechanisms also includes the most widely used mechanisms for the basic mean estimation problem: the Laplace and Gaussian mechanisms [Dwo+06b; Dwo+06a]. Despite these two mechanisms' sub-optimality, their widespread use stems from their simplicity and generality in conjunction with their high utility in practical settings.

To contextualize the utility of any hybrid estimator, we first utilize this class to generically define baseline estimators in the classic trust models. The TCM baseline estimator, applied only to the TCM users, is referred to as the `TCM-Only` estimator. It is formally defined as follows.

Definition 2.3.3. The `TCM-Only` estimator is:

$$\tilde{\mu}_T = \frac{1}{cn} \sum_{i \in T} x_i + Y_T,$$

where Y_T is a random variable with 0 mean and s_T^2 variance chosen such that DP is satisfied for all TCM users.

Lemma 2.3.4. $\tilde{\mu}_T$ has MSE:

$$\mathcal{E}_T = \frac{(1-c)^2}{cn}\sigma_T^2 + \frac{1-c}{n}\sigma_L^2 + s_T^2 + (\mu_T - \mu)^2.$$

Proof.

$$\begin{aligned}
\mathcal{E}_T &= \mathbb{E}[(\tilde{\mu}_T - \hat{\mu})^2] \\
&= \mathbb{V}[\tilde{\mu}_T - \hat{\mu}] + \mathbb{E}[\tilde{\mu}_T - \hat{\mu}]^2 \\
&= \mathbb{V}\left[\frac{1}{cn}\sum_{i\in T} x_i + Y_T - \frac{1}{n}\sum_{i\in[n]} x_i\right] + (\mu_T - \mu)^2 \\
&= \mathbb{V}\left[\frac{1}{cn}\sum_{i\in T} x_i - \frac{1}{n}\sum_{i\in T} x_i - \frac{1}{n}\sum_{i\in L} x_i + Y_T\right] + (\mu_T - \mu)^2 \\
&= \frac{(1-c)^2}{cn}\sigma_T^2 + \frac{1-c}{n}\sigma_L^2 + s_T^2 + (\mu_T - \mu)^2.
\end{aligned}$$

□

This error has three components, $\frac{(1-c)^2}{cn}\sigma_T^2 + \frac{1-c}{n}\sigma_L^2$, s_T^2, and $(\mu_T - \mu)^2$. The first component is the error induced by subsampling only the TCM users – we refer to this as the *excess sampling error*. The second component is the error due to DP – we refer to this as the *privacy error*. The third component is the *bias error* induced by the groups' means differing.

We now define the LM baseline estimator. Since the LM does not require trust in the curator, the estimator in this model can utilize the *full* set of users' data. We refer to this baseline estimator as the `Full-LM` estimator, and define it formally as follows.

Definition 2.3.5. Suppose each user i privately reports their data as $x_i + Y_{L,i}$, where $Y_{L,i}$ is a random variable with 0 mean and s_L^2 variance chosen such that DP is satisfied for user i. The `Full-LM` estimator is then:

$$\tilde{\mu}_F = \frac{1}{n}\sum_{i\in[n]}(x_i + Y_{L,i}).$$

Lemma 2.3.6. $\tilde{\mu}_F$ has MSE:

$$\mathcal{E}_F = \frac{s_L^2}{n}.$$

Proof.

$$\begin{aligned}
\mathcal{E}_F &= \mathbb{E}[(\tilde{\mu}_F - \hat{\mu})^2] \\
&= \mathbb{V}[\tilde{\mu}_F - \hat{\mu}] + \underbrace{\mathbb{E}[\tilde{\mu}_F - \hat{\mu}]^2}_{0} \\
&= \mathbb{V}\left[\frac{1}{n}\sum_{i\in[n]}(x_i + Y_{L,i}) - \frac{1}{n}\sum_{i\in[n]} x_i\right] \\
&= \frac{s_L^2}{n}.
\end{aligned}$$

□

This error only consists of a single simple component: the privacy error. Since the entire dataset is used, there is no excess sampling error and no bias error.

2.3.1.3 Measuring Utility Against Both Baselines

While we measure an estimator's absolute utility using MSE, we are primarily interested in a hybrid estimator's *relative* utility compared to the baseline estimators. Towards this, we first define how we measure a hybrid estimator's utility against the *best* of the two baseline estimators. Since no baseline estimator is "best" across all parameter regimes, we precisely characterize the parameter regimes where each baseline estimator is dominant. We then motivate and define a second, weaker measure of a hybrid estimator's utility against the *worst* of the two baseline estimators. While we primarily use the stronger of the two utility measures in this section, the weaker utility measure comes in handy for showing that our hybrid estimators never perform worse than both baseline estimators simultaneously.

For any hybrid estimator with MSE $\mathcal{E}$, we seek to compare its utility against the MSEs of the baseline estimators. Specifically, we define the hybrid estimator's *relative improvement* over the best of the two baseline estimators as its MSE improvement factor both baselines' MSEs. Formally, this is:

Definition 2.3.7. The relative improvement of an estimator with MSE $\mathcal{E}$ over the best baseline estimator is:

$$R(\mathcal{E}) = \frac{\min\{\mathcal{E}_T, \mathcal{E}_F\}}{\mathcal{E}}.$$

Thus, for a given setting of parameters, $R(\mathcal{E}) > 1$ implies that the hybrid estimator has higher utility than both baseline estimators simultaneously.

This measure of relative improvement can be rewritten to explicitly consider the parameter regimes (i.e., the ranges of parameters μ, σ^2, n, c, m, etc.) where each of the baseline estimators achieves the $\min\{\cdot\}$. That is, we determine the parameter configurations in which the `TCM-Only` estimator is better/worse than the `Full-LM` estimator. Intuitively, we expect that when very few users opt in to the TCM, the `TCM-Only` estimator's large excess sampling error will overshadow its smaller privacy error (relative to the `Full-LM` estimator's privacy error). This intuition is made precise by considering "critical values" of c and n that determine the regimes where each of the estimators yields better utility.

Lemma 2.3.8. Let n_{crit} and c_{crit} be defined as follows.

$$n_{crit} = \frac{cs_L^2 + (1-c)((1-c)\sigma_T^2 - c\sigma_L^2)}{c((\mu_T - \mu)^2 + s_T^2)}$$

$$c_{crit} = \begin{cases} \frac{\sigma_L^2}{\sigma_L^2 + s_L^2}, & \sigma_T = \sigma_L, \\ \frac{2\sigma_T^2 - \sigma_L^2 + s_L^2 - \sqrt{(\sigma_L^2 - s_L^2)^2 + 4s_L^2\sigma_T^2}}{2(\sigma_T^2 - \sigma_L^2)}, & \sigma_T \neq \sigma_L. \end{cases}$$

We have that $\mathcal{E}_T \leq \mathcal{E}_F$ if and only if $c > c_{crit}$ and $n \leq n_{crit}$.

Proof. Directly reduce the system of inequalities constructed by $\mathcal{E}_T \leq \mathcal{E}_F$ in conjunction with the regions given by the valid parameter ranges. This immediately yields the result. □

This characterization allows us to partition the definition of relative improvement into the behavior of each baseline estimator, rewritten as follows.

Definition 2.3.9. The relative improvement of an estimator with MSE $\mathcal{E}$ over the best baseline estimator is:

$$R(\mathcal{E}) = \frac{1}{\mathcal{E}} \cdot \begin{cases} \mathcal{E}_T & \text{if } c > c_{crit} \text{ and } n \leq n_{crit} \\ \mathcal{E}_L & \text{otherwise.} \end{cases}$$

The behavior of these two cases further depends on the privacy mechanism used, as that dictates s_T and s_L. For example, when using the ϵ-DP Laplace mechanism in the homogeneous setting where both group means are μ and variances are σ^2, these definitions of critical values and relative improvement become the following.

Lemma 2.3.10. Adding ϵ-DP Laplace noise for privacy, define $c_{crit} = \frac{\epsilon^2\sigma^2}{2m^2+\epsilon^2\sigma^2}$ and $n_{crit} = \frac{2m^2}{c(2cm^2-(1-c)\epsilon^2\sigma^2)}$. We have that $\mathcal{E}_T \leq \mathcal{E}_F$ if and only if $c > c_{crit}$ and $n \geq n_{crit}$.

Definition 2.3.11. Adding ϵ-DP Laplace noise for privacy, the relative improvement of an estimator with MSE $\mathcal{E}$ over the best baseline estimator is:

$$R(\mathcal{E}) = \frac{1}{\mathcal{E}} \cdot \begin{cases} \frac{1-c}{cn}\sigma^2 + \frac{2m^2}{c^2n^2\epsilon^2} & \text{if } c > c_{crit} \text{ and } n \geq n_{crit} \\ \frac{2m^2}{n\epsilon^2} & \text{otherwise.} \end{cases}$$

Thus, once the fraction of users opting in to the TCM is large enough, the `TCM-Only` estimator has better MSE than the `Full-LM` estimator. In all other regimes, the `Full-LM` estimator has better MSE than the `TCM-Only` estimator. This aligns with our intuition.

Ideally, hybrid estimators would have $R(\mathcal{E}) \geq 1$ for all parameters. If the parameter regions can be computed where each baseline estimator has the best MSE, then a hybrid estimator can be designed to use this knowledge to trivially ensure $R(\mathcal{E}) = 1$. However, depending on the setting (such as when variance is unknown), determining these regions precisely may not be feasible. In these cases, we want to at least ensure that the hybrid estimator is never performing worse than both baselines, and do so by defining the following measure of relative improvement.

Definition 2.3.12. The relative improvement of an estimator $\mathcal{E}$ over the worst baseline estimator is:

$$r(\mathcal{E}) = \frac{\max\{\mathcal{E}_T, \mathcal{E}_F\}}{\mathcal{E}}.$$

Our characterization of the critical values in Lemma 2.3.8 enables $r(\varepsilon)$ to be rewritten as follows.

Definition 2.3.13. The relative improvement of an estimator with MSE $\mathcal{E}$ over the worst baseline estimator is:

$$r(\mathcal{E}) = \frac{1}{\mathcal{E}} \cdot \begin{cases} \frac{s_L^2}{n} & \text{if } c > c_{crit} \text{ and } n \leq n_{crit} \\ \frac{1-c}{cn}\sigma^2 + s_T^2 & \text{otherwise.} \end{cases}$$

2.3.2 Hybrid Estimator Family

Depending on the setting, designing a hybrid estimator that outperforms *at least one* of these baselines in *all* parameter regimes can be trivial. Similarly, designing a hybrid estimator that outperforms *both* baselines in *some* regimes can be trivial. One challenge solved in this section is designing a hybrid estimator that provably outperforms *both* baselines across *all* regimes. To accomplish this, we design a family of estimators within the hybrid model.

Because this problem is not intuitively decomposable into distinct tasks, we are unable to utilize a specialization-based approach (Section 2.1) to designing a hybrid estimator. Instead, we adopt a direct-combination approach to design a family of hybrid mean estimators $\tilde{\mu}_H(w)$ parameterized by $w \in [0, 1]$. Informally, estimators in this family independently compute privatized estimates for the TCM and LM groups under their respective trust models, then output a convex combination of the estimates weighted by w.

$\tilde{\mu}_H(w)$ computes the privatized estimate for the TCM group via the `TCM-Only` baseline estimator $\tilde{\mu}_T$. To compute the privatized estimate for the LM group, $\tilde{\mu}_H(w)$ utilizes a new private estimator in the LM model, referred to as the `LM-Only` estimator. The `LM-Only` estimator $\tilde{\mu}_L$ is

nearly identical to the baseline `Full-LM` estimator $\tilde{\mu}_F$, except that $\tilde{\mu}_L$ uses only the data of the LM users (rather than using the data of *all* the users). It is formally defined as follows.

Definition 2.3.14. The `LM-Only` estimator is:

$$\tilde{\mu}_L = \frac{1}{(1-c)n} \sum_{i \in L} (x_i + Y_{L,i}),$$

where, for each $i \in L$, $Y_{L,i}$ is a random variable with 0 mean and s_L^2 variance chosen such that DP is satisfied for user i.

Lemma 2.3.15. $\tilde{\mu}_L$ has expected squared error:

$$\mathcal{E}_L = \frac{c^2}{(1-c)n}\sigma_L^2 + \frac{c}{n}\sigma_T^2 + \frac{1}{(1-c)n}s_L^2 + (\mu_L - \mu)^2.$$

Proof.

$$\begin{aligned}
\mathcal{E}_L &= \mathbb{E}[(\tilde{\mu}_L - \hat{\mu})^2] \\
&= \mathbb{V}[\tilde{\mu}_L - \hat{\mu}] + \mathbb{E}[\tilde{\mu}_L - \hat{\mu}]^2 \\
&= \mathbb{V}\left[\frac{1}{(1-c)n}\sum_{i \in L}(x_i + Y_{L,i}) - \frac{1}{n}\sum_{i \in [n]} x_i\right] + (\mu_L - \mu)^2 \\
&= \mathbb{V}\left[\frac{c}{(1-c)n}\sum_{i \in L} x_i - \frac{1}{n}\sum_{i \in T} x_i + \frac{1}{(1-c)n}\sum_{i \in L} Y_{L,i}\right] + (\mu_L - \mu)^2 \\
&= \frac{c^2}{(1-c)n}\sigma_L^2 + \frac{c}{n}\sigma_T^2 + \frac{1}{(1-c)n}s_L^2 + (\mu_L - \mu)^2.
\end{aligned}$$

□

In addition to the privacy error, due to the lack of TCM users, this estimator also has excess sampling error as well as bias error. Since it has strictly greater error than the `Full-LM` estimator, it is not used as one of the baseline estimators.

With the independent TCM and LM components of the hybrid estimator family defined, we now formally define $\tilde{\mu}_H(w)$ and derive its MSE.

Definition 2.3.16. The hybrid estimator family, parameterized by $w \in [0, 1]$, is:

$$\tilde{\mu}_H(w) = w\tilde{\mu}_T + (1-w)\tilde{\mu}_L.$$

Lemma 2.3.17. $\tilde{\mu}_H(w)$ has expected squared error:

$$\mathcal{E}_H(w) = \frac{(w-c)^2}{cn}\sigma_T^2 + \frac{(w-c)^2}{(1-c)n}\sigma_L^2 + w^2 s_T^2 + \frac{(1-w)^2}{(1-c)n}s_L^2 + (w\mu_T + (1-w)\mu_L - \mu)^2.$$

Proof.

$$\begin{aligned}
\mathcal{E}_H(w) &= \mathbb{E}[(\tilde{\mu}_H(w) - \hat{\mu})^2] \\
&= \mathbb{V}[\tilde{\mu}_H(w) - \hat{\mu}] + \mathbb{E}[\tilde{\mu}_H(w) - \hat{\mu}]^2 \\
&= \mathbb{V}[w\tilde{\mu}_T + (1-w)\tilde{\mu}_L - \hat{\mu}] + (w\mu_T + (1-w)\mu_L - \mu)^2 \\
&= \mathbb{V}[w\tilde{\mu}_T - c\hat{\mu}_T + (1-w)\tilde{\mu}_L - (1-c)\hat{\mu}_L] + (w\mu_T + (1-w)\mu_L - \mu)^2 \\
&= \frac{(w-c)^2}{cn}\sigma_T^2 + \frac{(w-c)^2}{(1-c)n}\sigma_L^2 + w^2 s_T^2 + \frac{(1-w)^2}{(1-c)n}s_L^2 + (w\mu_T + (1-w)\mu_L - \mu)^2.
\end{aligned}$$

□

Hybrid estimators in this family have all three types of error — excess sampling error, privacy error, and bias error — where the amounts of each error type depend on the weighting w.

2.3.3 Homogeneous, Known-Variance Setting

We now derive a concrete hybrid estimator in the homogeneous setting by carefully choosing a particular weighting for the hybrid estimator family from Definition 2.3.16. We then show, both theoretically and empirically, that the derived estimator always outperforms both baselines estimators.

To select a weighting for the hybrid estimator family, we restrict our focus to the homogeneous setting, where both groups' means are the same ($\mu = \mu_T = \mu_L$) and variances are the same ($\sigma^2 = \sigma_T^2 = \sigma_L^2$). Beyond simplifying the expressions that we analyze, the homogeneous

setting eliminates bias error from our defined estimators, which removes any dependence on μ from the derived error expressions. This is important, since the curator's goal is to learn μ from the data; thus, no particular knowledge of μ is assumed. Therefore, in the homogeneous setting, a weighting can be chosen by analyzing the hybrid estimator's derived error expressions without needing any knowledge of μ. However, there is still excess sampling error for the estimators in this setting — in other words, error expressions still depend on the data variance σ^2. Thus, in this portion of the thesis, we make the common assumption in statistical literature that σ^2 is known to the curator, and derive and analyze the optimal hybrid estimator from the convex family. In the subsequent portion of the thesis, we lift this assumption.

2.3.3.1 KVH Estimator

We now derive and analyze the "known-variance hybrid" (KVH) estimator by computing the optimal weighting w^* that minimizes $\mathcal{E}_H(w)$. This can be analytically computed and directly implemented by the curator, since each term of $\mathcal{E}_H(w)$ is known in this setting.

Definition 2.3.18. The known-variance hybrid estimator in the homogeneous setting is:

$$\tilde{\mu}_{KVH} = w^*\tilde{\mu}_T + (1 - w^*)\tilde{\mu}_L,$$

where $w^* = \frac{c(\sigma^2+s_L^2)}{\sigma^2+c(ns_T^2(1-c)+s_L^2)}$ is obtained by minimizing $\mathcal{E}_H(w)$ with respect to w.

Lemma 2.3.19. $\tilde{\mu}_{KVH}$ has expected squared error:

$$\mathcal{E}_{KVH} = \frac{(w^* - c)^2}{c(1-c)n}\sigma^2 + w^{*2}s_T^2 + (1 - w^*)^2\frac{s_L^2}{(1-c)n}.$$

Although all users' data are used here, weighting the estimates by w^* induces excess sampling error $\frac{(w^*-c)^2}{c(1-c)n}\sigma^2$, and the privacy error $w^{*2}s_T^2 + \frac{(1-w^*)^2}{(1-c)n}s_L^2$ is the weighted combination of the groups' privacy errors.

Now we compute and analyze the relative improvement in MSE of the KVH estimator over the best MSE of the `TCM-Only` and `Full-LM` estimators.

Theorem 2.3.20. The relative improvement of $\tilde{\mu}_{KVH}$ over the better of $\tilde{\mu}_T$ and $\tilde{\mu}_F$ is:

$$R(\mathcal{E}_{KVH}) = \gamma \cdot \begin{cases} \frac{1-c}{cn}\sigma^2 + s_T^2 & \text{if } c > c_{crit} \text{ and } n \leq n_{crit} \\ \frac{s_L^2}{n} & \text{otherwise,} \end{cases}$$

where $\gamma = \frac{(1-c)\sigma^2 s_L^2 + cn(c\sigma^2 + s_L^2)s_T^2}{n(\sigma^2 + cs_L^2) + (1-c)cn^2 s_T^2}$ and c_{crit} and n_{crit} are as defined in Lemma 2.3.8.

Proof. Direct application of Lemmas 2.3.4, 2.3.6, and 2.3.19 to Definition 2.3.9. □

Algebraic analysis of this relative improvement reveals that $R(\mathcal{E}_{KVH}) > 1$ when the number of TCM users is less than s_L^2/s_T^2. For the standard 0-mean additive noise DP mechanisms, this condition is trivially satisfied. For instance, when adding noise from the Laplace mechanism to achieve ϵ-DP, we have that $s_L^2/s_T^2 = c^2n^2 \geq cn = |T|$. Moreover, although $R(\mathcal{E}_{KVH})$ is theoretically unbounded, using the ϵ-DP Laplace mechanism in the high-privacy regime ($\epsilon \leq 1$) enables a tight characterization of the maximum possible relative improvement.

Corollary 2.3.21. The maximum relative utility of $\tilde{\mu}_{KVH}$ when using the Laplace mechanism in the high-privacy regime is bounded as:

$$17/8 \leq \max_{\substack{\epsilon \leq 1 \\ c,n,m,\sigma}} R(\mathcal{E}_{KVH}) \leq 16/7.$$

Proof. For upper-bounding $R(\mathcal{E}_{KVH})$, we first note that Popoviciu's inequality [Pop35] states that a random variable bounded in $[a, b]$ has variance at most $(b-a)^2/4$. For our purposes, this ensures $\sigma^2 \leq m^2/4$.

For real-world use cases, it is realistic to constrain ϵ to the "high-privacy" regime of $\epsilon \leq 1$. Thus, with $\epsilon \leq 1$ and $\sigma^2 \leq m^2/4$, we have $0 \leq \epsilon^2\sigma^2/m^2 \leq 1/4$.

Letting $y = \epsilon^2\sigma^2/m^2$, we now upper-bound the improvement ratio as follows.

$$R(\mathcal{E}_{KVH}) \leq \frac{2(2-c)m^2}{2m^2 - (1-c)\epsilon^2\sigma^2} = \frac{2(2-c)}{2-(1-c)y} \leq 16/7,$$

where the final inequality stems from constrained maximization across $c \in [0,1]$ and $y \in [0, 1/4]$ (justified by Popoviciu's inequality).

A lower-bound is given by the following concrete instance. Let $m = 1$, $\epsilon = 1$, $\sigma^2 = 1/4$, and $c = \frac{1}{18}\left(1 + \sqrt{\frac{288+n}{n}}\right)$. Then, as $n \to \infty$, we have that $R(\mathcal{E}_{KVH})$ converges to $17/8$. □

2.3.3.2 Empirical Evaluation of $R(\mathcal{E}_{KVH})$

To better understand what improvements one can expect from $\tilde{\mu}_{KVH}$ in practical applications, we empirically evaluate $R(\mathcal{E}_{KVH})$ using the ϵ-DP Laplace mechanism in the context of various datasets. Note that although the hybrid estimator's performance is dependent on the data distribution only through σ, n, and m, we use datasets to realistically motivate these values.

In Figure 2.13, we use three synthetic datasets from the Beta(α, β) distribution: Beta$(10, 10)$, Beta$(1, 1)$, and Beta$(0.1, 0.1)$. These symmetric distributions are chosen to induce different σ values — low ($\sigma \approx 0.109$), medium ($\sigma \approx 0.289$), and high ($\sigma \approx 0.456$). For each distribution, $R(\mathcal{E}_{KVH})$ is plotted across $n \in [10^3, 10^5]$, $c \in \{0.5\%, 5\%\}$, and $\epsilon \in \{0.1, 1\}$. Since the Beta distributions are supported on the interval $[0, 1]$, we let $m = 1$. Figures 2.13b,c,d show that in these settings, $R(\mathcal{E}_{KVH})$ is lower-bounded by 1 and is never much larger than 2, matching our mathematical analysis. Observe that the "peaking" behavior of some curves is caused by the n_{crit} and c_{crit} values being surpassed, which corresponds to the TCM group's data beginning to outperform the LM group's data in terms of MSE. The curves that do not appear to peak either have trivially surpassed the critical values (i.e., $n_{crit} < 1$ with $c > c_{crit}$) or have $c < c_{crit}$. Importantly, they do not change behavior at some n not shown in the figures.

In Figure 2.14, we use a real-world dataset of salaries of $n = 252,540$ employees in the University of California system in 2010 [Cal]. This dataset was chosen due to its relatively high asymmetry, with a maximum salary of $m \approx 2,349,033$ and standard deviation of $\sigma \approx 53,254$

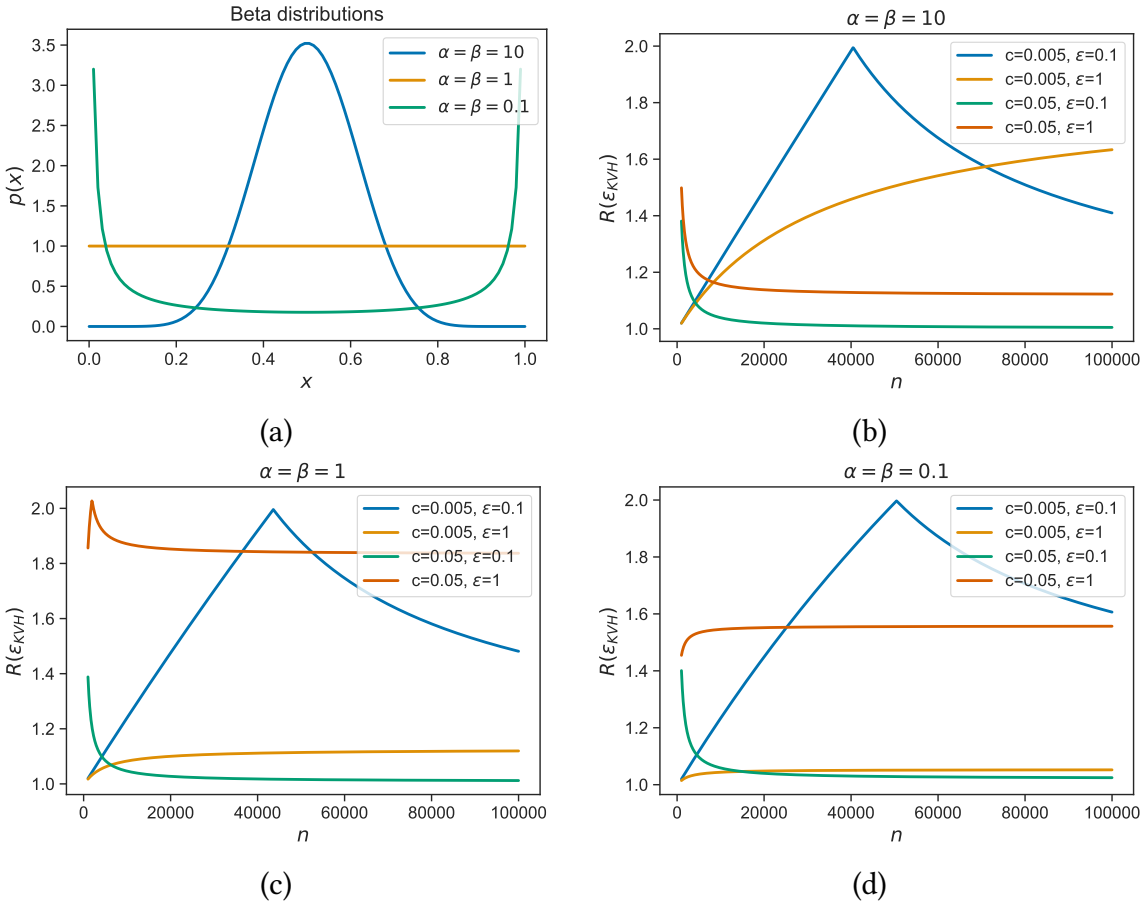
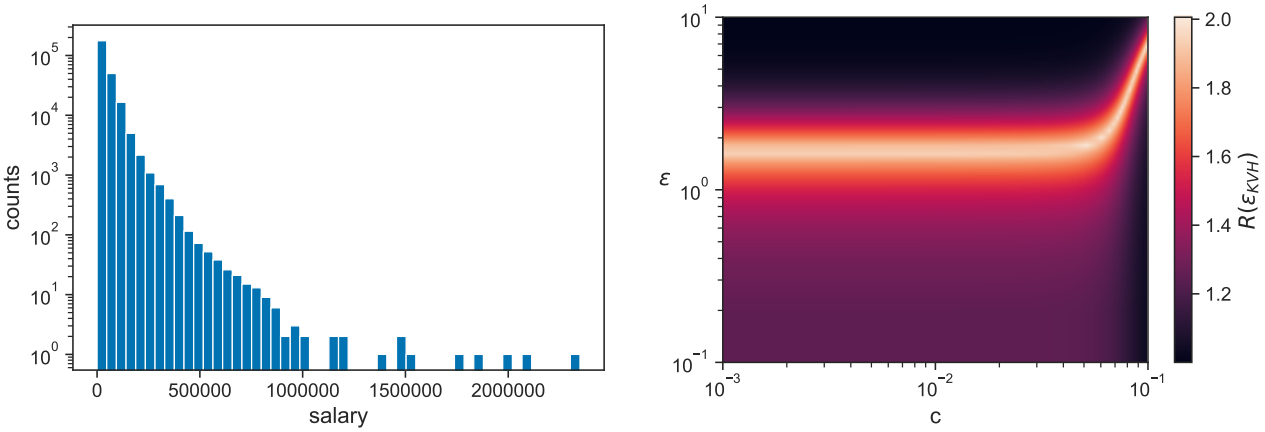


Figure 2.13: (a) Probability density functions of Beta(α, β) distributions for various α, β values. (b,c,d) The relative improvement $R(\mathcal{E}_{KVH})$ for each Beta distribution across a range of n values, for various c and ϵ values.



(a) (b)

Figure 2.14: (a) Distribution of salaries of UC employees. (b) The relative improvement $R(\mathcal{E}_{KVH})$ across a range of c and ϵ values.

(both assumed to be known). As σ, n, and m are determined by the dataset, we evaluate $R(\mathcal{E}_{KVH})$ across a large space of the remaining free parameters: $c \in [0.1\%, 10\%]$ and $\epsilon \in [0.1, 10]$. We see the relative improvement peak just above 2 in the high-privacy regime, with this maximum improvement continuing into the low-privacy regime.

2.3.4 Homogeneous, Unknown-Variance Setting

In this portion, we derive a different estimator from the hybrid family for the homogeneous setting, now applied to the case where the variance σ^2 of the data is not known. This is a more realistic setting, as an analyst with no knowledge of the distribution's mean typically also does not have knowledge of its variance.

The KVH estimator is able to use knowledge of the variance to weigh the estimates of the two groups so that the trade-off of excess sampling error and privacy error is optimally balanced. In this unknown-variance case, determining the optimal weighting is no longer viable. Nevertheless, we can heuristically choose a weighting which may (or may not) perform well depending on the underlying distribution. Thus, we propose a heuristic weighting choice for combining the groups' estimates and analyze it theoretically and empirically.

2.3.4.1 PWH Estimator

We now propose and analyze a hybrid estimator with a heuristically chosen weighting that is based on the amount of privacy noise each group adds. We choose this heuristic weighting by considering only the induced privacy error of each group's estimate. Thus, we refer to this as the "privacy-weighted hybrid" (PWH) estimator[10]. This weighting seeks solely to optimally balance privacy error between the groups, and therefore ignores the induced excess sampling error. Explicitly, from $\mathcal{E}_H(w)$ of Lemma 2.3.17 applied to the homogeneous setting, this weighting corresponds to choosing w to minimize $w^2 s_T^2 + (1-w)^2 \frac{s_L^2}{(1-c)n}$, stated in the following definition.

Definition 2.3.22. The privacy-weighted hybrid estimator is:

$$\tilde{\mu}_{PWH} = w_{PWH}\tilde{\mu}_T + (1 - w_{PWH})\tilde{\mu}_L,$$

where $w_{PWH} = \frac{s_L^2}{s_L^2 + (1-c)ns_T^2}$

Lemma 2.3.23. $\tilde{\mu}_{PWH}$ has MSE:

$$\mathcal{E}_{PWH} = \frac{(1-c)cn^2 s_T^4 \left(c\sigma^2 + s_L^2\right) + cns_L^2 s_T^2 \left(2(c-1)\sigma^2 + s_L^2\right) + (1-c)\sigma^2 s_L^4}{cn\left(s_L^2 + (1-c)ns_T^2\right)^2}.$$

This estimator has a mixture of both excess sampling error and privacy error. Since the privacy error was directly optimized, we expect this estimator to do well when the data variance σ^2 is small, as this will naturally induce small excess sampling error.

Now we are able to discuss the relative improvement of the PWH estimator over the baselines.

[10] We additionally investigated a naive weighting heuristic: weight the estimates based purely on the group size (i.e., $w = c$). We omit it because empirical evaluations showed that for practical parameters, it was inferior to the PWH estimator.

Theorem 2.3.24. The relative improvements of the PWH estimator $\tilde{\mu}_{PWH}$ over $\tilde{\mu}_T$ and $\tilde{\mu}_F$ are:

$$R(\mathcal{E}_{PWH}) = \gamma \cdot \begin{cases} \frac{1-c}{cn}\sigma^2 + s_T^2 & \text{if } c > c_{crit} \wedge n \leq n_{crit} \\ \frac{s_L^2}{n} & \text{otherwise,} \end{cases}$$

$$r(\mathcal{E}_{PWH}) = \gamma \cdot \begin{cases} \frac{s_L^2}{n} & \text{if } c > c_{crit} \wedge n \leq n_{crit} \\ \frac{1-c}{cn}\sigma^2 + s_T^2 & \text{otherwise,} \end{cases}$$

where $\gamma = \frac{cn\left(s_L^2+(1-c)ns_T^2\right)^2}{(1-c)cn^2s_T^4\left(c\sigma^2+s_L^2\right)+cns_L^2s_T^2\left(2(c-1)\sigma^2+s_L^2\right)+(1-c)\sigma^2s_L^4}$ and c_{crit} and n_{crit} are as defined in Definition 2.3.9.

Proof. Direct application of Lemmas 2.3.4, 2.3.6, and 2.3.23 to: Definition 2.3.9 for $R(\mathcal{E}_{PWH})$, and Definition 2.3.13 for $r(\mathcal{E}_{PWH})$. □

With the generic noise-addition privacy mechanisms, algebraic analysis of the weaker relative improvement measure reveals $r(\mathcal{E}_{PWH}) > 1$ unconditionally. That is, we confirm that the PWH estimator always outperforms at least one of the baseline mechanisms.

However, the regions where $R(\mathcal{E}_{PWH})$ is greater than 1 are difficult to obtain analytically with these generic mechanisms. By restricting our attention to the Laplace noise addition mechanism, we find that $R(\mathcal{E}_{PWH}) > 1$ is satisfied under certain conditions. The first is a "low relative privacy" regime where $\epsilon \geq \frac{\sqrt{2}m}{\sigma}$. That is, once ϵ is large enough, we have $R(\mathcal{E}_{PWH}) > 1$. For ϵ under this threshold, achieving $R(\mathcal{E}_{PWH}) > 1$ requires the following conditions on c and n: either $c \leq \frac{\epsilon^2\sigma^2}{2m^2}$, or $c > \frac{\epsilon^2\sigma^2}{2m^2} \wedge n < \frac{2m^2(1+c)}{c(2cm^2-\epsilon^2\sigma^2)}$. Since intuitively understanding these conditions can be challenging, we instead turn to an empirical evaluation of the estimator.

2.3.4.2 Empirical Evaluation of $R(\mathcal{E}_{PWH})$ and $r(\mathcal{E}_{PWH})$

Here, we perform an empirical evaluation of the PWH estimator analogous to the analysis done in Section 2.3.3.2.

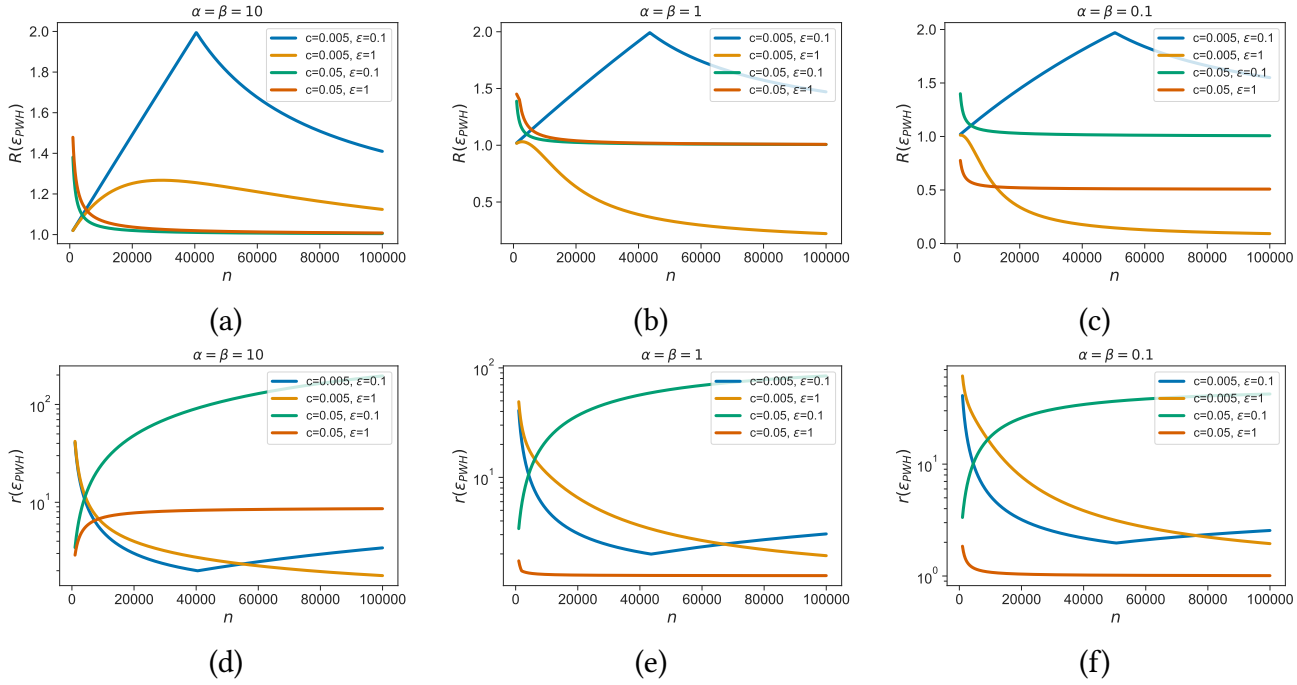
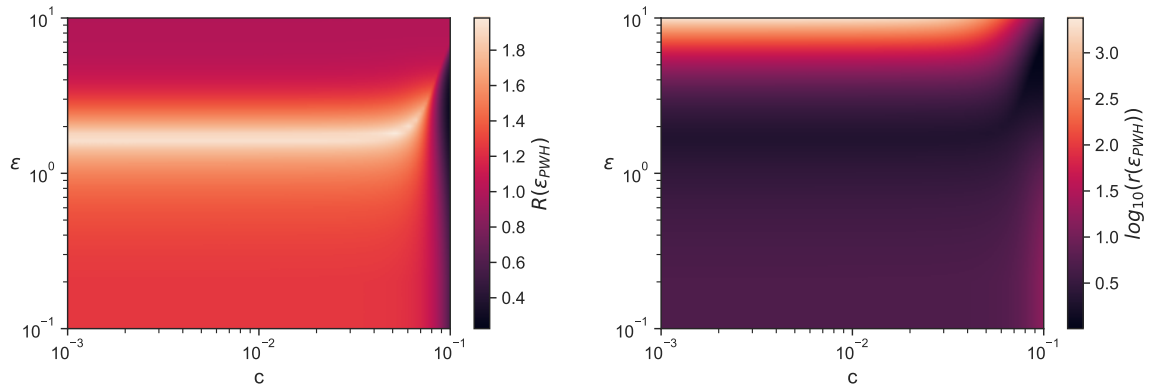


Figure 2.15: Across a range of n values, for various c and ϵ values for each Beta distribution (plotted in Figure 2.13a): (a,b,c) shows $R(\mathcal{E}_{PWH})$ values and (d,e,f) shows $r(\mathcal{E}_{PWH})$ values.

Figure 2.15 presents $R(\mathcal{E}_{PWH})$ (top row) and $r(\mathcal{E}_{PWH})$ (bottom row) using the same Beta distributions and parameters ($n \in [10^3, 10^5]$, $c \in \{0.5\%, 5\%\}$, and $\epsilon \in \{0.1, 1\}$). We find that there are many regions where $R(\mathcal{E}_{PWH})$ achieves a value of just greater than 1, and some regions where it achieves values competitive with the KVH estimator. Unsurprisingly, since this weighting is chosen without accounting for the variance, there are also clear regions where $R(\mathcal{E}_{PWH})$ is noticeably less than 1. Even in the regions where $R(\mathcal{E}_{PWH})$ is low, $r(\mathcal{E}_{PWH})$ (in the bottom row) shows that the PWH estimator often significantly improves over the worse of the two baseline estimators.

Figure 2.16 presents heat maps of $R(\mathcal{E}_{PWH})$ and $r(\mathcal{E}_{PWH})$ for the UC salaries dataset across the same parameters as before ($c \in [0.1\%, 10\%]$ and $\epsilon \in [0.1, 10]$), with the rightmost figure on a log scale. We find that $R(\mathcal{E}_{PWH})$ achieves a value of slightly greater than 1 across a large portion of the space. The results here tell a similar story to that of Figure 2.15. Most of the space has $R(\mathcal{E}_{PWH})$ values above 1, and even approaching 2 in a narrow region. There is also a small region at the large c values where the relative improvement drops below 0.5. The majority of the



(a) (b)

Figure 2.16: The relative improvements $R(\mathcal{E}_{PWH})$ (a) and $r(\mathcal{E}_{PWH})$ (b) across a range of c and ϵ values, with a log scale on (b).

space has $r(\mathcal{E}_{PWH})$ between 10 and 100, although it includes a region at the high ϵ values where this relative improvement exceeds $1,500$.

2.3.5 Heterogeneous Setting

Having examined our hybrid estimators in the heterogeneous setting, we now turn our focus to examining the effects of the groups' distributions diverging on the quality of our estimators. This is motivated by the fact that the hybrid model allows users to self-partition based on their trust preferences. Such self-partitioning may cause the groups' distributions to be different. For instance, since the TCM users have similar trust preferences, their data may also be more similar to each other's than to the LM users' data. This could manifest as variance-skewness between the groups. Alternatively, the TCM users may have fundamentally different data than the LM users, which would manifest as mean-skewness between the groups. Thus, we examine the case where the group means are the same but their variances are different, as well as the case where the group means are different but their variances are the same. To understand these skewness effects, we empirically evaluate $R(\mathcal{E}_{KVH})$[11].

[11]We also performed the same empirical evaluation with the unknown-variance PWH estimator. The results were very similar to the KVH estimator's, and the conclusions were the same. Thus, we omit them for brevity.

Although the heterogeneous setting is more general and complex, we can still derive the optimal weighting for the KVH estimator analogously to homogeneous KVH weighting of Definition 2.3.18.

Definition 2.3.25. The known-variance hybrid estimator in the heterogeneous setting is:

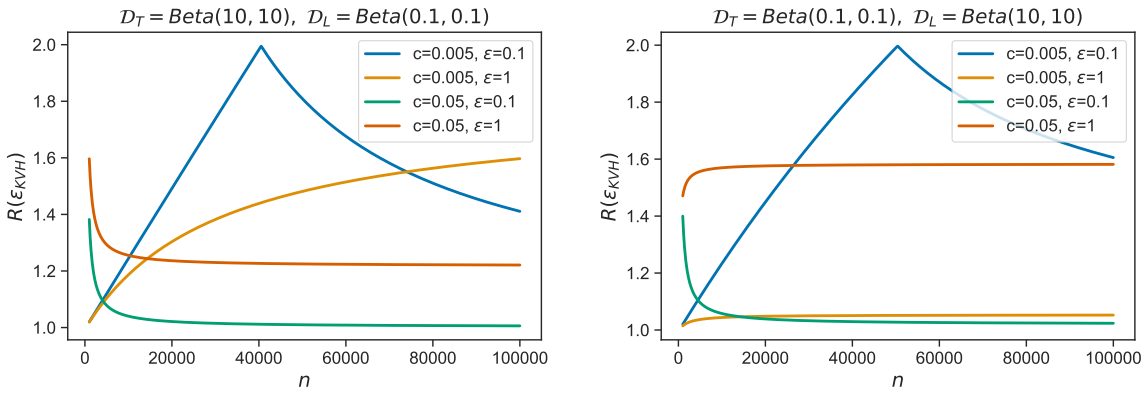
$$\tilde{\mu}_{KVH} = w^* \tilde{\mu}_T + (1 - w^*)\tilde{\mu}_L,$$

where $w^* = \frac{c(s_L^2 + c\sigma_L^2 + (1-c)(n(\mu_L - \mu)(\mu_L - \mu_T) + \sigma_T^2))}{cs_L^2 + (1-c)cn((\mu_L - \mu_T)^2 + s_T^2) + c\sigma_L^2 + (1-c)\sigma_T^2}$

2.3.5.1 Variance-Skewness

In the heterogeneous setting, we first analyze the case where $\mu_T = \mu_L$ but $\sigma_T^2 \neq \sigma_L^2$. This reduces the KVH estimator's weighting to $w^* = \frac{c(s_L^2 + c\sigma_L^2 + (1-c)\sigma_T^2)}{cs_L^2 + (1-c)cns_T^2 + c\sigma_L^2 + (1-c)\sigma_T^2}$. To gain insight into the effect of variance-skewness, we recall two Beta distributions previously used in our empirical evaluations: the low variance Beta$(10, 10)$ distribution ($\sigma = 0.109$) and the high variance Beta$(0.1, 0.1)$ distribution ($\sigma = 0.456$). We evaluate $R(\mathcal{E}_{KVH})$ in two scenarios: when the TCM group has data drawn from the low variance distribution but the LM group has data drawn from the high variance distribution, and vice versa. Figure 2.17 gives the results across the same range of n, c, and ϵ values as used in previous experiments.

The similarities between Figure 2.17 and Figure 2.13 demonstrate that our estimator is robust to deviations in the LM group's variance. For example, Figure 2.13b shows $R(\mathcal{E}_{KVH})$ when all the data is from the low variance distribution; that figure nearly exactly matches Figure 2.17a despite the fact that most of the data is now from the LM group's high variance distribution. As this applies to both of Figure 2.13's graphs, it is clear that the relative improvement heavily depends on the variance of the TCM group, regardless of whether the LM group had the low or high variance data. In fact, in both graphs, the difference in relative improvement from the homogeneous case with variance σ^2 to the heterogeneous case where only the TCM group has variance $\sigma_T^2 = \sigma^2$ does not vary by more than ± 0.1, and, typically, varies by less than ± 0.01.



(a) (b)

Figure 2.17: The relative improvement $R(\mathcal{E}_{KVH})$ values when: (a) the TCM group has low variance data but the LM group has high, and (b) when the TCM group has high variance data but the LM group has low.

2.3.5.2 Mean-Skewness

Rather than the groups' variances differing, we now analyze the case where $\mu_T \neq \mu_L$ but $\sigma_T^2 = \sigma_L^2$. This reduces the KVH estimator's weighting to $w^* = \frac{c(s_L^2+(1-c)n(\mu_L-\mu)(\mu_L-\mu_T)+\sigma^2)}{cs_L^2+(1-c)cn((\mu_L-\mu_T)^2+s_T^2)+\sigma^2}$. Importantly, this expression depends on the curator's knowledge of μ_T and μ_L – an unreasonable requirement, since the curator's overarching goal is to learn the mean from the user data. For applications where the groups' means are assumed to be different, computing separate estimates of each group's mean in their respective trust models would likely be more useful than a joint estimate. Thus, we instead explore mean-skewness from the point of view of a curator who *mistakenly believes* they are operating in the homogeneous setting, and thus uses the homogeneous weighting from Definition 2.3.18. This is useful in practice, as it demonstrates how a curator can use our analytical expressions for their specific problem instance to understand how utility is affected by misspecified assumptions about user data.

To analyze this case, we set up the following experiment, displayed in Figure 2.18. We start with the control for the experiments: set both groups to the same distribution $\mathcal{D}_T = \mathcal{D}_L$ and obtain $R(\mathcal{E}_{KVH})$. Next, we retain the distributional shape for both groups, but shift them in opposite directions; e.g., $\mathcal{D}_T - t, \mathcal{D}_L + t$ for some t. We obtain the new $R(\mathcal{E}_{KVH})$ values under

these distributions, and compare against the un-shifted results. For clarity, we denote the relative improvement on the t-shifted distribution as $R^t(\mathcal{E}_{KVH})$.

We expect that as the divergence in means t increases, the relative utility of our hybrid estimator will decrease. To test this hypothesis concretely, we use the medium variance Beta$(1, 1)$ distribution ($\sigma = 0.289$) from our previous empirical evaluations as the experiment's base distribution. We center this distribution at 1 without rescaling, inducing support on $[0.5, 1.5]$. Then we set both $\mathcal{D}_T$ and $\mathcal{D}_L$ to this distribution, and obtain $R^0(\mathcal{E}_{KVH})$ on it (Figure 2.18ab). Next, we add a small shift of $t = 0.25$ to each of the groups' distributions in opposite directions; i.e., $\mathcal{D}_T - 0.25$ and $\mathcal{D}_L + 0.25$, so that $|\mu_T - \mu_L| = 0.5$. These distributions, along with the corresponding $R^{0.25}(\mathcal{E}_{KVH})$ results, are shown in the second column of Figure 2.18. Finally, the third column of Figure 2.18 shows the analogous distributions and results when a large shift of $t = 0.5$ is added so that $|\mu_T - \mu_L| = 1$.[12] Unsurprisingly, these results depict a clear negative impact on the relative improvement as the means diverge, showing that our estimator is sensitive to skewness in the groups' means.

2.3.6 Hybrid Estimator Applications

Taking a step back from analyzing the utility of hybrid estimators, in this portion we demonstrate how more complex non-hybrid mechanisms can be easily extended into the hybrid model by inserting our hybrid estimator as a mean estimation primitive. In particular, we implement a hybrid variant of the classic DP K-means mechanism [Dwo11] using the PWH hybrid estimator as a sub-component, then empirically evaluate its effectiveness.

[12] One caveat to these shifts is that as the data distribution becomes wider, the noise required to ensure DP must increase. Since we are interested in the effect of mean-skewness here, and not the effect of distribution-width, we conservatively fix $m = 2$ for all experiments. That is, the same level of noise is used across shift-amounts, even if less noise may have sufficed to ensure DP.

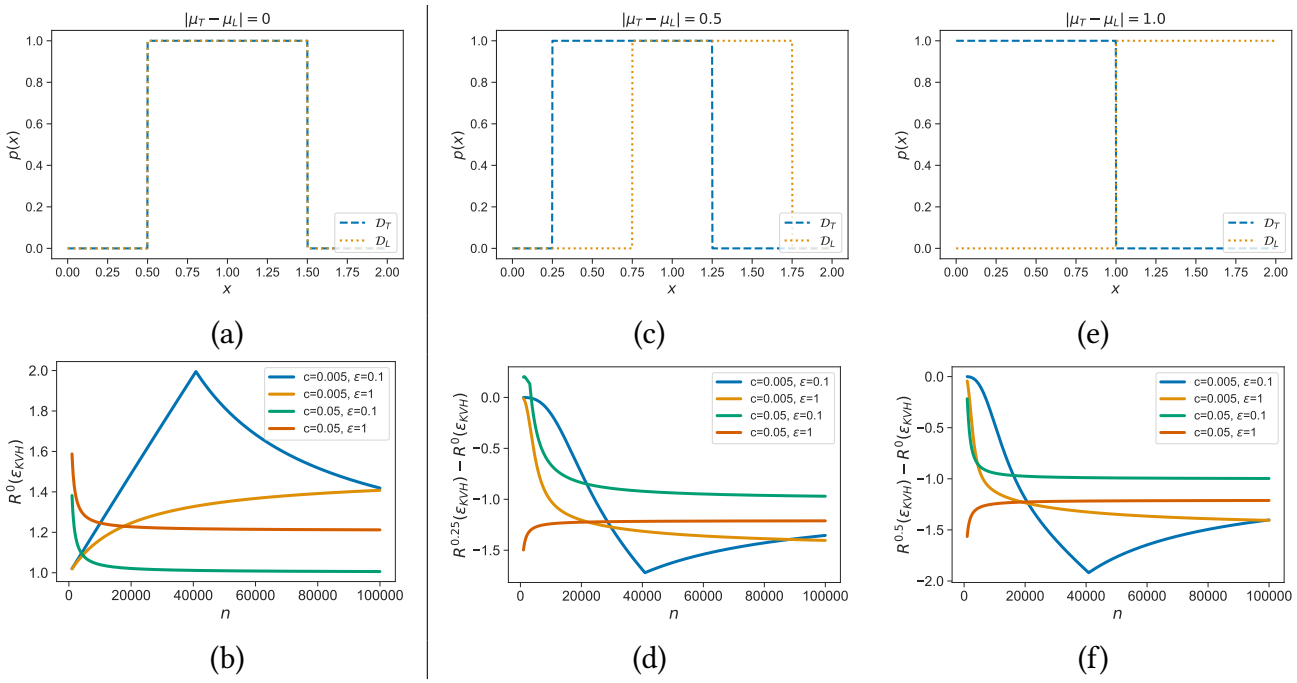


Figure 2.18: (Left column) Initial data distributions with no mean shift, and the KVH estimator's corresponding relative improvement. Small (middle column) and large (right column) mean shifts of the initial data distribution with $t = 0.25$ and $t = 0.5$ respectively, along with the KVH estimator's corresponding change in relative improvement.

The K-means problem is to partition n d-dimensional real-valued observations $x_1, \ldots, x_n$ into K clusters $C_1, \ldots, C_K$ such that the within-cluster sum of squares (WCSS) is minimized. Letting μ_k denote the center of cluster C_k, the formal problem is to solve the following:

$$\arg\min_{C_1,\ldots,C_K} \sum_{k=1}^{K} \sum_{x \in C_k} ||x - \mu_k||^2.$$

This problem is NP-hard to solve exactly, and thus heuristic algorithms are generally used to solve it approximately. The classic DP mechanism [Dwo11] for this problem was designed for the TCM. This mechanism partitions the total privacy budget across τ iterations, and each iteration refines the estimates of the clusters' centers. Each iterative refinement assigns the observations to their nearest cluster, then updates each cluster's center to the mean of all points within it while carefully applying Laplace noise.

2.3.6.1 Defining the Hybrid K-Means Mechanism

We extend this mechanism to the LM in a simple way. First, LM users expend a portion of their privacy budget reporting their data to the curator with Laplace noise. The curator uses their data analogously to the TCM case, except that in each iteration, LM users use a portion of their privacy budget to report the nearest cluster to them using a generalization of Randomized Response (very similar to the clients' query reporting in `BLENDER`) — this reduces bias in the cluster centers, relative to computing the nearest cluster directly based on their already reported data.

Other DP K-means mechanisms exist in both the TCM [NRS07; Su+16; BF16; Noc+16; Bal+17; LS20] and LM [NS18; SZY19; Xia+20; Ste20] which improve on our two non-hybrid K-means mechanisms. However, our goal here is to simply demonstrate how our hybrid estimator can be effectively leveraged in more complex applications. Thus, we present our hybrid K-means mechanism in Algorithm 2.7, which combines our simpler TCM and LM mechanisms in the following straightforward way. Each separate mechanism performs its iterative refinement as previously described. Then, at the end of each iteration, the TCM and LM cluster center estimates are combined using the PWH estimator on each dimension.

2.3.6.2 Evaluating the Hybrid K-Means Mechanism

We evaluate the hybrid mechanism in the following experiment, showing that it automatically achieves WCSS on par with the best baseline mechanism. The baselines here, analogous to our estimators' `TCM-Only` and `Full-LM` MSE baselines, are: the WCSS of the TCM variant using only TCM data, and the WCSS of the LM variant using all data. The dataset used for evaluations is shown in Figure 2.19a: 4 clusters of 2-dimensional spherical Gaussian data with scale $\sigma \approx 0.028$ and $40,000$ points per cluster. Rather than focusing on a more complex instance, we chose this relatively straightforward, low-dimensional clustering instance because we are using the single-dimensional mean estimators (in both the classic trust models as well as the hybrid model) independently across each dimension as well as repeatedly across iterations. For more

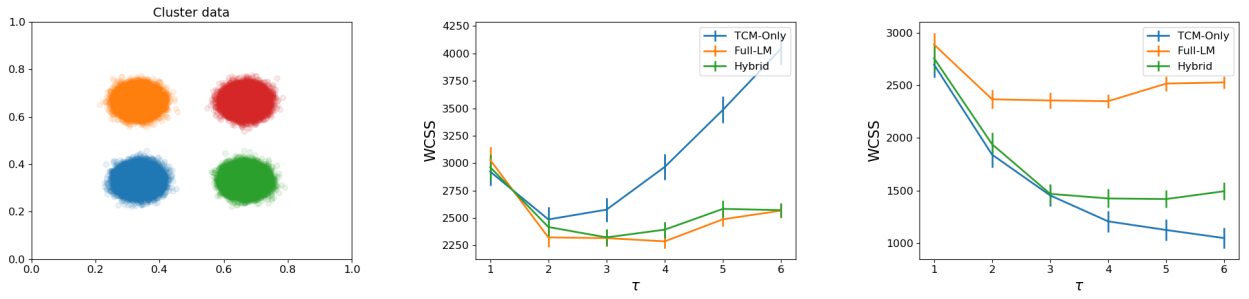
Algorithm 2.7 Hybrid-DP K-means

Input

- T, L: Sets of TCM and LM users, respectively.
- m: Maximum range of the data.
- d: Dimension of the data.
- K: Total number of clusters.
- τ: Number of estimation iterations.

Body

Initialize centers of clusters $C_1, \ldots, C_K$.
Let $b_T = \frac{(md+1)\tau}{\epsilon}$ and $b_L = \frac{md(\tau+1)}{\epsilon}$.
Each $i \in T$ reports $\tilde{x}_i = x_i$ to the curator.
Each $i \in L$ reports $\tilde{x}_i = x_i + Y_{L,i}$ to the curator, $Y_{L,i} \sim \text{Lap}^d(b_L)$.
for $t = 1 \ldots \tau$ **do**
    Assign each $\tilde{x}_i$ from T to closest cluster non-privately.
    Assign each $\tilde{x}_i$ from L to closest cluster with prob. $\frac{\exp(\epsilon/(\tau+1))-1}{K+\exp(\epsilon/(\tau+1))-1}$; to a uniformly random cluster otherwise.
    for $k = 1 \ldots K$ **do**
        Count T users in cluster k with DP: $\tilde{N}_T = |C_k \cap T| + Y_1,\ Y_1 \sim \text{Lap}(b_T)$.
        Compute mean of all T users' data in cluster k with DP: $\tilde{\mu}_T = \frac{1}{\tilde{N}_T}\left(\sum_{i \in T} x_i + Y_2\right),\ Y_2 \sim \text{Lap}^d(b_T)$.
        Count L users in cluster k: $\tilde{N}_L = |C_k \cap L|$.
        Compute mean of all L users' data in cluster k: $\tilde{\mu}_L = \frac{1}{\tilde{N}_L}\sum_{i \in L} \tilde{x}_i$.
        Let $c = \frac{\tilde{N}_T}{\tilde{N}_T + \tilde{N}_L}$, $s_T^2 = 2b_T^2$, and $s_L^2 = 2b_L^2$.
        Compute w_{PWH} as defined in Def. 2.3.22.
    end for
end for
Return: centers of $C_1, \ldots, C_k$.



(a) (b) (c)

Figure 2.19: (a) Clustering dataset with 4 clusters of 2d spherical Gaussians with $\sigma \approx 0.028$ and $40,000$ points per cluster. (b,c) WCSS values of each model's mechanism across a range of total iterations τ, 0.1% and 1% fractions of TCM users respectively, and $\epsilon = 7$.

complex instances, more advanced privacy-preserving clustering mechanisms in the classic trust models should be applied, each requiring their own adaptations to fit into our hybrid model. For this problem instance, the privacy budget for each mechanism is $\epsilon = 7$; this relatively high budget is necessitated by the TCM and LM mechanisms to achieve acceptable practical utility. In Figure 2.19bc, across a range of total iterations τ and fractions of TCM users 0.1% and 1%, we evaluate the mean WCSS values of each model's mechanism with 364 trials. The regimes where each non-hybrid mechanism is better than the other is unclear a priori, and the results here show one example of each. By simply combining the two using our hybrid estimator, the hybrid mechanism is able to maintain a WCSS approximately equal to the better of the two.

2.4 Privacy Amplification Via Intergroup Interaction

In this section, we address the third high-level question of this chapter. Specifically, we answer the question:

> *To what extent does the privacy of a hybrid mechanism depend on the computations performed and on interactions between the two groups?*

The benefit, and even the necessity, of intergroup interaction in the hybrid model is an open area of research. In Section 2.2 with the `BLENDER` mechanism, we have shown experimentally that

high utility is achievable by intelligently utilizing intergroup interaction. In Section 2.3 with hybrid mean estimators, we have shown mathematically that we can guarantee high utility for mean estimation without any intergroup interaction. However, both works only focus on intergroup interaction's effect on a mechanism's *utility* — neither consider its effect on *privacy*. Each group is assumed to independently guarantee privacy, without considering how subsequent interaction and processing by the curator may affect the DP guarantee. The post-processing property of DP ensures that such interaction and processing will never degrade privacy, but the question of whether it improves privacy has remained unstudied. It is precisely this effect on privacy that we introduce and examine in this portion. We find that for our hybrid mean estimators, the differential privacy guarantee against certain adversaries can be significantly improved.

We are specifically interested in users' privacy against adversaries who can view the output of the curator's computation (i.e., output-viewing adversaries). This is the classic adversary model that the TCM protects against. The LM protects against a larger class of adversaries: the output-viewing adversaries, as well as against the curator itself. However, the LM's singular DP guarantee does not distinguish between these adversary types. In the hybrid model, each group's DP guarantee may be overly conservative against output-viewing adversaries. This is because it does not account for the curator's joint processing of the LM users' reports — which each include their own privacy noise — in conjunction with the TCM group's privacy noise.

Our Contributions

In this section, we concretely investigate users' DP guarantee against output-viewing adversaries as a result of: 1) the combined privacy noise from both groups, in conjunction with 2) the intergroup interaction strategy of the curator. We show that these two components together can serve to amplify users' privacy against this adversary class. This provides a two-tier DP guarantee for LM users — their standard DP guarantee against the curator, and an improved guarantee against output-viewing adversaries — and an improved DP guarantee for TCM users.

To accomplish this, we first analyze the intergroup privacy amplification of our hybrid mean estimator family. To start, we show how some DP mechanisms may not enable any such privacy amplification. We immediately follow this up by showing how other DP mechanisms may enable significant intergroup privacy interaction. Then, turning our focus to the BLENDER mechanism, we detail why its interaction strategy does not provide any intergroup privacy amplification. Together, these examples highlight the value of looking at the effects of intergroup interaction not only on utility, but also on privacy.

2.4.1 Hybrid Mean Estimator Amplification

For the hybrid mean estimator family (Definition 2.3.16) that generically utilizes additive noise mechanisms to ensure DP, we describe how intergroup privacy amplification may arise. We then show that users' DP guarantee depends strongly on which concrete additive noise mechanism is chosen.

Recall that the hybrid estimator family utilizes no intergroup interaction. I.e., the curator only outputs once: after it has received all the LM users' reports, computed both groups' estimates, then combined them. For adversaries that can only view the output of this curator, the combined noise from *all* the LM users and the TCM group can serve to improve the DP guarantee. To see this, we rewrite the estimator as

$$\begin{aligned}\tilde{\mu}_H(w) &= w\tilde{\mu}_T + (1-w)\tilde{\mu}_L \\ &= w\left(\frac{1}{cn}\sum_{i\in T} x_i + Y_T\right) + (1-w)\left(\frac{1}{(1-c)n}\sum_{i\in L}(x_i + Y_{L,i})\right) \\ &= \underbrace{\left(\frac{w}{cn}\sum_{i\in T} x_i + \frac{1-w}{(1-c)n}\sum_{i\in L} x_i\right)}_{\text{non-private hybrid mean estimator}} + \underbrace{\left(wY_T + \frac{1-w}{(1-c)n}\sum_{i\in L} Y_{L,i}\right)}_{\text{joint privacy noise}}.\end{aligned}$$

Thus, this joint privacy noise is providing some DP guarantee for the mechanism as a whole, rather than individual noises protecting the individual groups.

There is one caveat: the TCM users' noise is provided by the curator and never revealed to them, but the LM users each provide their own noise. DP requires that the privacy noise not be known to an adversary. Any noise that is known cannot be considered towards the DP guarantee. Here, we assume LM users are semi-honest; i.e., they apply the specified mechanism properly to their data, but they know the privacy noise they add. Thus, LM user i's knowledge of their own privacy noise weakens the joint noise term by an additive $\frac{1-w}{(1-c)n}Y_{L,i}$ amount. Furthermore, they may choose to form coalitions with other users and share this knowledge to adversarially weaken the joint privacy noise term. The largest such coalition, denoted by A, reduces the joint privacy noise by $\frac{1-w}{(1-c)n}\sum_{i\in A}Y_{L,i}$. Excluding the largest such coalition's noise enables the remaining joint privacy noise to be analyzed for a DP guarantee.

The DP guarantee from the remaining joint noise depends on the privacy mechanisms used by the TCM group and each LM user. To establish this claim, we prove two theorems here. The first is that TCM group and LM users all utilizing the ϵ-DP Laplace mechanism does not enable any privacy amplification. The second is that the TCM group and LM users all utilizing the (ϵ, δ)-DP Gaussian mechanism can enable significant privacy amplification.

2.4.1.1 Amplification Analysis with the Laplace Mechanism

For our first amplification analysis, we show that the ϵ-DP Laplace mechanism would yield a joint noise term which guarantees ϵ'-DP where $\epsilon' = \epsilon$. That is, there is no intergroup privacy amplification when using this mechanism to ensure privacy of our hybrid estimators.

Theorem 2.4.1. Assume the curator adds Laplace noise of variance s_T^2 to provide an ϵ-DP guarantee for the TCM group, and that each LM user adds Laplace noise of variance s_L^2 to provide an ϵ-DP guarantee for themselves. The users' ϵ'-DP guarantee against output-viewing adversaries is given by $\epsilon' = \epsilon$.

Proof. In this proof, we show that the unweighted sum of n reports, each privatized by the ϵ-DP Laplace mechanism, only provides an ϵ'-DP joint guarantee of $\epsilon' = \epsilon$. A convex weighting

of the terms in this sum, necessary for normalization (to compute the mean instead of sum) as well as to account for differing report weightings (such as the difference in weight between the TCM group's joint report compared to each LM user's individual report) yields the same joint guarantee, proving our claim. To formalize this, we define the privatized sum of n reports as

$$\begin{aligned}\tilde{S}_n &= \sum_{i\in[n]}(x_i + Y_i) \\ &= \underbrace{\sum_{i\in[n]} x_i}_{S_n} + \underbrace{\sum_{i\in[n]} Y_i}_{Y}\end{aligned}$$

where $x_i \in [0, m]$, $Y_i \sim \text{Lap}(b)$, and $b = m/\epsilon$ for each i. We show here that the joint noise Y provides ϵ'-DP for S_n against output-viewing adversaries, where $\epsilon' = \epsilon$.

To begin, we show that accounting for the joint privacy noise never yields a DP guarantee that is weaker than any user's original DP guarantee. First, note that each user at least has the ϵ-DP guarantee via their own privacy noise. Thus, by the post-processing property of DP, we have $\epsilon' \leq \epsilon$ as a trivial upper bound. If $\epsilon \leq \epsilon'$ without any adversarial users, then our upper bound implies $\epsilon \leq \epsilon'$ with an arbitrary number of adversarial users.

Having established an upper bound, we turn to lower-bounding ϵ'; i.e., determining a maximal level of privacy amplification. For simplicity, we perform this analysis assuming that no users are adversarial; however, the results of this analysis are the same if we assume that there are $n' < n$ adversarial users, so this assumption is without loss of generality. We begin this analysis by examining the probability distribution of each user's individual noise, and how all users' privacy noises combine into a new probability distribution. Each user's individual privacy noise

Y_j sampled from the Laplace distribution with scale b can be defined in terms of the distribution's characteristic function (i.e., the Fourier transform of the distribution's probability density function [Luk72]) as

$$\begin{aligned}\varphi_{Y_j}(t) &= \mathbb{E}[e^{itY_j}] \\ &= \frac{1}{1+b^2t^2}.\end{aligned}$$

The characteristic function of Y is then

$$\begin{aligned}\varphi_Y(t) &= \mathbb{E}[e^{it\sum_{j\in[n]}Y_j}] \\ &= \prod_{j\in[n]}\varphi_{Y_j}(t) \\ &= \left(\frac{1}{1+b^2t^2}\right)^n.\end{aligned}$$

Y's probability density function, $p_Y(x)$, can be recovered from the characteristic function [Bil08] via the inverse Fourier transform as

$$\begin{aligned}p_Y(x) &= \frac{1}{2\pi}\int_{\mathbb{R}} e^{itx}\overline{\varphi_Y(t)}dt \\ &= \frac{1}{2\pi}\int_{\mathbb{R}}\frac{e^{itx}}{(1+b^2t^2)^n}dt \\ &= \frac{2^{\frac{1}{2}-n}}{\sqrt{\pi}b^{\frac{1}{2}+n}\Gamma(n)}\mathcal{K}_{\frac{1}{2}-n}\left(\frac{|x|}{b}\right)|x|^{n-\frac{1}{2}},\end{aligned}$$

where $\overline{\varphi_Y(t)}$ is the complex conjugate of $\varphi_Y(t)$ and $\mathcal{K}_\cdot(\cdot)$ is the modified Bessel function of the second kind [AS68]. For ϵ'-DP, noting that sensitivity $\Delta_1(S_n) = m$, we must bound

$$-\epsilon' \leq \max_{k\in[-m,m]}\log\left(\frac{p_Y(x)}{p_Y(x+k)}\right) \leq \epsilon'.$$

Consider the instance where $k = m$ and $x \to \infty$:

$$\begin{aligned}
&\lim_{x\to\infty} \log\left(\frac{p_Y(x)}{p_Y(x+k)}\right) \\
&= \lim_{x\to\infty} \log\left(\frac{\mathcal{K}_{\frac{1}{2}-n}\left(\frac{x}{b}\right)}{\mathcal{K}_{\frac{1}{2}-n}\left(\frac{x+m}{b}\right)}\left(\frac{x}{x+m}\right)^{n-\frac{1}{2}}\right) \\
&= \frac{m}{b}.
\end{aligned}$$

By the definition of b, we have $\frac{m}{b} = \epsilon$. Therefore

$$\begin{aligned}
\epsilon &= \lim_{x\to\infty} \log\left(\frac{p(x)}{p(x+k)}\right) \\
&\leq \max_{k\in[-m,m]} \log\left(\frac{p(x)}{p(x+k)}\right) \\
&\leq \epsilon'.
\end{aligned}$$

Thus, we conclude that $\epsilon' = \epsilon$. □

2.4.1.2 Amplification Analysis with the Gaussian Mechanism

Having proven a negative amplification result based on the Laplace mechanism, we now turn to a positive amplification result based on the Gaussian mechanism. Specifically, consider the hybrid estimator family where the Gaussian mechanism is used to ensure privacy; i.e., where the curator adds $Y_T \sim \text{Normal}(0, s_T^2)$ and each LM user i adds $Y_{L,i} \sim \text{Normal}(0, s_L^2)$, where s_T^2 and s_L^2 are calibrated to ensure (ϵ, δ)-DP for both groups. Analyzing the joint noise of such a hybrid estimator provides the following amplified DP guarantee against output-viewing adversaries. We first provide the mathematical result and proof, then illustrate the impact of the result with a brief empirical evaluation.

Theorem 2.4.2. Assume the curator adds Gaussian noise of variance s_T^2 to provide an (ϵ, δ)-DP guarantee for the TCM group, and that each LM user adds Gaussian noise of variance s_L^2 to

provide an (ϵ, δ)-DP guarantee for themselves. Furthermore, assume that the largest adversarial coalition is A. Define

$$s'^2 = w^2 s_T^2 + \left(\frac{1-w}{(1-c)n}\right)^2 |L \setminus A| s_L^2.$$

The users' ϵ'-DP guarantee against output-viewing adversaries is given by:

$$\epsilon' = \frac{\sqrt{2\ln(1.25/\delta)}m}{ns'} \cdot \begin{cases} \frac{w}{c} & \text{if } w \leq c \\ \frac{1-w}{1-c} & \text{otherwise.} \end{cases}$$

Proof. Let $\hat{\mu}_H(w)$ denote the non-private hybrid mean estimator, defined as

$$\hat{\mu}_H(w) = \frac{w}{cn}\sum_{i\in T} x_i + \frac{1-w}{(1-c)n}\sum_{i\in L} x_i.$$

Let Y denote the joint privacy noise without the largest adversarial coalition, defined as

$$Y = wY_T + \frac{1-w}{(1-c)n}\sum_{i\in L\setminus A} Y_{L,i}.$$

We first compute the sensitivity $\Delta_2(\hat{\mu}_H(w)) = \max \|\hat{\mu}_H(w) - \hat{\mu}'_H(w)\|_2$, where $\hat{\mu}_H(w)$ is the estimator on any dataset $D = T \cup L$ and $\hat{\mu}'_H(w)$ is the estimator on any neighboring dataset $D' = T' \cup L'$ differing in the data of at most one user. If the data of one T user is changed, then $\max \|\hat{\mu}_H(w) - \hat{\mu}'_H(w)\|_2 \leq \frac{wm}{cn}$. If instead the data of one L user is changed, then $\max \|\hat{\mu}_H(w) - \hat{\mu}'_H(w)\|_2 \leq \frac{(1-w)m}{(1-c)n}$. When $w \leq c$, we have that $\frac{wm}{cn} \leq \frac{(1-w)m}{(1-c)n}$. Thus,

$$\Delta_2(\hat{\mu}_H(w)) = \begin{cases} \frac{wm}{cn} & \text{if } w \leq c \\ \frac{(1-w)m}{(1-c)n} & \text{otherwise.} \end{cases}$$

We now characterize the joint privacy noise. Let $Y_T \sim \text{Normal}(0, s_T^2)$ and $Y_{L,i} \sim \text{Normal}(0, s_{L,i}^2)$ such that s_T satisfies (ϵ, δ)-DP for the TCM group and $s_{L,i}$ satisfies (ϵ, δ)-DP for each LM user i. Because a weighted combination of Gaussians is also a Gaussian, we have

$$Y \sim \text{Normal}\left(0,\ s'^2 := w^2 s_t^2 + \left(\frac{1-w}{(1-c)n}\right)^2 |L \setminus A| s_L^2\right).$$

Recall that the classic privacy result for the Gaussian mechanism (Theorem 1.1.7) guarantees (ϵ, δ)-DP for a function f with sensitivity $\Delta_2(f)$ by adding noise from $\text{Normal}(0, s^2)$ such that $s = \sqrt{2\log(1.25/\delta)}\Delta_2(f)/\epsilon$. Applying this result to our problem with a fixed $\delta' = \delta$ and solving ϵ', we have

$$\begin{aligned}
\epsilon' &= \frac{\sqrt{2\ln(1.25/\delta)}\Delta_2(\hat{\mu}_H(w))}{s'} \\
&= \frac{\sqrt{2\ln(1.25/\delta)}m}{ns'} \cdot \begin{cases} \frac{w}{c} & \text{if } w \leq c \\ \frac{1-w}{1-c} & \text{otherwise.} \end{cases}
\end{aligned}$$

□

For practical applications, even with a moderate fraction of adversarial LM users, the privacy amplification achieved when using the Gaussian mechanism can be significant. To make this concrete, consider the UC salary dataset used in the previous experiments (previously displayed in Figure 2.14). Suppose we compute the KVH estimator with each group using the Gaussian mechanism with $\epsilon = 1$ and $\delta = 10^{-7}$. In Figure 2.20, we plot all users' amplified ϵ' value across $c \in [0.1\%, 10\%]$ and across the fraction of LM users assumed to be adversarial. In most of the space, the privacy amplification that users receive corresponds to more than a doubling of their privacy budget.

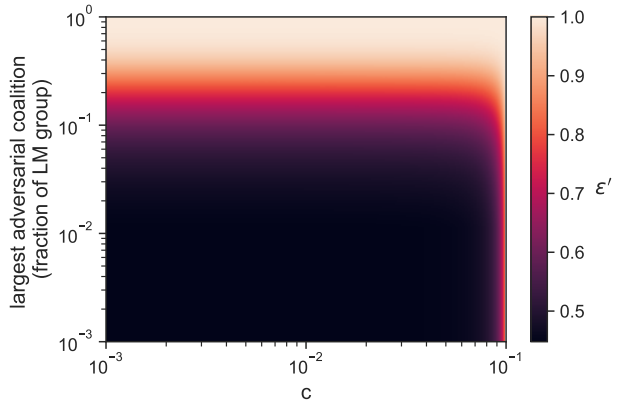


Figure 2.20: The amplified (ϵ', δ)-DP guarantee when the $(1, 10^{-7})$-DP Gaussian mechanism is used in the KVH estimator.

2.4.1.3 Amplification Disparity Between the Gaussian and Laplace Mechanisms

We have proven in this section that our hybrid estimator using the ϵ-DP Laplace mechanism cannot amplify privacy, whereas using the (ϵ, δ)-DP Gaussian mechanism can amplify privacy by more than a factor of two. However, we have not discussed why this may be the case. The possible causes are either the mechanisms themselves, or the DP definition being utilized (i.e., pure DP for the Laplace mechanism and approximate DP for the Gaussian mechanism). One promising approach to understanding the root cause could be to analyze our hybrid estimator using the general (ϵ, δ)-DP variant of the Laplace mechanism (Theorem 1.1.5), and studying the resulting joint privacy noise in the approximate DP setting. If privacy amplification is still not possible, this indicates that the mechanism itself is the root cause. On the other hand, if privacy amplification is possible using the (ϵ, δ)-DP Laplace mechanism, this hints that relaxing the DP definition from pure to approximate DP might be the primary factor for making amplification possible. We do not perform this analysis in this work, and instead leave it as an intriguing direction for future study.

2.4.2 BLENDER Amplification

Now that we have shown how a *lack* of intergroup interaction can facilitate privacy amplification, we turn our focus to BLENDER's intergroup interaction strategy designed to improve utility. Informally, BLENDER takes advantage of the TCM group by having it identify the heavy hitters. The TCM group then passes the identified heavy hitters on to the LM users, who perform frequency estimation. The curator then combines the LM users' reports and outputs the heavy hitters along with their frequencies.

One might be tempted to analyze this final output for an amplified DP guarantee. However, the initial output of the curator — the privatized list of heavy hitters from the TCM group — has already been released to all LM users. Unless *all* LM users are guaranteed to be non-adversarial, the TCM users gain no further benefit from the incorporation of the LM users' privacy noise. Conversely, the LM users may experience privacy amplification through the combination of their locally incorporated privacy noise. However, this is solely due to *intragroup* interaction in the LM. Such intragroup interaction (and related topics) are currently an active area of research on the LM [Bit+17; Che+19; Bal+19a; Erl+19; GPV19; BC20; Gha+21], and we do not explore them further here.

2.5 Related Works

In this section, we discuss other works related to our exploration of the hybrid DP model in this chapter. We first briefly discuss the most relevant subsequent work to ours, which takes a theoretical approach to proving the hybrid model's power over the classic trust models. We then discuss some of the other trust models in differential privacy beyond the two classic trust models and our hybrid model. Finally, we briefly discuss other works on the non-hybrid DP mean estimation problem.

Proof of the Hybrid Model's Power

The most directly related research to the topics presented in this chapter is the subsequent work of Beimel et al. [Bei+20]. Their work examines precisely the same hybrid DP model as this work, the combined *trusted curator/local* model, and has the same goal of understanding whether this hybrid model is more powerful than its constituent models.

To accomplish this goal, they perform theoretical analyses on multiple tasks that each exhibit a known gap between what utility is achievable in both of the classic trust models. Their results demonstrate that it is possible to solve problems in the hybrid model which cannot be solved with reasonable utility in the TCM or LM separately. This is concrete proof that for certain tasks, the hybrid model can be strictly more powerful than the classic trust models (even when compared against provably optimal mechanisms in the classic trust models).

They additionally show that there are problems which cannot be solved in the TCM or LM separately, and can be solved in the hybrid model, but only if the TCM and LM groups interact with each other. This definitively proves that for certain tasks, intergroup interaction is indeed *necessary* for achieving high utility. However, they do not analyze the privacy implications of such intergroup interaction (e.g., as we do in this chapter with intergroup privacy amplification).

Finally, they analyze a problem which *does not* significantly benefit from the hybrid model: basic hypothesis testing. They prove that if there exists a hybrid model mechanism that distinguishes between two distributions effectively, then there also exists a TCM or LM mechanism which does so nearly as effectively. This result informally hints at a lack of power of the hybrid model for the problem of mean estimation in certain settings. Specifically, the lack of power that it implies is that a hybrid mechanism cannot achieve more than a constant factor improvement over optimal mechanisms in the classic trust models. This roughly aligns with our findings for mean estimation in our setting, where our hybrid estimators improve over baseline estimators in the classic trust models, but (for practical parameters) only by a constant factor (Corollary 2.3.21).

Alternative Hybrid Models of Trust

Beyond the TCM/LM hybrid model, there are multiple alternative hybrid models in the DP literature. We discuss the two most popular and promising such models here.

The most popular is the *public/private* hybrid model of Beimel et al. [BNS13] and Ji and Elkan [JE13]. In this model, most users desire the guarantees of differential privacy, but some users have made their data available for use without requiring any privacy guarantees. In this model, some works assume that DP is achieved in the TCM [HCB16; Pap+17], while others assume that DP is achieved in the LM [XSM16; Wan+21]. In both cases, the works show that by operating in the public/private hybrid model, one can significantly improve utility relative to either model separately. Recently, theoretical works [BMA19; Bas+20a] have explored the limits of this model's power via lower bounds on the sample complexity of fundamental statistical problems.

Another powerful hybrid trust model recently introduced is the *shuffle* model, which was conceptually proposed by Bittau et al. [Bit+17] before being mathematically defined and analyzed for its DP guarantees by Cheu et al. [Che+19] and Erlingsson et al. [Erl+19]. In this model, users privately submit their data under the LM via an anonymous channel to the curator. The anonymous channel randomly permutes the users' contributions so that the curator has no knowledge of what data belongs to which user. This "shuffling" enables users to achieve improved DP guarantees over their LM guarantees in isolation. Several works have since improved the original analyses and expanded the shuffle model to achieve even greater improvements in the users' DP guarantee [Bal+19b; GPV19; Bal+19a; Gha+20a; Gha+20b; Gha+21].

Non-hybrid DP Mean Estimation

For the DP mean estimation problem in this chapter, our work leveraged straightforward baseline mean estimators in the classic trust models to enable us to obtain exact finite-sample utility expressions. However, DP mean estimation of distributions under both the TCM and LM has

been studied since the models' introductions [Blu+05; War65; DJW13], and continues to be actively studied [Fel17; KV18; Ach+18; GRS19; DJW18; Kam+19a; Kam+19b; Jos+19a; Du+20; BS19; KSU20; Gha+20b; Bis+20].

The goal of mean estimation research under both models is to maximize utility while minimizing the sample complexity by making various distributional assumptions. Some assumptions are stronger than those made in this section, such as assuming the data is drawn from a narrow family of distributions. Other assumptions are weaker, such as requiring only that the mean lies within a certain range or that higher moments are bounded. Because of the complexity of the mechanisms and their reliance on the distributional assumptions in the related works, their utility expressions are typically bounds or asymptotic rather than exact. Since we need exact finite-sample utility expressions to precisely quantify the utility of our hybrid estimator relative to the baselines, we are unable to use their estimators and assumptions. Nevertheless, the related works show a practically significant sample complexity gap between the TCM and LM in their respective settings, further motivating mean estimation in the hybrid model.

2.6 Future Directions

There are several promising directions for future work on the hybrid model in general as well as for the specific applications of heavy hitter estimation and mean estimation. We discuss these directions here.

User Behavior in the Hybrid Model

For the hybrid model, we believe the most interesting direction is exploring how a hybrid mechanism's utility may be impacted in several ways by the users' ability to self-partition into their preferred trust models. For instance, one such way that a hybrid mechanism's utility can be impacted is through distributional differences in the users' data across the two trust model groups. In analyzing the heavy hitter estimation problem with `BLENDER`, we had implicitly assumed that

each user's data in the TCM and LM groups were drawn i.i.d. from the same distribution. Then, in analyzing the mean estimation problem with our hybrid estimator family, we examined how utility is impacted when the users' data in the TCM group comes from a different distribution than the users' data in the LM group. However, both scenarios assumed that regardless of a user's group choice, the user would be guaranteed the same level of privacy; i.e., the mechanism satisfies (ϵ, δ)-DP for the user no matter their group choice. A new avenue to explore is how a hybrid mechanism's utility is impacted when the privacy level is allowed to differ between the two groups. In the LM, a similar concept has been recently introduced as the "privacy elasticity of behavior" [Dek+22], which quantifies a user's propensity to change their behavior when the privacy level is changed. In our hybrid setting, this new direction instead considers a user's propensity to change their preferred trust model when the privacy level is changed.

Depending on the extent to which users trust the curator versus their desired level of privacy, allowing the privacy level to vary between the groups could dramatically impact the hybrid mechanism's final utility. For instance, consider the scenario where all users are guaranteed $\epsilon = 1$ DP regardless of their trust model choice, and suppose 1% choose the TCM while the remaining 99% choose the LM. Furthermore, suppose that most LM users only weakly prefer the LM over the TCM, but they highly prefer a strong DP guarantee. In this scenario, it may be possible to counterintuitively *improve* the final utility of the hybrid mechanism by *decreasing* the ϵ value guaranteed to users who choose the TCM group. A decrease for the TCM group to $\epsilon = 0.95$ (while leaving the LM group's ϵ at 1) may then incentivize a new TCM opt-in rate of 10%. Although each TCM user's contribution to the group's utility would diminish, the overall utility of the TCM group would increase due to the disproportionate number of new users opting in.

Understanding this behavior in practice (in controlled lab settings or through observational studies after deployment), as well as modeling this behavior theoretically and analyzing its impact on a mechanism's utility, is a promising path to increasing the power of the hybrid model.

Applications

For the specific applications studied in this chapter, heavy hitter estimation and mean estimation, the most apparent direction for future research is in improving the sub-mechanisms that our designed hybrid mechanisms were built upon. Because the hybrid mechanisms that we designed were fairly modular in nature, the sub-mechanisms that each group utilized can be interchanged with alternative DP mechanisms without explicitly affecting how the other group should function. This means that as newer research is conducted on these problems in either the TCM or LM (in isolation) and improved DP mechanisms are designed, the new mechanisms can be trivially incorporated into our hybrid mechanisms and thus improve their utility.

Specific to the heavy hitter estimation problem with `BLENDER`, another important direction for future work is to shift from an empirical utility analysis to a theoretical utility analysis. Under certain assumptions, this could enable a priori estimates of the utility, rather than relying on costly and time-consuming simulations.

For the mean estimation problem, an important direction for future work is in relaxing the assumptions considered in this chapter (e.g., the known boundedness of data, the knowledge and finiteness of the variance, etc.). This is important because of how fundamental the mean estimation problem is in practical data analysis problems — having high-utility hybrid mean estimators for many possible distributional/knowledge settings would enable DP mechanism designers to more easily build their own complex hybrid mechanisms by leveraging simple hybrid building blocks.

Finally, we believe designing new algorithms in the hybrid model for other real-world applications in data analysis and machine learning is a fruitful direction for future work.

2.A Chapter Appendix

Deferred BLENDER Proofs

In this portion, we detail the previously deferred proofs (Section 2.2.1) for the unbiasedness of BLENDER's various estimators.

Lemma 2.2.3. *$\hat{\sigma}^2_{O,\langle q,u\rangle}$ is an unbiased variance estimate for the opt-in group's record probabilities if $m_O = 1$.*

Proof. Given the head list, the distribution of `EstimateOptinProbabilities`' estimate for a record $\langle q, u\rangle$ is given by $r_{O,\langle q,u\rangle} = p_{\langle q,u\rangle} + \frac{Y}{|D_T|}$, where $Y \sim \text{Laplace}(b_T)$ with b_T being the scale parameter and recalling that $|D_T|$ is the total number of records from the opt-in users used to estimate probabilities. The empirical estimator for $r_{O,\langle q,u\rangle}$ is $\hat{r}_{O,\langle q,u\rangle} = \frac{1}{|D_T|}\sum_{j=1}^{|D_T|} X_j + Y$, where $X_j \sim \text{Bernoulli}(p_{\langle q,u\rangle})$ is the random variable indicating whether report j was record $\langle q, u\rangle$.

The expectation of this estimator is given by $\mathbb{E}[\hat{r}_{O,\langle q,u\rangle}] = p_{\langle q,u\rangle}$. Thus, $\hat{r}_{O,\langle q,u\rangle}$ is an unbiased estimator for $p_{\langle q,u\rangle}$. We define $\hat{p}_{O,\langle q,u\rangle} = \hat{r}_{O,\langle q,u\rangle}$ to explicitly reference it as the estimator of $p_{\langle q,u\rangle}$. The variance for this estimator is

$$
\begin{aligned}
\sigma^2_{O,\langle q,u\rangle} &= \mathbb{V}[\hat{p}_{O,\langle q,u\rangle}] && (2.4)\\
&= \mathbb{V}\Big[\frac{1}{|D_T|}\Big(\sum_{j=1}^{|D_T|} X_j + Y\Big)\Big] \\
&= \frac{1}{|D_T|^2}\Big(\mathbb{V}\big[\sum_{j=1}^{|D_T|} X_j\big] + \mathbb{V}\left[Y\right]\Big) && (2.5)\\
&= \frac{1}{|D_T|^2}\Big(\sum_{j=1}^{|D_T|} \mathbb{V}\left[X_j\right] + \mathbb{V}\left[Y\right]\Big) && (2.6)\\
&= \frac{1}{|D_T|^2}\big(|D_T| \cdot p_{\langle q,u\rangle}(1 - p_{\langle q,u\rangle})\big) + 2\Big(\frac{b_T}{|D_T|}\Big)^2 \\
&= \frac{p_{\langle q,u\rangle}(1 - p_{\langle q,u\rangle})}{|D_T|} + 2\Big(\frac{b_T}{|D_T|}\Big)^2.
\end{aligned}
$$

Equality 2.5 comes from the independence between Y and all X_j. Equality 2.6 relies on an assumption of independence between X_j, X_k for all $j \neq k$ (i.e., the iid assumption discussed prior to the theorem statements).

To compute this variance, we need to use the data in place of the unknown $p_{\langle q,u\rangle}$. Using $\hat{p}_{O,\langle q,u\rangle}$ directly in place of $p_{\langle q,u\rangle}$ requires a $\frac{|D_T|}{|D_T|-1}$ factor correction (known as "Bessel's correction"[13]) to generate an unbiased estimate. Thus, the variance of each opt-in record probability estimate is: $\hat{\sigma}^2_{O,\langle q,u\rangle} = \frac{|D_T|}{|D_T|-1}\left(\frac{\hat{p}_{O,\langle q,u\rangle}(1-\hat{p}_{O,\langle q,u\rangle})}{|D_T|} + 2\left(\frac{b_T}{|D_T|}\right)^2\right)$. □

Lemma 2.2.5. *$\hat{p}_{C,\langle q,u\rangle}$ is an unbiased estimate of the clients' record probabilities.*

Proof. Reporting query/URL records is a two-stage process. A query is first selected followed by a URL being selected, thus forming a complete record. Debiasing records is similarly accomplished in two stages.

1. Debiasing query probability estimates: Let $r_{C,q}$ denote the probability that the mechanism has received query q as a report, and let p_q be the true probability of a user having query q. We want to learn p_q based on $r_{C,q}$. By the design of our mechanism,

$$\begin{aligned} r_{C,q} &= t \cdot p_q + \sum_{q' \neq q} p_{q'}(1-t)\frac{1}{k-1} \\ &= t \cdot p_q + \frac{1-t}{k-1}\sum_{q' \neq q} p_{q'} \\ &= t \cdot p_q + \frac{1-t}{k-1}(1-p_q). \end{aligned}$$

Solving for p_q in terms of $r_{C,q}$ yields $p_q = \frac{r_{C,q}-\frac{1-t}{k-1}}{t-\frac{1-t}{k-1}}$. Using the obtained data for the query $\hat{r}_{C,q}$, we estimate $p_{C,q}$ as $\hat{p}_{C,q} = \frac{\hat{r}_{C,q}-\frac{1-t}{k-1}}{t-\frac{1-t}{k-1}}$.

2. Debiasing record probability estimates: Analogously, let $r_{C,\langle q,u\rangle}$ denote the probability that the mechanism has received a record $\langle q,u\rangle$ as a report, and recall $p_{\langle q,u\rangle}$ is the record's true probability in the dataset. Then $r_{C,\langle q,u\rangle} = t \cdot t_q \cdot p_{\langle q,u\rangle} + \left(t\frac{1-t_q}{k_q-1}\right)(p_q - p_{\langle q,u\rangle}) + \left(\frac{1-t}{k-1}\cdot\frac{1}{k_q}\right)(1-p_q)$, recalling from the mechanism that k_q is the number of URLs associated with query q and t_q is the probability of truthfully reporting u given that query q was reported. Solving for $p_{\langle q,u\rangle}$ yields $p_{\langle q,u\rangle} = \frac{r_{C,\langle q,u\rangle}-\left(t\frac{1-t_q}{k_q-1}p_q+\frac{(1-t)(1-p_q)}{(k-1)k_q}\right)}{t(t_q-\frac{1-t_q}{k_q-1})}$.

[13] https://en.wikipedia.org/wiki/Bessel's_correction

Using the obtained data for the empirical report estimate $\hat{r}_{C,\langle q,u\rangle}$ together with the query estimate $\hat{p}_{C,q}$, we estimate $p_{\langle q,u\rangle}$ as $\hat{p}_{C,\langle q,u\rangle} = \frac{\hat{r}_{C,\langle q,u\rangle} - \left(t\frac{1-t_q}{k_q-1}\hat{p}_{C,q} + \frac{(1-t)(1-\hat{p}_{C,q})}{(k-1)k_q}\right)}{t(t_q - \frac{1-t_q}{k_q-1})}$. □

Lemma 2.2.6. *$\hat{\sigma}^2_{C,\langle q,u\rangle}$ is an unbiased variance estimate of the clients' record probabilities if $m_C = 1$.*

Proof. We first derive the variance estimate for the client group's query probabilities, then move on to the variance estimate for their record probabilities.

From the proof of Lemma 2.2.5, the distribution of the reported query q from the client algorithm is given by $r_{C,q} = t \cdot p_q + \frac{1-t}{k-1}(1-p_q)$, and so the true probability of query q is distributed as $p_q = \frac{r_{C,q} - \frac{1-t}{k-1}}{t - \frac{1-t}{k-1}}$. The empirical estimator for p_q is $\hat{p}_{C,q} = \frac{\hat{r}_{C,q} - \frac{1-t}{k-1}}{t - \frac{1-t}{k-1}}$, where $\hat{r}_{C,q}$ is the empirical estimator of $r_{C,q}$ defined explicitly as $\hat{r}_{C,q} = \frac{1}{|D_C|}\sum_{j=1}^{|D_C|} X_j$, where $X_j \sim \text{Bernoulli}(r_{C,q})$ is the random variable indicating whether report j was query q and recalling that $|D_C|$ is the total number of records from the client users.

The variance of $\hat{r}_{C,q}$ is

$$\begin{aligned}
\mathbb{V}[\hat{r}_{C,q}] &= \mathbb{V}\Big[\frac{1}{|D_C|}\sum_{j=1}^{|D_C|} X_j\Big] \\
&= \Big(\frac{1}{|D_C|}\Big)^2 \sum_{j=1}^{|D_C|} \mathbb{V}\left[X_j\right] && (2.7) \\
&= \big(\frac{1}{|D_C|}\big)^2 \big(|D_C| \cdot r_{C,q}(1-r_{C,q})\big) && (2.8) \\
&= \frac{r_{C,q}(1-r_{C,q})}{|D_C|},
\end{aligned}$$

where equality 2.7 relies on an assumption of independence between X_j, X_k for all $j \neq k$ (i.e., the iid assumption discussed prior to the theorem statements).

Then, the variance of $\hat{p}_{C,q}$ is

$$\sigma^2_{C,q} = \mathbb{V}[\hat{p}_{C,q}] = \mathbb{V}\Big[\frac{\hat{r}_{C,q} - \frac{1-t}{k-1}}{t - \frac{1-t}{k-1}}\Big] = \frac{r_{C,q}(1-r_{C,q})}{|D_C|\big(t - \frac{1-t}{k-1}\big)^2}.$$

To compute this variance, we need to use the data in place of the unknown $r_{C,q}$. Using $\hat{r}_{C,q}$ directly in place of $r_{C,q}$ requires including Bessel's $\frac{|D_C|}{|D_C|-1}$ factor correction to yield an unbiased estimate. Thus, the variance of the query probability estimates by the client mechanism is:

$\hat{\sigma}^2_{C,q} = \left(\frac{1}{t-\frac{1-t}{k-1}}\right)^2 \frac{\hat{r}_{C,q}(1-\hat{r}_{C,q})}{|D_C|-1}$.

We now derive the variance estimate for the record probabilities. For a given query q and corresponding URL u in head list, denote X_i^q as the indicator random variable that is 1 if user i reported query q and 0 otherwise, and similarly denote $X_i^{\langle q,u\rangle}$ as the indicator random variable that is 1 if user i reported query q and URL u and 0 otherwise. Thus, $X_i^q \sim \text{Bern}(r_{C,q})$ and $X_i^{\langle q,u\rangle} \sim \text{Bern}(r_{C,\langle q,u\rangle})$. The covariance between these two random variables is given by

$$\text{Cov}[X_i^q, X_i^{\langle q,u\rangle}] = \mathbb{E}[X_i^q X_i^{\langle q,u\rangle}] - \mathbb{E}[X_i^q]\,\mathbb{E}[X_i^{\langle q,u\rangle}] = r_{C,\langle q,u\rangle} - r_{C,\langle q,u\rangle} r_{C,q} = r_{C,\langle q,u\rangle}(1 - r_{C,q}).$$

Also, due to the i.i.d. assumption, for any other user j, we have $\text{Cov}(X_i^q, X_j^{\langle q,u\rangle}) = 0$. Thus, we have the covariance between our empirical query and record estimates as

$$\begin{aligned}
\text{Cov}[\hat{r}_q, \hat{r}_{\langle q,u\rangle}] &= \text{Cov}\left[\frac{1}{|D_C|}\sum_{i\in D_C} X_i^q, \frac{1}{|D_C|}\sum_{i\in D_C} X_i^{\langle q,u\rangle}\right] \\
&= \frac{1}{|D_C|^2}\text{Cov}\left[\sum_{i\in D_C} X_i^q, \sum_{i\in D_C} X_i^{\langle q,u\rangle}\right] \\
&= \frac{1}{|D_C|^2}\sum_{i,j\in D_C} \text{Cov}[X_i^q, X_j^{\langle q,u\rangle}] \\
&= \frac{1}{|D_C|^2}\sum_{i\in D_C} \text{Cov}[X_i^q, X_i^{\langle q,u\rangle}] \\
&= \frac{r_{C,\langle q,u\rangle}(1 - r_{C,q})}{|D_C|}.
\end{aligned}$$

Utilizing this covariance expression, we can now compute the desired variance estimate as:

$$
\begin{aligned}
&\sigma^2_{C,\langle q,u\rangle} = \mathbb{V}[\hat{p}_{C,\langle q,u\rangle}] \\
&= \mathbb{V}\left[\frac{\hat{r}_{C,\langle q,u\rangle} - \big(t\frac{1-t_q}{k_q-1}\hat{p}_{C,q} + \frac{(1-t)(1-\hat{p}_{C,q})}{(k-1)k_q}\big)}{t(t_q - \frac{1-t_q}{k_q-1})}\right] \\
&= \frac{1}{t^2(t_q - \frac{1-t_q}{k_q-1})^2}\,\mathbb{V}\left[\hat{r}_{C,\langle q,u\rangle} - \big(t\frac{1-t_q}{k_q-1}\hat{p}_{C,q} + \frac{(1-t)(1-\hat{p}_{C,q})}{(k-1)k_q}\big)\right] \\
&= \frac{1}{t^2(t_q - \frac{1-t_q}{k_q-1})^2}\,\mathbb{V}\left[\hat{r}_{C,\langle q,u\rangle} - \hat{p}_{C,q}\big(\frac{1-t}{(k-1)k_q} - t\frac{1-t_q}{k_q-1}\big)\right] \\
&= \frac{1}{t^2(t_q - \frac{1-t_q}{k_q-1})^2}\cdot \\
&\quad\left(\mathbb{V}\left[\hat{r}_{C,\langle q,u\rangle}\right] + \big(\frac{1-t}{(k-1)k_q} - t\frac{1-t_q}{k_q-1}\big)^2\,\mathbb{V}\left[\hat{p}_{C,q}\right] + 2\big(\frac{1-t}{(k-1)k_q} - t\frac{1-t_q}{k_q-1}\big)\mathrm{Cov}[\hat{p}_{C,q}, \hat{r}_{C,\langle q,u\rangle}]\right) \\
&= \frac{1}{t^2(t_q - \frac{1-t_q}{k_q-1})^2}\cdot \\
&\quad\left(\frac{r_{C,\langle q,u\rangle}(1-r_{C,\langle q,u\rangle})}{|D_C|} + \big(\frac{1-t}{(k-1)k_q} - t\frac{1-t_q}{k_q-1}\big)^2\sigma^2_{C,q} + 2\big(\frac{1-t}{(k-1)k_q} - t\frac{1-t_q}{k_q-1}\big)\frac{1}{t-\frac{1-t}{k-1}}\mathrm{Cov}[\hat{r}_{C,q}, \hat{r}_{C,\langle q,u\rangle}]\right) \\
&= \frac{1}{t^2(t_q - \frac{1-t_q}{k_q-1})^2}\cdot \\
&\quad\left(\frac{r_{C,\langle q,u\rangle}(1-r_{C,\langle q,u\rangle})}{|D_C|} + \big(\frac{1-t}{(k-1)k_q} - t\frac{1-t_q}{k_q-1}\big)^2\sigma^2_{C,q} + 2\big(\frac{1-t}{(k-1)k_q} - t\frac{1-t_q}{k_q-1}\big)\frac{1}{t-\frac{1-t}{k-1}}\frac{r_{C,\langle q,u\rangle}(1-r_{C,q})}{|D_C|}\right).
\end{aligned}
$$

Using our already computed estimates $\hat{r}_{C,q}, \hat{r}_{C,\langle q,u\rangle}$, and $\hat{\sigma}^2_{C,\langle q,u\rangle}$ (in place of $r_{C,q}, r_{C,\langle q,u\rangle}$, and $\sigma^2_{C,\langle q,u\rangle}$ respectively) and applying Bessel's correction, we obtain the stated result. □

Chapter 3

Quantifying the Privacy–Utility Trade-off

To address the second high-level challenge of this thesis (Section 1.2) that hinders DP's adoption in practice — the lack of tools for analyzing complex DP mechanisms' utilities — we present in this chapter a novel method for quantifying DP mechanisms' privacy–utility trade-offs[14]. Our new method is especially suited for complex, hyperparameterized DP mechanisms, whose designers are faced with a series of compounding challenges when attempting to quantify the mechanisms' privacy–utility trade-offs — in fact, even the *notion* of a privacy–utility trade-off for such mechanisms is ill-defined. Thus, our first contribution is to propose a formal definition for the privacy–utility trade-off of hyperparameterized mechanisms. Grounded in this definition, we then design `DPareto`, a practical method to quantify DP mechanisms' privacy–utility trade-offs using only empirical measurements by leveraging well studied multi-objective Bayesian optimization techniques. Finally, we analyze the performance of `DPareto` on several machine learning tasks involving multiple models, architectures, and optimizers and compare its performance against baseline methods. Our findings definitively establish that `DPareto` is both highly efficient and highly effective at quantifying the privacy–utility trade-off of complex, hyperparameterized DP mechanisms.

[14]This chapter is based on work in our publication [Ave+20].

3.1 Overview

Quantifying the trade-off between privacy and utility is a central topic in the DP literature. For any given utility measure, a formal analysis of a mechanism's privacy–utility trade-off quantifies how the mechanism's utility changes when its given privacy level (e.g., ϵ) is changed, yielding a curve on the privacy vs. utility plane. Since the choice of privacy level is generally regarded as a policy decision [Woo+18], quantifying this trade-off is essential to decision makers tasked with balancing privacy and utility in real-world deployments of DP mechanisms [AS19b].

Quantifying the privacy–utility trade-off is not straightforward in practice. Primarily, analytical characterizations of the trade-off are only available for relatively simple mechanisms amenable to mathematical treatment. Conducting such analyses for complex mechanisms of practical interest is often infeasible. Further, DP mechanisms have more hyperparameters than their non-private counterparts, most of which affect both privacy and utility.

> **Example:** A motivating application for this chapter is differentially private deep learning. Differentially private stochastic optimization has been employed to train feed-forward [Aba+16], convolutional [Car+18], and recurrent [McM+18] neural networks, showing that reasonable accuracies can be achieved when selecting hyperparameters carefully. The canonical non-private stochastic optimization algorithm, stochastic gradient descent (SGD) [RM51; KW52], utilizes hyperparameters that specify the learning rate, batch size, and number of epochs. The state-of-the-art differentially private variant of the SGD mechanism, DP-SGD [Aba+16], extends SGD by clipping and adding Gaussian noise to the computed gradients in order to satisfy DP. This explicitly adds two new hyperparameters — a clipping amount and a Gaussian noise scale — to ensure privacy, and these new hyperparameters also implicitly affect utility.

In practice, tuning a DP mechanism's hyperparameters to achieve a desirable privacy–utility trade-off can be an arduous task, especially when utility analyses for the mechanism are loose

or unavailable. Exacerbating this challenge is the fact that for some hyperparameterized DP mechanisms, the privacy level is not *explicitly* specified as one of the inputs. Instead, the mechanism's privacy level must first be calculated from the hyperparameters. For example, in DP-SGD, the privacy level must be calculated using the computationally intensive "moments accountant" method [Aba+16]. Moreover, the same level of privacy can be obtained through the hyperparameters in multiple ways; e.g., a given privacy level can be obtained for DP-SGD by either adjusting the Gaussian noise amount or the gradient clipping amount while leaving the other fixed.

Taken together, these challenges can make achieving any specific level of privacy or utility with a hyperparameterized DP mechanism difficult. As a result, no practical methods exist to quantify hyperparameterized mechanisms' entire privacy–utility trade-offs — even the definition of the "privacy–utility trade-off" is unclear in the presence of such hyperparameters. Consequently, the central problem that we consider in this chapter of the thesis is:

> *How can we rigorously define a hyperparameterized DP mechanism's privacy–utility trade-off, and then how can we design a practical method for quantifying it?*

We address this problem by first formalizing the quantification of a mechanism's privacy–utility trade-off as a Pareto front estimation problem (Section 3.2). We then develop `DPareto`, a method to estimate the privacy–utility Pareto front using Bayesian optimization techniques (Section 3.3). To determine the effectiveness of `DPareto`, we empirically evaluate it on a variety of machine learning tasks (Section 3.4).

3.2 Defining the Privacy–Utility Trade-Off

Our first contribution is to concretely define what a hyperparameterized DP mechanism's privacy–utility trade-off is for a given utility measure. We preface this contribution by motivating the definition. We then detail the concrete formal definition, which enables us to reformulate the central

problem of this chapter in a more precise way. Finally, to make our definition more intuitive, we provide comprehensive illustrative examples from both machine learning and differential privacy.

Motivation Our proposed privacy–utility trade-off definition is motivated by what a practitioner using a hyperparameterized DP mechanism primarily needs to understand about the mechanism's privacy and utility. For instance, we described how in DP-SGD, there can be many settings of hyperparameters that achieve the same privacy level but that achieve different utility levels. It is likely that many of these hyperparameter settings do not correspond to high utility and are therefore not of interest to the practitioner. Instead, the practitioner is mainly interested in which hyperparameter settings correspond to optimal utility for the DP mechanism at each privacy level. Moreover, the practitioner needs both the privacy and utility to be concretely quantified; for example, there may exist a theoretically optimal privacy analysis for the mechanism, but if it is unknown or unavailable to the practitioner, then they will instead utilize the best analysis tools that they have at their disposal (e.g., the moments accountant method). Grounded in this, we abstractly define the privacy–utility trade-off of hyperparameterized mechanisms as: the quantification of the optimal utility that the mechanism can attain at any privacy level realizable by the chosen privacy analysis.

3.2.1 The Privacy–Utility Pareto Front

We now formalize the abstract definition of a hyperparameterized DP mechanism's privacy–utility trade-off using the notion of Pareto fronts.

To discuss the trade-off between privacy and utility for a given problem, we consider a hyperparameterized family of mechanisms $\mathcal{M} = \{\mathcal{M}_\lambda : \mathcal{D}^n \to \mathcal{W} \mid \lambda \in \Lambda\}$. Here, $\lambda \in \Lambda$ is a particular setting of hyperparameters, and $\mathcal{M}_\lambda : \mathcal{D}^n \to \mathcal{W}$ is a hyperparameterized DP mechanism which takes n records from $\mathcal{D}$ and outputs a value in $\mathcal{W}$. For example, in the context of a

machine learning application, the family $\mathcal{M}$ could consist of a set of DP machine learning mechanisms which take as input a training dataset $D = (d_1, \ldots, d_n)$ containing n example-label pairs $d_i = (x_i, y_i) \in \mathcal{D}$ and produce as output the parameters $w \in \mathcal{W} \subset \mathbb{R}^\ell$ of a predictive model.

To capture the privacy–utility trade-off across $\mathcal{M}$, we introduce two oracles to model the effect of hyperparameter changes on the privacy and utility of $\mathcal{M}_\lambda$.

- **Privacy Oracle:** We define the privacy oracle $\mathsf{P}_\delta : \Lambda \to [0, +\infty]$ to be a function that given a choice of hyperparameters λ returns a value $\epsilon = \mathsf{P}_\delta(\lambda)$ such that $\mathcal{M}_\lambda$ satisfies (ϵ, δ)-DP for a given δ. A concrete example of a privacy oracle, and the one that we primarily use in this work, is the moments accountant technique proposed for DP-SGD [Aba+16]. We note that the privacy oracle may not provide an exact, theoretically tight privacy accounting for the mechanism; however, we assume the privacy oracle is *sound* in the sense that mechanism's true privacy guarantee will never be worse than what the oracle returns (i.e., if an optimal analysis would conclude that the mechanism's privacy level for a given choice of hyperparameters λ is ϵ^* , then $\epsilon^* \leq \mathsf{P}_\delta(\lambda)$).

- **Utility Oracle:** We define the instance-specific utility oracle $\mathsf{U}_D : \Lambda \to [0, 1]$ to be a function that given a choice of hyperparameters λ returns some utility measure of the output distribution of $\mathcal{M}_\lambda(D)$. A concrete example of a utility oracle, and the one that we primarily use in this work, is the classification accuracy of a trained model on a test dataset. In practice, unlike the privacy oracle, the utility oracle will generally be noisy due to intentional randomness in the mechanism (e.g., to ensure differential privacy, or to initialize the model's parameters) as well as randomness inherent in measuring the utility of the output distribution itself (e.g., using a held-out test dataset to approximate the trained model's generalizability to a larger data space).

As informally discussed in this section's motivation, the practical notion of a mechanism's privacy–utility trade-off is defined by the hyperparameter settings for $\mathcal{M}_\lambda$ that simultaneously

achieve maximal privacy and utility on a given input D. We formalize this using the concept of a *Pareto front*. Informally, a Pareto front of a collection of points $\Gamma \subset \mathbb{R}^p$ contains all points in Γ where none of the coordinates can be decreased without increasing other coordinates while remaining inside Γ. Formally, it is defined as follows.

Definition 3.2.1. Let $\Gamma \subset \mathbb{R}^p$ and $u, v \in \Gamma$. We say that u *dominates* v if $u_i \leq v_i$ for all $i \in [p]$, and we write $u \preceq v$. The *Pareto front* of Γ is the set of all non-dominated points:

$$\mathcal{PF}(\Gamma) = \{u \in \Gamma \mid v \not\preceq u,\ \forall\, v \in \Gamma \setminus \{u\}\}.$$

Grounded in this definition, we define the privacy–utility trade-off of a mechanism $\mathcal{M}$ with hyperparameter space Λ and oracles P_δ and U_D to be the Pareto front $\mathcal{PF}(\Gamma)$ of the 2-dimensional set $\Gamma = \{(\mathsf{P}_\delta(\lambda), 1 - \mathsf{U}_D(\lambda)) \mid \lambda \in \Lambda\}$[15]. Figure 3.1 provides a high level illustration of a Pareto front generated from evaluating privacy and utility oracles on a set of hyperparameter settings.

Leveraging this new Pareto front definition of a mechanism's privacy–utility trade-off, we formally restate this chapter's central problem as follows.

> *For any given hyperparameter space Λ, privacy oracle P_δ, and instance-specific utility oracle U_D, how can we design a practical method to estimate a DP mechanism's privacy–utility Pareto front $\mathcal{PF}(\Gamma) = \{(\mathsf{P}_\delta(\lambda),\ 1 - \mathsf{U}_D(\lambda) \mid \lambda \in \Lambda\}$?*

Remarks on the Oracles We conclude this portion by providing two brief remarks to clarify the definitions and roles of the privacy and utility oracles.

Parametrizing the privacy oracle P_δ in terms of a fixed δ stems from the convention that ϵ is considered the most important privacy parameter[16], whereas δ is chosen to be a negligibly small

[15] Since the points in the Pareto front are traditionally those that *minimize* each dimension, we use anti-utility for the second coordinate to maintain consistency. Specifically, we use $1-\mathsf{U}_D(\lambda)$ because utility can often be normalized to lie in $[0, 1]$ in practice. However, such normalization is not required in this chapter, and any measure of anti-utility (e.g., $-\mathsf{U}_D(\lambda)$) could be used instead.

[16] This choice is without loss of generality since there is a connection between the two parameters guaranteeing the existence of a valid ϵ for any valid δ [BBG18].

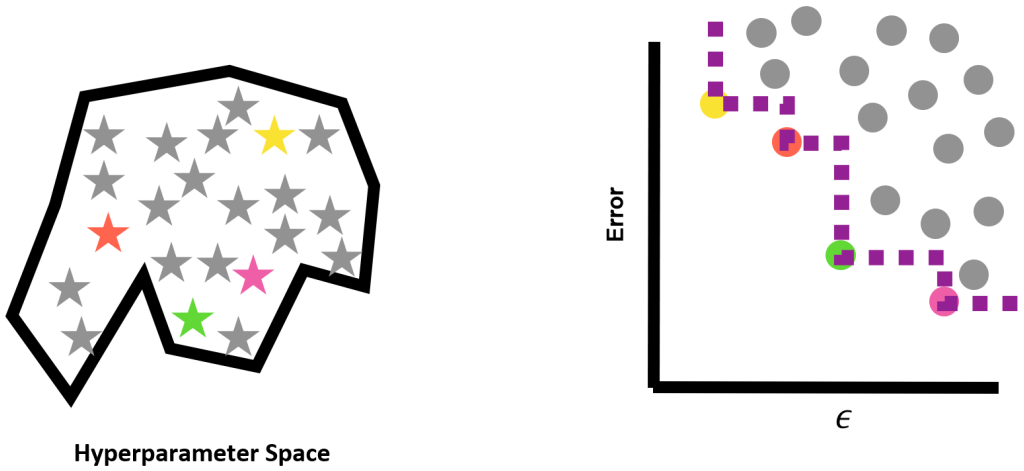


Figure 3.1: *Left:* A complex space of hyperparameter settings, with several points arbitrarily selected from it. *Right:* The privacy–utility Pareto front generated from privacy and utility oracle evaluations of each hyperparameter setting. Colored points represent Pareto optimal points, whereas grey points are dominated by at least one Pareto optimal point.

value ($\delta \ll 1/n$). This choice is also aligned with recent uses of DP in machine learning; i.e., the privacy analysis is conducted under the framework of Rényi DP [Mir17], and the reported privacy guarantee is then obtained a posteriori and converted to a standard (ϵ, δ)-DP guarantee for some fixed δ [Aba+16; GSC17; McM+18; Fel+18; WBK19]. In particular, in our subsequent evaluations with gradient perturbation for stochastic optimization methods (Section 3.4), we implement the privacy oracle using the moments accountant technique proposed by Abadi et al. [Aba+16] coupled with the tight bounds provided by Wang et al. [WBK19] for Rényi DP amplification by subsampling without replacement. More generally, privacy oracles can take the form of analytic formulas or numerically optimized calculations, but future advances in empirical or black-box evaluation of DP guarantees could also play the role of privacy oracles.

Parametrizing the utility oracle U_D by a fixed input is a choice justified by the type of applications we analyze in our experiments (Section 3.4). Other applications may require different variations, which our framework can easily accommodate by extending the definition of the utility oracle. We also stress that since the mechanisms in $\mathcal{M}$ are randomized, the utility $\mathsf{U}_D(\lambda)$ is

a property of the output *distribution* of $\mathcal{M}_\lambda(D)$. This means that, in practice, one may have to implement the oracle approximately, e.g., via sampling. In particular, in our subsequent experiments we use a test dataset to measure the utility of a hyperparameter setting by evaluating $\mathcal{M}_\lambda(D)$ a fixed number of times R to obtain model parameters $w_1, \ldots, w_R$, and then compute $\mathsf{U}_D(\lambda)$ to be the average accuracy of the models on the test set.

3.2.2 Two Illustrative Examples

To concretely illustrate the oracles and Pareto front concept, we consider two distinct examples: private logistic regression and the sparse vector technique. We select these two examples because they are both computationally light and have a low-dimensional hyperparameter space, which enables us to compute their privacy–utility Pareto fronts nearly exactly via a fine-grained grid search over the hyperparameter space. For brevity, we subsequently refer to Pareto fronts computed via fine-grained grid search as the "exact" or "true" Pareto fronts with respect to the potentially inexact (e.g., loose or noisy) privacy and utility oracles.

3.2.2.1 Private Logistic Regression

Our first illustrative example is a DP variant of a common machine learning problem: privately training a simple logistic regression model with ℓ_2 regularization. For this problem, we use the ADULT dataset [Koh+96] partitioned into training and test sets.

We train the model with differential privacy using the method of Wu et al. [Wu+17, Algorithm 2] with default parameters[17]. In this method, the model is trained with mini-batch projected SGD, then Gaussian noise is added to the output (the model's parameters) to ensure privacy. The only hyperparameters that need to be specified in this example are the regularization amount γ and

[17]Default parameters are the smoothness (i.e., Lipschitz and strong convexity parameters of the loss) and the learning rate.

the standard deviation of the Gaussian noise σ, while the rest are fixed[18]. In this example, both hyperparameters affect privacy and accuracy simultaneously.

We implement the privacy and utility oracles as follows. For the privacy oracle, we compute the mechanism's global sensitivity according to Wu et al. [Wu+17, Algorithm 2] and find the ϵ corresponding to a fixed $\delta = 10^{-6}$ using the exact analysis of the Gaussian mechanism (Theorem 1.1.8). To implement the utility oracle, we evaluate the classification error of the model on the test set, averaging over 50 runs for each setting of the hyperparameters.

To obtain the exact Pareto front for this problem, we perform a fine-grained grid search over $\gamma \in [10^{-4}, 10^{0}]$ and $\sigma \in [10^{-1}, 10^{1}]$. The Pareto front and its corresponding hyperparameter settings are displayed in Figure 3.2, along with the values returned by the privacy and utility oracles across the entire range of hyperparameters.

3.2.2.2 Sparse Vector Technique

Our second illustrative example is the *sparse vector technique* (SVT) [Dwo+09], a DP mechanism for interactively posing m queries against a fixed sensitive database and releasing only the indices of queries that exceed a certain threshold. The naming of the mechanism reflects the fact that it is specifically designed to achieve good utility when only a small number of queries are expected to be above the threshold. As a foundational DP mechanism, SVT has found applications in a number of problems, and several variants of it have been proposed [LSL17].

Algorithm 3.1 details our construction of a non-interactive variant of Lyu et al.'s SVT [LSL17, Algorithm 7]. The standard SVT mechanism is parameterized by the target privacy ϵ. In order to illustrate mechanisms whose hyperparameters affect both privacy and utility simultaneously, we propose a non-standard construction of the SVT mechanism. Specifically, ours takes as input a total noise level b and is tailored to answer m binary queries $q_i : \mathcal{D}^n \to \{0, 1\}$ with sensitivity $\Delta = 1$ and fixed threshold $T = 1/2$. The privacy and utility of the mechanism are controlled by the noise level b and the bound on the number of answers C. In this example, either increasing

[18]The fixed hyperparameters are the mini-batch size $m = 1$ and number of epochs $T = 10$.

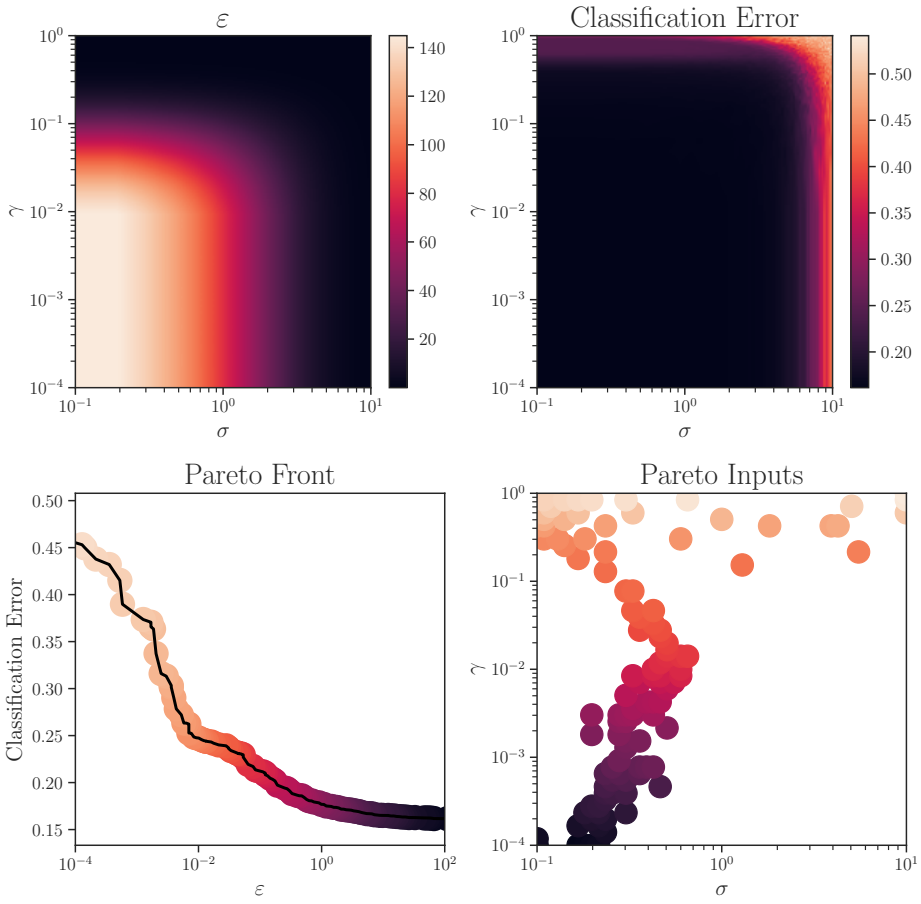


Figure 3.2: *Top:* Values returned by the privacy and utility oracles across a range of hyperparameters in the private logistic regression example. *Bottom:* The Pareto front and its corresponding set of input points.

b or decreasing C improves privacy and diminishes utility. The noise level is split across two parameters b_1 and b_2 controlling how much noise is added to the threshold and to the query answers respectively[19].

Algorithm 3.1 Sparse Vector Technique (SVT)

Input

- D: Sensitive dataset.
- $q_1, \ldots, q_m$: m binary queries.

Hyperparameters

- b: Noise scale.
- C: Upper-bound on number of answers.

Body

Let $c = 0$ and $w = (0, \ldots, 0) \in \{0, 1\}^m$.
Let $\rho \sim \text{Laplace}(b_1)$, where $b_1 = b/(1 + (2C)^{1/3})$.
Let $b_2 = b - b_1$.
for $i \in [m]$ **do**
  Let $\nu \sim \text{Laplace}(b_2)$.
  if $q_i(D) + \nu \geq \frac{1}{2} + \rho$ **then**
    Set $w_i = 1$ and $c = c + 1$.
    if $c \geq C$, **break**
  end if
end for
Return: w.

We implement the privacy and utility oracles as follows. The privacy analysis of Algorithm 3.1 (whose proof we defer to the end-of-chapter Appendix 3.A) yields the following closed-form privacy oracle for our mechanism: $\mathsf{P}_0 = (1 + (2C)^{1/3})(1 + (2C)^{2/3})b^{-1}$. For the utility oracle, we use the F_1-*score* between the vector of true answers $(q_1(D), \ldots, q_m(D))$ and the vector w returned by SVT. This measures how well the mechanism identifies the support of the queries that return 1, while penalizing for false positives and false negatives. This is a non-standard utility measure for SVT: prior utility analyses of SVT focus on providing an interval around the threshold outside which the output is guaranteed to have no false positives or false negatives [DR+14].

[19]The noise level split is based on Lyu et al.'s suggested privacy budget allocation [LSL17, Section 4.2].

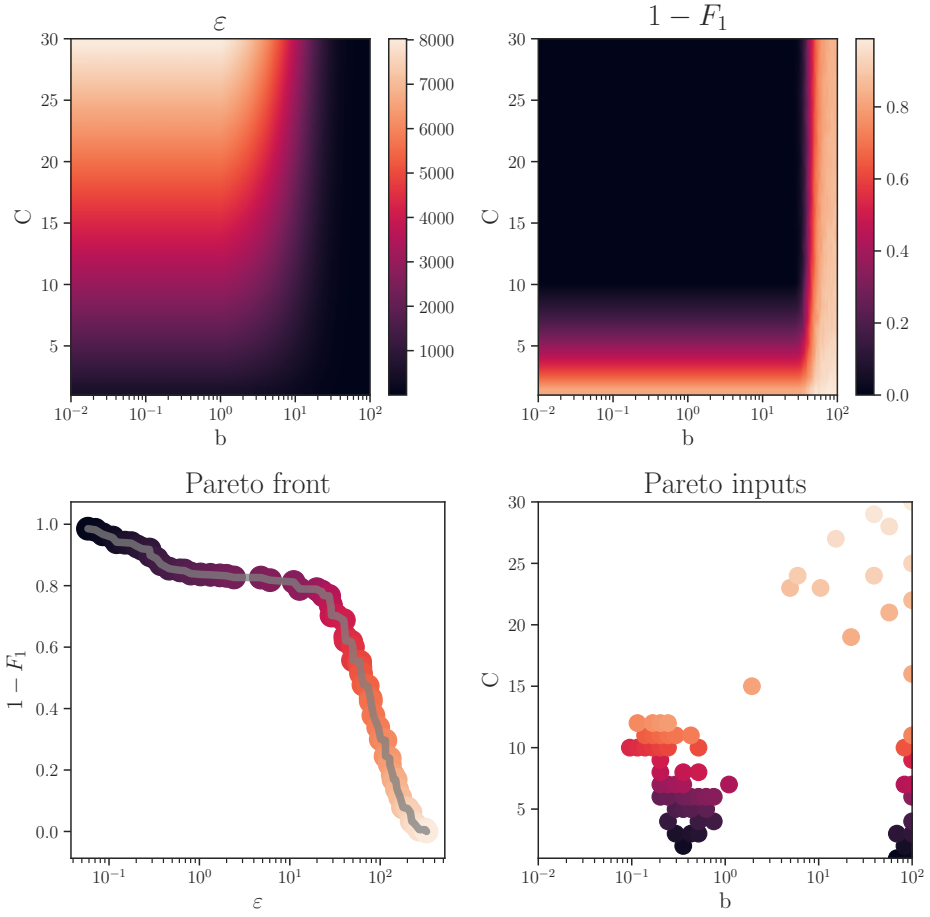


Figure 3.3: *Top:* Values returned by the privacy and utility oracles across a range of hyperparameters in the SVT example. *Bottom:* The Pareto front and its corresponding set of input points.

Our utility measure is more fine-grained and relevant for practical applications, although no theoretical utility analysis of SVT in terms of F_1-score exists in literature.

In this example, we set $m = 100$ and pick queries at random such that exactly 10 of them return a 1. Since the utility of SVT is sensitive to the query order, we evaluate the utility oracle by running the mechanism 50 times with a random query order and compute the mean F_1-score. The Pareto front and its corresponding hyperparameter settings are displayed in Figure 3.3, along with the values returned by the privacy and utility oracles across the entire range of hyperparameters.

3.3 Estimating the Privacy–Utility Pareto Front

With DP mechanisms' privacy–utility trade-offs formally defined in terms of their privacy–utility Pareto fronts, our second contribution is `DPareto`, a method for empirically estimating these Pareto fronts. To begin, we define what it means to empirically estimate a Pareto front, and how the utility of an estimated Pareto front is measured. We then provide a brief background on multi-objective Bayesian optimization (BO), which forms the foundation of `DPareto`. Putting together these two components, we then detail our `DPareto` method in full. To make `DPareto`'s process more intuitive, we revisit the mechanisms from our earlier illustrative examples and use them to demonstrate how the various components of `DPareto` work together to estimate the mechanisms' Pareto fronts.

3.3.1 Empirical Pareto Fronts and their Utility Measures

For practical DP mechanisms (e.g., using DP-SGD for a real-world machine learning task), computing the *exact* privacy–utility Pareto front is infeasible due to the expensive computation of both the privacy and utility oracles. Thus, the privacy–utility Pareto front must instead be *estimated* using a small number of oracle evaluations. However, even when the number of oracle evaluations is fixed, different methodologies for where to evaluate the oracles (i.e., methods for selecting hyperparameters) will generally yield different estimated Pareto fronts. Therefore, we must define how to measure the utility of an empirically estimated Pareto front.

Towards this, we leverage the *hypervolume* of an estimated Pareto front [EK08; CDD14a; Knu+17], defined as follows. Let $\mathsf{E} = \{(\lambda, \mathsf{P}_\delta(\lambda), \mathsf{U}_D(\lambda))\}_{i=1}^k$ be an arbitrary set of k hyperparameters and corresponding privacy and utility oracle evaluations. Let $\mathcal{PF}(\Gamma_\mathsf{E})$ be the empirical Pareto front corresponding to these evaluations, where $\Gamma_\mathsf{E} = \{(\mathsf{P}_\delta(\lambda), \mathsf{U}_D(\lambda))\}_{i=1}^k$ denotes the privacy and utility oracle evaluations from E. Let $\bar{v} \in \mathbb{R}^p$ be some chosen "anti-ideal" point[20]; i.e.,

[20]For instance, in the private logistic regression example, the anti-ideal point could correspond to the worst-case ϵ and worst-case classification error.

a point known a prioi to be dominated by all points in Γ_{E}. The hypervolume $\mathsf{HV}_{\bar{v}}(\mathcal{PF}(\Gamma_{\mathsf{E}}))$ of the region dominated by the Pareto front and bounded by the anti-ideal point is formally defined as:

$$\mathsf{HV}_{\bar{v}}(\mathcal{PF}(\Gamma_{\mathsf{E}})) = \mu(\{v \in \mathbb{R}^p \mid v \preceq \bar{v},\ \exists u \in \mathcal{P}\ u \preceq v\}),$$

where μ denotes the standard Lebesgue measure on $\mathbb{R}^p$. The choice of which specific anti-ideal point to use does not make a difference regarding the Pareto optimality of any other potential point, it only changes the measured hypervolume of a Pareto front. However, this measurement is crucial when comparing multiple different Pareto fronts. Thus, for brevity, we assume the anti-ideal point is fixed and henceforth drop it from our notation.

This quantification enables us to compare the quality of different methods' estimated Pareto fronts. Moreover, in this chapter, we make no assumptions about the behavior of an empirical Pareto front or the relationship between points on an empirical Pareto front (e.g., we do not assume concavity/convexity). As a result, the empirical Pareto fronts are always *conservative*, meaning that they never overlap (or overestimate in terms of hypervolume) the true underlying Pareto front. Therefore, the hypervolume utility measure additionally corresponds directly to more accurate Pareto fronts; i.e., the larger the hypervolume of an estimated Pareto front, the closer that estimated Pareto front is to the true Pareto front. This is important for scenarios where the true Pareto front cannot be efficiently computed, but where multiple candidate empirical Pareto fronts are available for comparison — simply put, the one with the largest hypervolume is best. Figure 3.4 illustrates this concept.

3.3.2 Multi-Objective Bayesian Optimization

Bayesian optimization (BO) [Moč75] is a strategy for sequential decision making useful for optimizing expensive-to-evaluate black-box objective functions, and has become increasingly relevant in machine learning due to its success in optimizing model hyperparameters [SLA12; Jen+17].

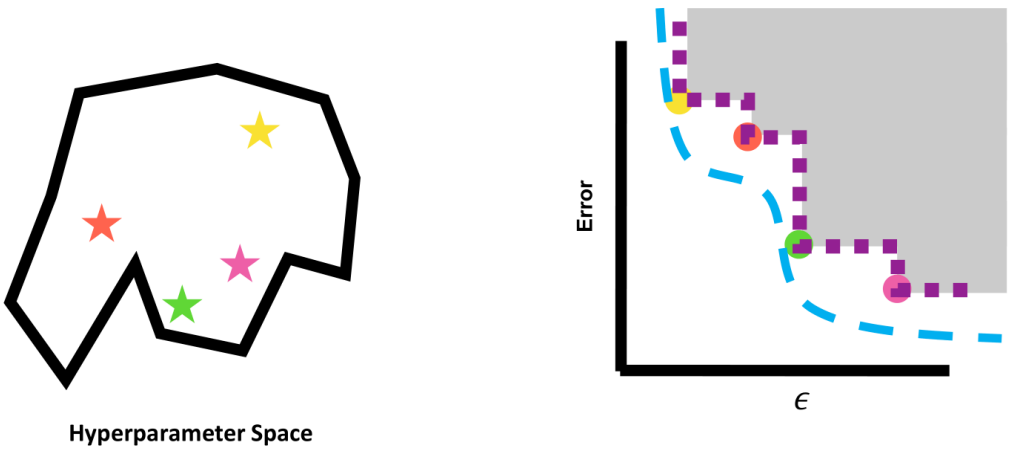


Figure 3.4: *Left:* Hyperparameter settings that correspond to Pareto optimal points in the privacy–utility plane. *Right:* The empirical privacy–utility Pareto front corresponding to the hyperparameter settings' privacy and utility oracle evaluations. The grey shaded area represents the estimated Pareto front's dominated region from which its hypervolume is computed. The blue curve represents the mechanism's true (but unknown) underlying privacy–utility Pareto front.

In its standard form, BO is used to estimate a minimum of an objective function $f(\lambda)$ on some subset $\Lambda \subseteq \mathbb{R}^p$ of a Euclidean space by generating a sequence of evaluations of the objective at locations $\lambda_1, \ldots, \lambda_k$. Each point in this sequence is generated by building a surrogate model of the objective function using prior evaluations of the objective function, then applying a prespecified criterion to select a new location λ_{k+1} based on the surrogate model. In the single-objective case, a common choice is to select the location that, in expectation under the surrogate model, gives the best improvement to the current estimated minimum.

When used in multi-objective optimization problems where a single point may not exist which minimizes all objective functions simultaneously, BO aims to estimate a Pareto front using a minimal number of evaluations. This makes multi-objective BO a clear candidate to help achieve our goal of estimating the privacy–utility Pareto front in a practical way. Although in this chapter we only work with two objective functions, we detail here the general case of minimizing p objectives $f_1, \ldots, f_p$ simultaneously. This generalization can be used, for instance, to introduce the running time of the mechanism as a third objective to be traded off against privacy and utility.

3.3.2.1 Standard Bayesian Optimization Loop

At a high level, BO works as follows. Let $\lambda_1, \ldots, \lambda_k$ be a set of locations in Λ, and let $\mathcal{V} = \{v_1, \ldots, v_k\}$ be the set such that each $v_i \in \mathbb{R}^p$ is the vector of objective evaluations $(f_1(\lambda_i), \ldots, f_p(\lambda_i))$. The following is then repeatedly iterated over until a budget to collect new locations has been exhausted.

1. Fit a surrogate model of the objectives $f_1(\lambda) \ldots, f_p(\lambda)$ using the available evaluations $\mathsf{E} = \{(\lambda_i, v_i)\}_{i=1}^k$. The standard approach is to use a Gaussian Process (GP) [RW05].

2. For each objective f_j, calculate the predictive distribution over $\lambda \in \Lambda$ using the surrogate model. If GPs are used, the predictive distribution of each output can be fully characterized by their mean $m_j(\lambda)$ and variance $s_j^2(\lambda)$ functions, which can be computed in closed form.

3. Use the posterior distribution of the surrogate model to form an acquisition function $\alpha(\lambda; \mathcal{I})$, where $\mathcal{I}$ represents the evaluations E and the GP posterior conditioned on E.

4. Collect the next evaluation point λ_{k+1} at the (numerically estimated) global maximum of $\alpha(\lambda; \mathcal{I})$.

There are two key aspects of any BO method: (1) the surrogate model of the objectives, and (2) the acquisition function $\alpha(\lambda; \mathcal{I})$. In this chapter, we use two independent GPs as the surrogate models, one for each objective[21]. We now provide a detailed overview of the acquisition functions that we are interested in for estimating Pareto fronts.

Acquisition Function with Pareto Front Hypervolume To collect new points when estimating a Pareto front, we define an acquisition function $\alpha(\lambda; \mathcal{I})$ using the hypervolume measure (Section 3.3.1). The acquisition function's purpose is to select a location which will most improve the current estimated Pareto front's hypervolume.

[21] Surrogate model generalizations that utilize multi-output GPs [ARL12] are possible, and are a promising direction for future work.

Such an acquisition function is designed as follows. Let $\mathcal{PF}(\mathcal{V})$ be the Pareto front computed with the objective evaluations in $\mathcal{I}$, and let $\bar{v} \in \mathbb{R}^p$ be a chosen anti-ideal point. First, we define the change in hypervolume given a new point $v \in \mathbb{R}^p$:

$$\Delta_{\mathcal{PF}}(v) = \mathsf{HV}(\mathcal{PF}(\mathcal{V} \cup \{v\})) - \mathsf{HV}(\mathcal{PF}(\mathcal{V})).$$

This quantity is positive only if v lies in the set $\tilde{\Gamma}$ of points non-dominated by $\mathcal{PF}(\mathcal{V})$. Therefore, the *probability of improvement* (PoI) over the current Pareto front when selecting a new hyperparameter λ can be computed using the GP surrogate models trained on $\mathcal{I}$ as follows:

$$\begin{aligned}\mathsf{PoI}(\lambda) &= \Pr[(f_1(\lambda), \ldots, f_p(\lambda)) \in \tilde{\Gamma} \mid \mathcal{I}] \\ &= \int_{v \in \tilde{\Gamma}} \prod_{j=1}^{p} \phi_j(\lambda; v_j) dv_j \ ,\end{aligned}$$

where $\phi_j(\lambda; \cdot)$ is the predictive Gaussian density for f_j with mean $m_j(\lambda)$ and variance $s_j^2(\lambda)$.

The $\mathsf{PoI}(\lambda)$ function accounts for the probability that a given $\lambda \in \Lambda$ has to improve the Pareto front, and it can be used as a criterion to select new points. However, in this chapter, we opt to use an enhanced variant of PoI due to its superior computational and practical properties: the *hypervolume probability of improvement* (HVPoI) [CDD14b]. The HVPoI is given by

$$\alpha(\lambda; \mathcal{I}) = \Delta_{\mathcal{PF}}(m(\lambda)) \cdot \mathsf{PoI}(\lambda) \ , \tag{3.1}$$

where $m(\lambda) = (m_1(\lambda), \ldots, m_p(\lambda))$. This acquisition function weighs the probability of improving the Pareto front with a measure of how much improvement is expected (estimated using the GP surrogate models). The HVPoI has been shown to work well in practice, and efficient implementations have recently been published [Knu+17].

3.3.3 Defining `DPareto`

Our proposed method to estimate privacy–utility Pareto fronts of hyperparameterized DP mechanisms, `DPareto`, utilizes these recent advances in practical multi-objective BO [CDD14a; Knu+17].

We present `DPareto` in Algorithm 3.2, and it works as follows. Initially, `DPareto` is provided with a small set E of k_0 randomly sampled hyperparameters and their corresponding privacy and utility oracle evaluations. `DPareto` then selects k new hyperparameters iteratively in the following steps. First, it fits GP models to transformed evaluations in E (we describe what these transformations are shortly). It then maximizes the HVPoI acquisition function to obtain a new hyperparameter setting and corresponding privacy and utility oracle evaluations. The new hyperparameter setting and oracle evaluations are then added into E, and the process repeats. Once the k new evaluations have been collected, `DPareto` returns the empirical Pareto front constructed from the full set of privacy and utility oracle evaluations.

Algorithm 3.2 `DPareto`

Input

- Λ: Hyperparameter space.
- $\mathsf{P}_\delta, \mathsf{U}_D$: Privacy and utility oracles.
- $\bar{v}$: Anti-ideal point.
- k_0: Number of initial random evaluations to collect.
- k: Number of new points to collect with BO.

Body

Initialize evaluation set $\mathsf{E} = \emptyset$.
for $i = 1, \ldots, k_0$ **do**
  Let λ_i be a sample from a random distribution over Λ.
  Let v_i be the oracle evaluations $(\mathsf{P}_\delta(\lambda_i), 1 - \mathsf{U}_D(\lambda_i))$.
  Append (λ_i, v_i) to evaluation set E.
end for
for $i = k_0 + 1, \ldots, k_0 + k$ **do**
  Fit GPs to transformed privacy and utility evaluations using E.
  Let λ_i be the $\arg\max$ of the HVPoI acquisition function with anti-ideal point $\bar{v}$.
  Let v_i be the oracle evaluations $(\mathsf{P}_\delta(\lambda_i), 1 - \mathsf{U}_D(\lambda_i))$.
  Append (λ_i, v_i) to evaluation set E.
end for
Return: Pareto front $\mathcal{PF}(\{v \mid (\lambda, v) \in \mathsf{E}\})$.

The output domains for the privacy and utility oracles may not be well-modeled by a GP, which models outputs on the entire real line. For instance, the output domain for the privacy oracle is $[0, +\infty]$. The output domain for the utility oracle depends on the chosen measure of utility. A common choice of utility oracle for ML tasks is accuracy, which has output domain $[0, 1]$. Thus, neither the privacy nor utility oracles are well-modeled by a GP as is. Therefore, in both cases, we transform the outputs so that we are modeling a GP in the transformed space[22]. For privacy, we use a simple log transform. For accuracy, we use a logit transform $\text{logit}(x) = \log(x) - \log(1 - x)$. With this, both oracles have transformed output domain $[-\infty, +\infty]$. Although it is possible to *learn* the transformation using Warped GP [SGR04], which has the advantage of having both the covariance matrix and the nonlinear transformation learned simultaneously under the same probabilistic framework, we choose to use fixed transformations for simplicity and efficiency.

With the exception of the transformations related to the privacy and utility oracles, nothing about `DPareto` as described in Algorithm 3.2 is, by design, specific to differential privacy. In Section 3.6, we briefly describe modifications to `DPareto` that may improve its performance by incorporating aspects specific to differential privacy.

To increase adoption of `DPareto` and to enable others to more easily build on it, we have publicly released the code for its implementation and experiments under an open-source license[23].

3.3.4 Two Illustrative Examples: Revisited

We revisit the examples discussed in Section 3.2.2 to concretely illustrate how the components of `DPareto` work together to estimate the privacy–utility Pareto front, and how the estimated Pareto front compares to the true Pareto front.

[22] Depending on the exact privacy and utility values observed, the GPs may be able to model the oracles reasonably well without transformation. In this case, we hypothesize that `DPareto` would have similar performance compared to if the transformation had been performed. However, transforming the observed values more closely matches the GPs' foundational assumptions a priori, so we expect the corresponding learned GPs to generally be a better fit when incorporating these transformations.

[23] https://github.com/amzn/differential-privacy-bayesian-optimization

3.3.4.1 Private Logistic Regression

For the logistic regression example, we initialize the GP models with $k_0 = 250$ random hyperparameter pairs (γ_i, σ_i). γ_i takes values in $[10^{-4}, 10^0]$ and σ_i takes values in $[10^{-1}, 10^1]$, both sampled uniformly on a logarithmic scale. The privacy and mean utility of the trained models corresponding to each sample are computed, and GPs are fit to these values as surrogate models for each oracle.

Figure 3.5 shows the results of the surrogate model evaluations and oracle evaluations, as well as the true Pareto front and `DPareto`'s estimated Pareto front. The predicted means of these surrogate models are shown in the top row of the figure. Comparing directly to the oracles' true values in Figure 3.2, we observe that the surrogate models have modeled them well in the high σ and γ regions, but are still learning the low regions. The bottom-left of Figure 3.5 shows the exact Pareto front of the problem, along with the output values of the initial sample and the corresponding empirical Pareto front. The empirical Pareto front lies almost exactly on the true one, except in the extremely-high privacy region ($\epsilon < 10^{-2}$). This indicates that the selection of random points (γ_i, σ_i) was already quite good outside of this region.

The goal of `DPareto` is to select new points in the input domain whose outputs will bring the empirical front closer to the true one. This is the purpose of the acquisition function; the bottom-right of the figure shows the HVPoI acquisition function evaluated over all (γ_i, σ_i) pairs. The acquisition function's maximizer, marked with a star, is used as the next location to evaluate the oracles. Given the current surrogate models, the HVPoI seems to be making a sensible choice; i.e., it is selecting a point where ϵ and classification error are both predicted to have relatively low values, looking to improve the upper-left region of the Pareto front.

3.3.4.2 Sparse Vector Technique

For the sparse vector technique example, we initialize the GP models with $k_0 = 250$ random hyperparameter pairs (C_i, b_i). The C_i values are sampled uniformly in the interval $[1, 30]$, and

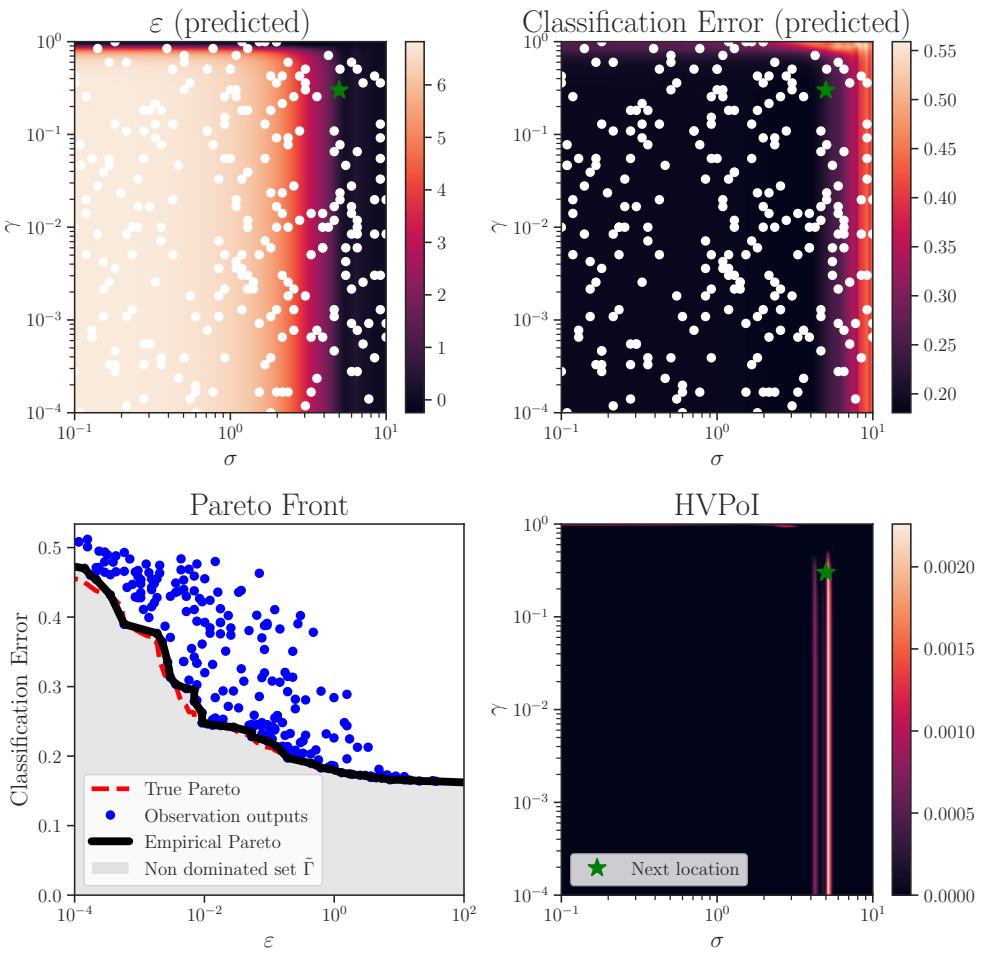


Figure 3.5: *Top:* Mean predictions of the privacy (ϵ) and the utility (classification error) oracles using their respective GP models in the private logistic regression example. The locations of the $k_0 = 250$ sampled points are plotted in white. *Bottom left:* Empirical and true Pareto fronts. *Bottom right:* HVPoI and the selected next location.

the b_i values are sampled uniformly in the interval $[10^{-2}, 10^2]$ on a logarithmic scale. The privacy and utility values are computed for each of the samples, and GPs are fit to these values as surrogate models for each oracle.

Figure 3.6 shows the results of the surrogate model evaluations and oracle evaluations, as well as the true Pareto front and `DPareto`'s estimated Pareto front. The predicted means of these surrogate models are shown in the top row of the figure. We observe that both surrogate models have modeled their oracles reasonably well, comparing directly to the oracles' true values in Figure 3.3. The bottom-left of Figure 3.6 shows the exact Pareto front of the problem, along with the output values of the initial sample and the corresponding empirical Pareto front. The empirical Pareto front lies very close to the true one, which indicates that the selection of points (C_i, b_i) is already quite good.

`DPareto` uses the HVPoI function to select new points in the input domain whose outputs will bring the empirical Pareto front closer to the true one. The bottom-right of the figure shows this function evaluated over all (C_i, b_i) pairs. The function's maximizer, marked with a star, is used as the next location to evaluate the oracles. Given the current surrogate models, the HVPoI appears to be making a sensible choice: selecting a point where ϵ is predicted to have a medium value and $1 - F_1$ is predicted to have a low value, looking to improve the gap in the lower-right corner of the Pareto front.

3.4 Evaluating `DPareto`

With `DPareto` defined, our final contribution is to answer the question of how well `DPareto` performs. We first define how we measure and contextualize the utility of `DPareto`. We then provide the high-level details of the experimental setup for estimating the Pareto fronts of a variety of machine learning models trained with DP stochastic optimization mechanisms. Finally, we detail the concrete empirical evaluations and discuss their results.

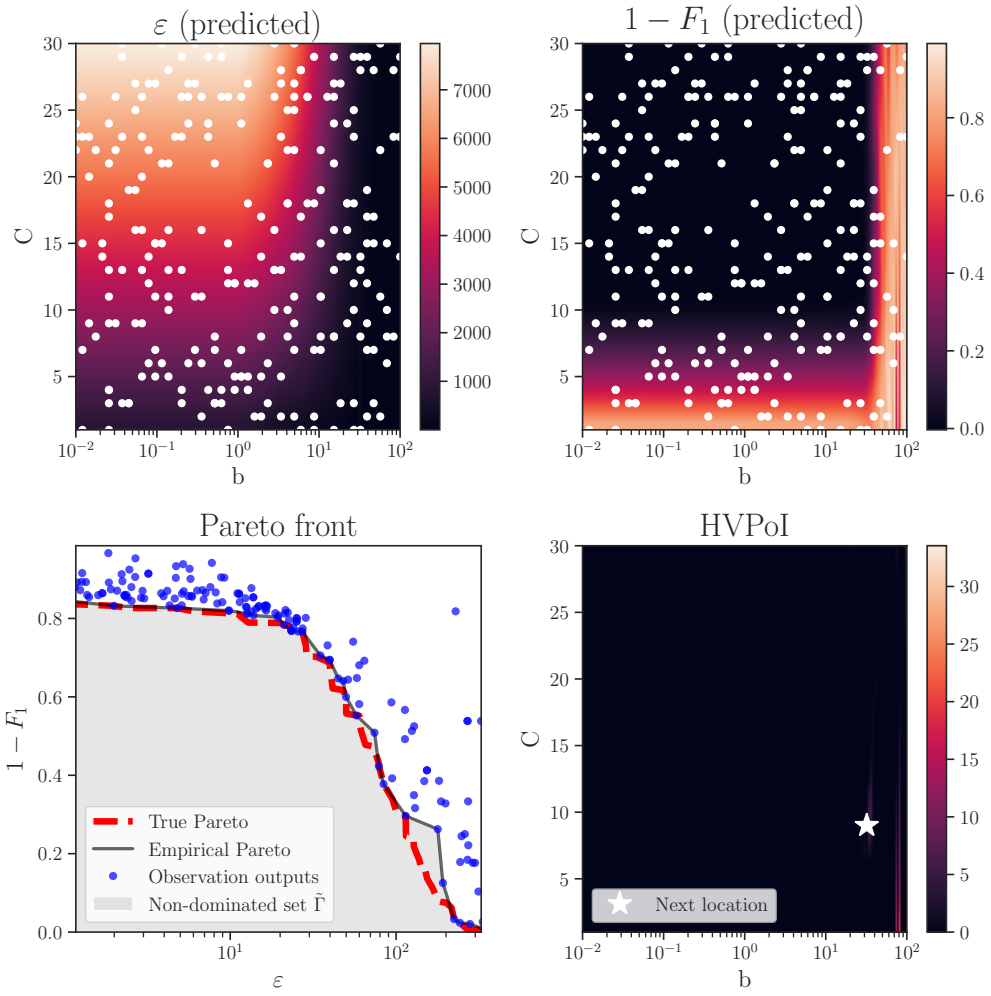


Figure 3.6: *Top:* Mean predictions of the privacy (ϵ) and the utility ($1 - F_1$) oracles using their respective GP models in the sparse vector technique example. The locations of the $k_0 = 250$ sampled points are plotted in white. *Bottom left:* Empirical and true Pareto fronts. *Bottom right:* HVPoI and the selected next location.

3.4.1 Utility Measures and Baseline Methods

For any privacy–utility Pareto front estimation method, we measure its performance in two ways: the method's effectiveness and its efficiency. We define a method's *effectiveness* as how accurately it is able to estimate the DP mechanism's true Pareto front using a fixed number of oracle evaluations. Specifically, we measure this accuracy by evaluating the hypervolume of the method's estimated privacy–utility Pareto front (Section 3.3.1). Analogously, we define a method's *efficiency* as how many hyperparameter evaluations are needed to yield an estimated privacy–utility Pareto front which achieves a fixed level of accuracy.

To understand how effectively and efficiently `DPareto` estimates privacy–utility Pareto fronts, we must put its utility into context by comparing against the utility of baseline methods. Contextualizing `DPareto`'s utility is particularly important for the application of differentially private deep learning, since obtaining a true Pareto front is computationally prohibitive. Because no prior methods exist to estimate the privacy–utility Pareto fronts of hyperparameterized DP mechanisms, we define two simple baseline methods: random search and grid search. These respectively function by evaluating the privacy and utility oracles for hyperparameters that had been randomly sampled from a distribution or selected according to a predefined grid. We utilize two distributions for random search: the uniform distribution, and a distribution carefully constructed to attempt to induce the most favorable results for random search. The latter distribution was designed by reviewing literature [Aba+16; McM+18] in addition to our experience training these models. In each of our experiments, random search using this carefully constructed distribution outperforms random search with the uniform distribution as well as grid search. Thus, for brevity, we subsequently only discuss random search with the carefully constructed distribution, and omit the others.

3.4.2 Evaluation Setup

Our goal is to evaluate `DPareto`'s effectiveness and efficiency for estimating privacy–utility Pareto fronts on a variety of machine learning tasks. Towards this, we concretely define each of the requisite components; i.e., the datasets used to train the models, the optimization domains of the hyperparameters, and the DP stochastic optimization mechanisms.

3.4.2.1 Datasets

We analyze two classic problems: binary classification of income with the ADULT dataset [Koh+96], and multiclass classification of handwritten digits with the MNIST dataset [LeC+98]. The ADULT dataset is composed of 123 binary demographic features, with the task of predicting whether each individual in the dataset has income above or below \$50k. It has 40k points in the training set and 1.6k points in the test set. The MNIST dataset is composed of 28×28 gray-scale images, each representing a single digit 0-9, with the task of predicting the digit. It has 60k images in the training set and 10k in the test set.

3.4.2.2 Models

For the ADULT dataset, we consider logistic regression (LogReg) and linear support vector machine (SVM) models. With these, we evaluate the privacy–utility trade-off induced by the choice of model and DP optimization mechanism (DP-SGD vs. DP-Adam, detailed subsequently in Section 3.4.2.4). When using the MNIST dataset, we fix the DP optimization mechanism as DP-SGD, but use a more expressive multilayer perceptron (MLP) model and evaluate the privacy–utility trade-off induced by the choice of network architecture. The first model, MLP1, has a single hidden layer with 1000 neurons. This is the same model architecture as used by Abadi et al. [Aba+16], but without a differentially private PCA dimensionality reduction pre-step. The second model, MLP2, has two hidden layers with 128 and 64 units. In both cases, the models use ReLU activations.

Algorithm	Dataset	Epochs (T)	Lot Size (m)	Learning Rate (η)	Noise Variance (σ^2)	Clipping Norm (L)
LogReg+SGD	ADULT	$[1, 64]$	$[8, 512]$	$[5\times10^{-4}, 5\times10^{-2}]$	$[0.1, 16]$	$[0.1, 4]$
LogReg+Adam	ADULT	$[1, 64]$	$[8, 512]$	$[5\times10^{-4}, 5\times10^{-2}]$	$[0.1, 16]$	$[0.1, 4]$
SVM+SGD	ADULT	$[1, 64]$	$[8, 512]$	$[5\times10^{-4}, 5\times10^{-2}]$	$[0.1, 16]$	$[0.1, 4]$
MLP1+SGD	MNIST	$[1, 400]$	$[16, 4000]$	$[1\times10^{-3}, 5\times10^{-1}]$	$[0.1, 16]$	$[0.1, 12]$
MLP2+SGD	MNIST	$[1, 400]$	$[16, 4000]$	$[1\times10^{-3}, 5\times10^{-1}]$	$[0.1, 16]$	$[0.1, 12]$

Table 3.1: Optimization domains used in each of the `DPareto` experimental evaluations.

Hyperparameter	Distribution	Parameters	Type	Accept Range
Epochs	Uniform	$a = 1, b = 64$	Integer	$[1, 64]$
Lot Size	Normal	$\mu = 128, \sigma = 64$	Integer	$[8, 512]$
Learning Rate	Shifted Exp.	$\lambda = 10$, shift $= 1\times10^{-3}$	Real	$[1\times10^{-3}, 1\times10^{-1}]$
Noise Variance	Shifted Exp.	$\lambda = 1\times10^{-1}$, shift $= 1\times10^{-1}$	Real	$[1\times10^{-1}, 16]$
Clipping Norm	Shifted Exp.	$\lambda = 1\times10^{-1}$, shift $= 1\times10^{-1}$	Real	$[1\times10^{-1}, 4]$

Table 3.2: ADULT sampling distributions for random search.

3.4.2.3 Hyperparameter Optimization Domain

Table 3.1 defines the optimization domain Λ for each of the different experiments, which all hyperparameter selection methods (i.e., `DPareto`, random search, and grid search) operate within.

For experiments on both the ADULT and MNIST datasets, we have carefully constructed distributions for random search in order to generate as favorable results for it as possible. We constructed these distributions by reviewing literature (namely Abadi et al. [Aba+16] and McMahan et al. [McM+18]) in addition to our experience from training these DP models. The precise distributions are detailed in Tables 3.2 and 3.3.

The Pareto fronts generated by random search using these constructed distributions have significantly greater hypervolume than those yielded by random search using the naive strategy of sampling from the uniform distribution, thereby justifying the choice of these distributions.

Hyperparameter	Distribution	Parameters	Type	Accept Range
Epochs	Uniform	$a = 1, b = 400$	Integer	$[1, 400]$
Lot Size	Normal	$\mu = 800, \sigma = 800$	Integer	$[16, 4000]$
Learning Rate	Shifted Exp.	$\lambda = 10$, shift $= 1\times10^{-3}$	Real	$[1\times10^{-3}, 5\times10^{-1}]$
Noise Variance	Shifted Exp.	$\lambda = 5\times10^{-1}$, shift $= 1\times10^{-1}$	Real	$[1\times10^{-1}, 16]$
Clipping Norm	Shifted Exp.	$\lambda = 5\times10^{-1}$, shift $= 1\times10^{-1}$	Real	$[1\times10^{-1}, 12]$

Table 3.3: MNIST sampling distributions for random search.

3.4.2.4 DP Stochastic Optimization Mechanisms

We perform experiments using privatized variants of two popular optimization algorithms, SGD and Adam. SGD proceeds iteratively, where on each iteration, it estimates the gradient using a single example (or small batch of examples) picked uniformly at random (without replacement) [Bot10]. Adam [KB15] extends SGD by computing adaptive estimates of lower-order moments.

As a privatized version of SGD, we use DP-SGD, detailed in Algorithm 3.3. DP-SGD is a mini-batched SGD implementation with clipped gradients and Gaussian noise. This mechanism is similar to Abadi et al.'s [Aba+16], but differs in two ways. First, it utilizes sampling without replacement to generate fixed-size mini-batches, rather than using Poisson sampling with a fixed probability which generates variable-sized mini-batches. Using fixed-size mini-batches is a more natural approach, and more closely aligns with standard practice in non-private ML. Second, for the privacy oracle, we use the moments accountant implementation of Wang et al. [WBK19] which supports sampling without replacement. In Algorithm 3.3, the function $\mathsf{clip}_L(v)$ acts as the identity if $\|v\|_2 \leq L$, and otherwise returns $(L/\|v\|_2)v$. This clipping operation ensures that $\|\mathsf{clip}_L(v)\|_2 \leq L$ so that the ℓ_2-sensitivity of any gradient to a change in one datapoint in d is always bounded by L/m.

We then design the DP-Adam mechanism in the same way that the non-private Adam optimizer is extended from SGD. At the time of this work, this was the first known implementation of DP-Adam; now, this optimizer is a standard part of DP ML libraries. Our privatized version of Adam is given in Algorithm 3.4, and uses the same gradient perturbation technique as DP-SGD. Here the notation $\tilde{g}^{\odot 2}$ denotes the vector obtained by squaring each coordinate of $\tilde{g}$. DP-Adam uses three numerical constants that are not present in DP-SGD: κ, β_1 and β_2. To simplify our empirical evaluations, we fix those constants to the defaults suggested in Kingma and Ba [KB15].

Algorithm 3.3 DP-SGD

Input

- D: dataset of n points $(d_1, \ldots, d_n)$.

Hyperparameters

- η: Learning rate.
- m: Mini-batch size.
- T: Number of epochs.
- σ^2: Gaussian noise variance.
- L: Clipping norm bound.

Body

Initialize $w = 0$.
for $t = 1, \ldots, T$ **do**
  for $k = 1, \ldots, n/m$ **do**
    Let $S \subset [n]$ be a size m uniformly random sample without replacement.
    Compute $\bar{g} = \frac{1}{m} \sum_{i \in S} \mathsf{clip}_L(\nabla \ell(d_i, w))$.
    Let $\tilde{g} = \bar{g} + \frac{2L}{m} Y$, where $Y \sim \text{Gaussian}(0, \sigma^2 I)$.
    Update $w = w - \eta \tilde{g}$.
  end for
end for
Return: w.

Algorithm 3.4 DP-Adam

Input

- D: dataset of n points $(d_1, \ldots, d_n)$.

Hyperparameters

- η: Learning rate.
- m: Mini-batch size.
- T: Number of epochs.
- σ^2: Gaussian noise variance.
- L: Clipping norm bound.

Body

Fix $\kappa = 10^{-8}$, $\beta_1 = 0.9$, and $\beta_2 = 0.999$.
Initialize $w = 0$, $\mu = 0$, $\nu = 0$, and $i = 0$.
for $t = 1, \ldots, T$ **do**
  for $k = 1, \ldots, n/m$ **do**
    Let $S \subset [n]$ be a size m uniformly random sample without replacement.
    Compute $\bar{g} = \frac{1}{m} \sum_{i \in S} \mathsf{clip}_L(\nabla \ell(d_i, w))$.
    Let $\tilde{g} = \bar{g} + \frac{2L}{m} Y$, where $Y \sim \text{Gaussian}(0, \sigma^2 I)$.
    Update $\mu = \beta_1 \mu + (1 - \beta_1)\tilde{g}$, $\nu = \beta_2 \nu + (1 - \beta_2)\tilde{g}^{\odot 2}$, and $i = i + 1$.
    De-bias $\hat{\mu} = \mu / (1 - \beta_1^i)$ and $\hat{\nu} = \nu / (1 - \beta_2^i)$.
    Update $w = w - \eta \hat{\mu} / (\sqrt{\hat{\nu}} + \kappa)$.
  end for
end for
Return: w.

3.4.3 Empirical Evaluations

With the experimental setup in place, we empirically evaluate DPareto's effectiveness and efficiency on a variety of machine learning tasks. To begin, we explicitly compare DPareto's performance to that of the random search and grid search baseline methods. We follow this comparison with a discussion on the computational overhead incurred by DPareto. We then briefly examine DPareto's variability across multiple executions. Finally, we demonstrate DPareto's versatility by using it as an analysis tool to quantify the performance of various combinations of models and DP optimizers. Taken together, our findings in this section show that DPareto is a practically useful method for quantifying the privacy–utility trade-offs of hyperparameterized DP mechanisms.

3.4.3.1 Comparing DPareto to Baseline Methods

We compare DPareto's utility first against the random search baseline, then against the grid search baseline.

DPareto vs. Random Search We evaluate DPareto and the random search method on several combinations of models, DP optimizers, and datasets, plotting the results in Figure 3.7.

In this figure, the top two plots show how the Pareto fronts' hypervolumes expand as new points are sampled. In nearly every experiment, DPareto yields a greater hypervolume than random search — a direct indicator that DPareto has estimated the Pareto front to a higher degree of accuracy. This can be seen by examining the bottom left plot of the figure, which directly shows both methods' estimated Pareto fronts of the MLP2 model. Specifically, while the random search method only marginally improves over its initial random points, DPareto is able to thoroughly explore the high-privacy regime (i.e., small ϵ). The bottom right plot of the figure compares DPareto's Pareto front given 256 sampled points against the random search method given significantly more sampled points, 1500. While both approaches yield similar Pareto fronts, the efficiency of DPareto is particularly highlighted by the points that are *not* on the front. Namely,

Model + DP-Optimizer	Mean Difference	**95% C.I.**
LogReg+SGD	0.158	$(0.053, 0.264)^*$
LogReg+Adam	0.439	$(0.272, 0.607)^*$
SVM+SGD	0.282	$(0.161, 0.402)^*$

Table 3.4: Mean hypervolume differences between `DPareto` and 19 independent repetitions of 256 iterations of random search. Two-sided 95% confidence intervals (C.I.) for these differences, as well as t-tests for the mean, are included. Asterisks indicate significance at the $p < 0.001$ level.

nearly all the points chosen by `DPareto` are close to its estimated Pareto front, whereas many points chosen by random search are nowhere near its estimated Pareto front.

To formally establish the utility benefit of `DPareto` over random search, we perform a statistical analysis using experiments on the ADULT dataset. Specifically, we perform 19 new repetitions of the random search method, with each repetition being budgeted 256 sampled points (to match the number of `DPareto` points). For each repetition, we compute the resulting Pareto front's hypervolume, then compute the hypervolume difference to `DPareto`'s Pareto front. Under the mild assumption that `DPareto` is deterministic[24], we then compute the two-sided confidence intervals for these differences. We also compute the t-statistic for these differences being zero, finding that all are highly significant ($p < 0.001$). These results are shown in Table 3.4. This demonstrates that the observed differences between both methods' Pareto fronts are in fact statistically significant.

`DPareto` vs. Grid Search The random search baseline method generally outperforms the grid search baseline method, so we minimize our exposition of grid search results in this chapter. However, for completeness, we highlight one experiment using grid search with two different grid sizes — both of which perform significantly worse than `DPareto`.

For this experiment, we define hyperparameter ranges as the limiting values from Table 3.2's distribution. We first evaluate a grid size of 3 values per hyperparamater; this corresponds to 243

[24]This assumption is not strictly true, since `DPareto` is seeded with a random set of points. However, running an equal number of repetitions of `DPareto` would be an extremely costly exercise with results expected to be nearly identical.

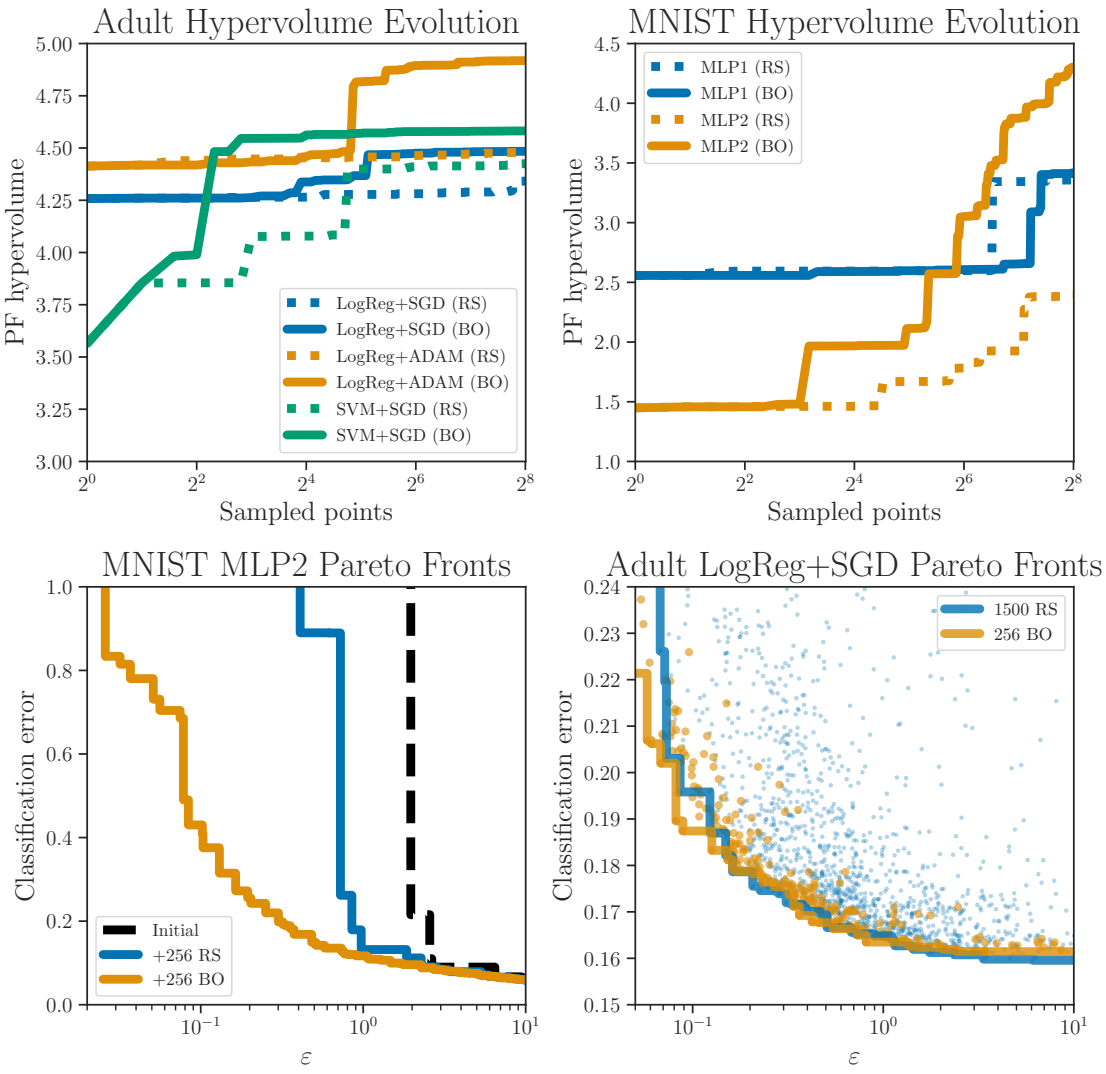


Figure 3.7: *Top:* Hypervolumes of the Pareto fronts computed by the various models, optimizers, and architectures on the ADULT and MNIST datasets (respectively) by both `DPareto` (marked BO) and random search (marked RS). *Bottom left:* Pareto fronts learned for the MLP2 architecture on the MNIST dataset with `DPareto` and random search, including the shared points they were both initialized with. *Bottom right:* ADULT dataset `DPareto` sampled points and corresponding Pareto front compared with the larger set of random search points and corresponding Pareto front.

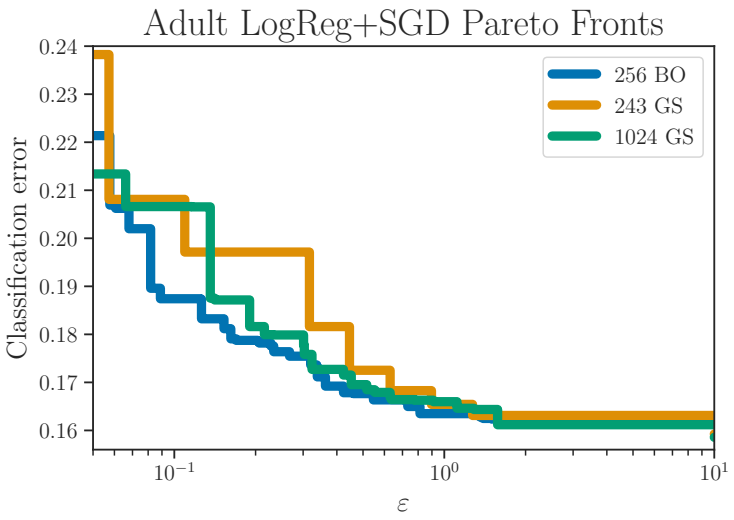


Figure 3.8: Grid search experiment results (marked GS) compared with DPareto's Bayesian optimization approach (marked BO).

total hyperparamater settings, which is approximately the same amount that DPareto is budgeted. We then evaluate a grid size of 4 values per hyperparameter; this corresponds to 1024 hyperparameter settings, which is approximately 4 times more than DPareto is budgeted. As can be seen in Figure 3.8, DPareto outperforms grid search even when significantly more hyperparameter settings are evaluated.

3.4.3.2 Computational Overhead of DPareto

Although the empirical evaluations show that DPareto produces high-quality Pareto fronts more efficiently than the random search and grid search baselines, we must examine the computational cost it incurs. Namely, we are interested in the running time of DPareto, excluding the oracle evaluation time (i.e., the model training time and moments accountant runtime).

Towards this, for experiments on both datasets, we measure the time that DPareto takes to (a) initialize the GP models with the 16 random points, plus (b) iteratively select the subsequent 256 hyperparameter settings and incorporate their corresponding privacy and utility results into

the GP models. For both the ADULT and MNIST datasets, despite the differences in their hyperparameter domains and their observed privacy and utility values, DPareto's overhead remains fairly consistent at approximately 45 seconds of total wall-clock time. This represents a negligible fraction of the total Pareto front computation time for either dataset. Specifically, it accounts for less than 0.1% of the total time for estimating the ADULT Pareto fronts, and less than 0.01% for the MNIST Pareto fronts. Thus, we conclude that DPareto's negligible overhead is more than offset by its improved Pareto fronts.

We remark that although the overhead is negligible, DPareto does have a shortcoming relative to traditional methods: it is an inherently sequential process which cannot be easily parallelized. Random search and grid search, on the other hand, can be trivially parallelized to an arbitrarily high degree which is bounded only by one's computational resources. Improving upon this facet of DPareto is beyond the scope of this chapter; however, it is a promising direction for future work.

3.4.3.3 DPareto's Variability

Understanding the variability of DPareto — i.e., the extent to which its estimated Pareto fronts change between independent executions — is important for practical deployments. In order to understand its variability, recall that in our experiments, we implement the utility oracle by repeatedly running mechanism $\mathcal{M}_\lambda$ with a fixed choice of hyperparameters, and then reporting the average utility across runs. However, using these same runs, we can also take the best and worst utilities observed for each choice of hyperparameters.

Figure 3.9 displays the estimated Pareto fronts from considering the best and worst runs in addition to the Pareto front obtained from the average over runs. We find that the variability of the estimated Pareto front is small, typically less than 3 percentage points in classification error. Moreover, we generally observe higher variability in the high-privacy regime (i.e., small ϵ). This is expected, since greater privacy is achieved by increasing the variance of the noise added to

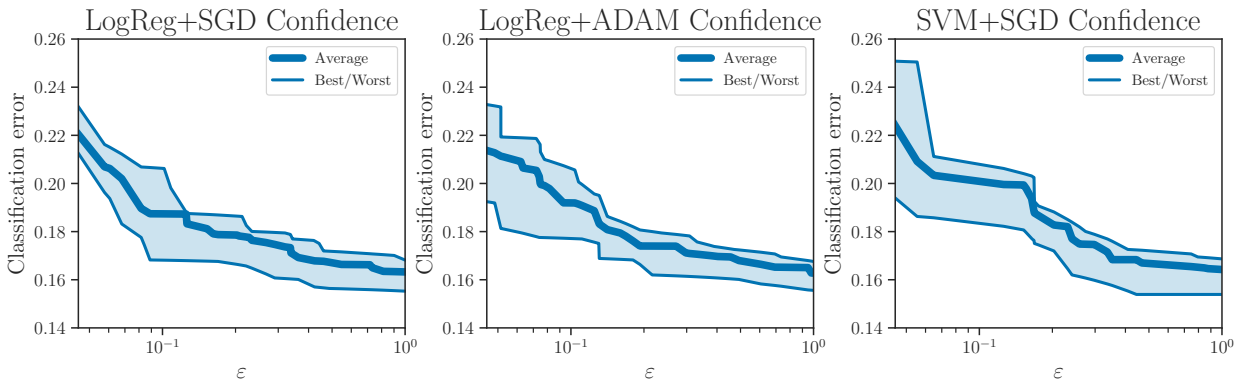


Figure 3.9: Variability of `DPareto`'s estimated Pareto fronts across models and optimizers on the ADULT dataset.

the model's gradients during training. These types of plots can be useful to decision makers who need to understand what amount of variability they should expect in practice from `DPareto`.

3.4.3.4 `DPareto`'s Versatility

One of the primary purposes of these empirical evaluations is to demonstrate the versatility of `DPareto` as an analysis tool, by using it to compare multiple approaches to the same problem. In Figure 3.10, the left plot shows Pareto fronts of the ADULT dataset for multiple optimizers (DP-SGD and DP-Adam) as well as multiple models (logistic regression and SVM), and the right plot shows Pareto fronts of the MNIST dataset for different model architectures (MLP1 and MLP2). We find that on the ADULT dataset, the logistic regression model optimized using DP-Adam is nearly always better than the other model/optimizer combinations. We also find that on the MNIST dataset, although both architectures perform similarly in the low-privacy regime, the MLP2 architecture significantly outperforms the MLP1 architecture in the high-privacy regime. These findings demonstrate that analysts and practitioners can use `DPareto` to efficiently create similar Pareto fronts in order to perform privacy–utility trade-off comparisons.

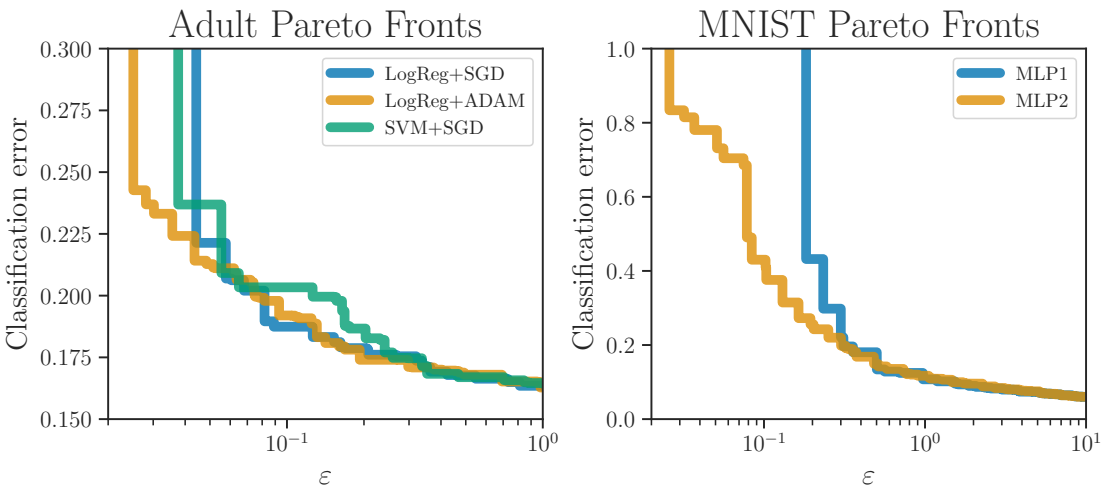


Figure 3.10: *Left:* Pareto fronts for combinations of models and optimizers on the ADULT dataset. *Right:* Pareto fronts for different MLP architectures on the MNIST dataset.

3.5 Related Works

While this chapter presents the first examination of DP mechanisms' privacy–utility trade-offs using multi-objective optimization and Pareto fronts, there are several works on adjacent topics that merit discussion.

Non-private optimization `DPareto` is built upon an active area of multi-objective optimization research on efficiently computing Pareto fronts *without* regards to privacy. `DPareto`'s point-selection process aligns with Couckuyt et al. [CDD14b], but other approaches may provide promising alternatives for improving `DPareto`. For example, Zuluaga et al. [ZKP16] propose an acquisition function that focuses on a uniform approximation of the Pareto front instead of a hypervolume based acquisition function. However, their technique does not apply out-of-the-box to the problems that we consider because it assumes a discrete hyperparameter space.

Several aspects of this chapter are related to recent work on single-objective optimization. For non-private single-objective optimization, there is an abundance of recent work in machine learning on hyperparameter selection, using BO [Kle+17; Gol+17] or other methods [Li+17] to maximize a model's utility.

U.S. Decennial Census The threat model and outputs of the `DPareto` algorithm are closely aligned with the methodology used by the U.S. Census Bureau to choose the privacy parameter ϵ for their deployment of DP to release data from the 2020 decennial census. In particular, the Census Bureau combines a graphical approach to represent the privacy–utility trade-off for their application [GAP18] together with economic theory to pick a particular point to balance the trade-off [AS19a]. Their graphical approach works with Pareto fronts identical to the ones computed by our algorithm, which they construct using data from previous censuses [Abo18a]. Although the specifics of their hyperparameter tuning are not entirely clear, we infer that their chosen hyperparameters are primarily related to post-processing steps, and therefore only affect utility[25].

ML and DP Recently, several related problems at the intersection of machine learning and differential privacy have emerged regarding hyperparameter selection and utility.

One such problem is how to perform the hyperparameter tuning process in a privacy-preserving way. Kusner et al. [Kus+15] and subsequently Smith et al. [Smi+18] use BO to find near-optimal hyperparameter settings for a given model while preserving the privacy of the data during the utility evaluation stage. Aside from the single-objective focus of this setting, our case is significantly different in that we are primarily interested in *training* the models with DP, not in protecting the privacy of the data used to evaluate an already-trained model.

Another problem is how to choose utility-maximizing hyperparameters when privately training models. When privacy is independent of the hyperparameters, this reduces to the non-private hyperparameter optimization task. However, two variants of this question do not have this trivial reduction. The first variant inverts the stated objective to study the problem of maximizing privacy given constraints on the final utility [Lig+17; Ge+19]. The second variant, closely aligning with this chapter's setting, studies the problem of choosing utility-maximizing, but privacy-dependent, hyperparameters. This is particularly challenging, since the privacy's dependence on

[25] Or, in the case of invariant forcing, the chosen hyperparameters only impact privacy in ways which are not quantifiable within standard DP theory.

the hyperparameters may be non-analytical and computationally expensive to determine. Approaches to this variant have been studied [MA18; Vee18]; however, the proposed strategies are 1) based on heuristics, 2) only applicable to the differentially private SGD problem, and 3) do not provide a computationally efficient way to find the Pareto optimal points for the privacy–utility trade-off of a given model. Wu et al. [Wu+17] provide a practical analysis-backed approach to privately training utility-maximizing models (again, for the case of SGD with a fixed privacy constraint), but hyperparameter optimization is naively performed using grid-search. By contrast, this chapter provides a computationally efficient way to *directly* search for Pareto optimal points for the privacy–utility trade-off of arbitrary hyperparameterized algorithms.

Another important problem at the intersection of DP and ML revolves around the DP "selection" or "maximization" problem [CHS14]. This problem asks how to choose an item (from a predefined universe) to maximize a data-dependent function while still protecting the privacy of that data. For this problem, Liu and Talwar [LT19] provided a way to choose hyperparameters that approximately maximize the utility of a given differentially private model in a way that protects the privacy of both the training and test data sets. Based on this, Mohapatra et al. [Moh+22] then improved on this work by using a Renyi DP [Mir17] analysis. Subsequently, Papernot and Steinke [PS] followed by Koskela and Kulkarni [KK23] built on these works to devise novel strategies for performing DP hyperparameter tuning with low privacy and computational costs. However, this only optimizes utility with fixed privacy — it does not address our problem of directly optimizing for the selection of hyperparameters that generate privacy–utility points which fall on the Pareto front.

The final problem is regarding data-driven algorithm configuration. Specifically, the problem is how to tune the hyperparameters of combinatorial optimization algorithms while maintaining DP [BDV18]. The setting considered in Balcan et al. [BDV18] assumes there is an underlying distribution of problem instances, and a sample from this distribution is used to select hyperparameters that will have good computational performance on future problem instances sampled

from the same distribution. In this case, the authors consider a threat model where the whole sample of problem instances used to tune the algorithm needs to be protected. A similar problem of data-driven algorithm selection has been considered, where the problem is to choose the best algorithm to accomplish a given task while maintaining the privacy of the data used [Kot+17]. For both, only the utility objective is being optimized, assuming a fixed constraint on the privacy.

3.6 Future Directions

For both `DPareto` and the more general problem of quantifying the privacy–utility trade-off of DP mechanisms, there are several interesting directions for future work on both the theoretical and applied sides. We discuss three select open problems whose solutions would significantly enhance the effectiveness of `DPareto` (or of any future privacy–utility Pareto front estimation methods).

The first open problem is on the privacy side. As designed, `DPareto` is a system to non-privately estimate the Pareto front of DP mechanisms. However, estimating the Pareto front requires evaluating the utility oracle many times, each time computing over potentially sensitive user data; e.g., in order to train the underlying model on the training data, and then to compute the utility of the trained model on the test data. One challenging open problem is how to tightly quantify the DP guarantee of the Pareto front estimation method itself. This involves analyzing the privacy guarantees for compositions of the utility oracle. Naively applying DP composition theorems immediately yields conservative bounds on the privacy for both the training and test sets of user data used by the utility oracle (assuming a small amount of calibrated noise is added to the utility oracle's output). This follows from observing that individual privacy–utility points evaluated by `DPareto` enjoy the DP guarantees computed by the privacy oracle, and the rest of the algorithm only involves post-processing and sequential composition. However, these bounds would be prohibitively large for practical use. We expect that a more advanced analysis could yield significantly tighter guarantees since for each point we are only releasing its utility values

rather than releasing the trained models themselves. For a decision maker, tightly quantifying DPareto's privacy guarantee would provide an end-to-end privacy guarantee for their entire workflow, and allow the Pareto front to be made publicly available.

The second open problem is on the Bayesian optimization side. Recall that the estimated Pareto fronts contain only the privacy–utility values of the trained models, along with the corresponding hyperparameters that induced them. In practice, a decision maker may be interested in finding a hyperparameter setting that induces a particular point on the estimated Pareto front but which was not previously explicitly evaluated by DPareto. Currently, there is no method to find such hyperparameters. The only recourse is to select the hyperparameters of the nearest desirable privacy–utility point that DPareto *did* explicitly evaluate. The open problem here is how to design an improved (but still computationally efficient) method to extract this information from DPareto's underlying Gaussian processes.

The final open problem is how to improve the performance of DPareto by modifying it to be specific to differential privacy (or even specific to differentially private deep learning), rather than directly leveraging general techniques in multi-objective Bayesian optimization. One straightforward starting point is to more carefully construct the Gaussian processes that model the privacy and utility oracles in two mutually compatible ways. The first modification is to redesign the privacy oracle's GP to utilize the fact that the privacy oracle, unlike the utility oracle, is noiseless. This would eliminate uncertainty at each evaluated hyperparameter setting's measured privacy value, but would retain the Bayesian optimization's required uncertainty across the unevaluated hyperparameter settings. The second modification is that rather than using independent GPs for both oracles, instead use a single multi-output GP [ARL+12] encoded with the known inverse relationship between privacy and optimal utility. We hypothesize that carefully incorporating both changes will noticeably improve DPareto's effectiveness and efficiency.

3.A Chapter Appendix

Deferred Sparse Vector Technique Privacy Proof

Here, we detail the proof for our SVT variant's (Algorithm 3.1) DP guarantee, which we use to implement the privacy oracle P_0 in the sparse vector technique illustrative example. The proof is based on observing that our implementation is just a simple re-parameterization of Lyu et al.'s mechanism [LSL17, Alg. 7], where some of the parameters have been fixed up-front. For concreteness, we reproduce their mechanism as Algorithm 3.5. The result then follows from a direct application of Theorem 4 of their work, which shows that Algorithm 3.5 is $(\epsilon_1 + \epsilon_2, 0)$-DP.

Algorithm 3.5 Sparse Vector Technique of Lyu et al. [LSL17, Alg. 7], with $\epsilon_3 = 0$

Input

- D: Sensitive dataset.
- $q_1, \ldots, q_m$: m binary queries.
- Δ: Sensitivity bound for the queries.

Hyperparameters

- $T_1, \ldots, T_m$: Thresholds for each query.
- C: Upper-bound on number of answers.
- ϵ_1, ϵ_2: Differential privacy parameters.

Body

Let $c = 0$ and $w = (\bot, \ldots, \bot) \in \{\bot, \top\}^m$
Let $\rho \sim \text{Laplace}(\Delta/\epsilon_1)$, where $b_1 = b/(1 + (2C)^{1/3})$.
for $i \in [m]$ **do**
    Let $\nu \sim \text{Laplace}(2C\Delta/\epsilon_2)$.
    if $q_i(D) + \nu \geq T_i + \rho$ **then**
        Set $w_i = \top$ and $c = c + 1$.
        if $c \geq C$, **break**
    end if
end for
Return: w.

Comparing Algorithm 3.5 with our SVT implementation in Algorithm 3.1, we see that they are virtually the same mechanisms, but where we have fixed $\Delta = 1$, $T_i = 1/2$, $\varepsilon_1 = 1/b_1$ and

$\varepsilon_2 = 2C/b_2$. Thus, by expanding the definitions of b_1 and b_2 as a function of b and C, we can verify that Algorithm 3.1 is $(\epsilon, 0)$-DP with

$$\begin{aligned}
\epsilon &= \epsilon_1 + \epsilon_2 \\
&= \frac{1}{b_1} + \frac{2C}{b_2} \\
&= \frac{1 + (2C)^{1/3}}{b} + \frac{(2C)^{2/3}(1 + 2C)^{1/3}}{b} \\
&= \frac{(1 + (2C)^{1/3})(1 + (2C)^{2/3})}{b}.
\end{aligned}$$

□

Chapter 4

Pushing the Boundaries of Private, Large-Scale Query Answering

In order to address the final high-level challenge of this thesis (Section 1.2), we focus in this chapter on the open question of efficiently and effectively answering large numbers of queries while ensuring DP[26]. We begin with an overview of the problem, describing it precisely and providing a detailed motivating example. We then separately analyze the problem in two distinct settings. In both settings, we ground our work in the state-of-the-art DP mechanism for large-scale query answering: the *Relaxed Adaptive Projection* (RAP) mechanism [Ayd+21].

The first setting is a classic setting in the DP literature where all queries are known to the mechanism in advance. Within this setting, we identify challenges in the RAP mechanism's original analysis, then overcome them with an enhanced implementation and analysis. We then extend the capabilities of the RAP mechanism to be able to answer a more general and powerful class of queries (r-of-k thresholds) than previously considered. Empirically evaluating this class, we find that the mechanism is able to answer orders of magnitude larger sets of queries than prior works and does so quickly and with high utility.

We then define a second setting motivated by real-world considerations and whose definition is inspired by work in the field of machine learning. In this new setting, a mechanism is only given partial knowledge of queries that will be posed in the future, and it is expected to answer

[26]This chapter is based on work in our publication [AK23].

these future queries with high utility. We formally define this setting and how to measure a mechanism's utility within it. We then comprehensively empirically evaluate our extended RAP mechanism's utility within this new setting. From this evaluation, we find that even with weak partial knowledge of the future queries, the mechanism is able to efficiently and effectively answer arbitrary queries posed in the future. Taken together, the results from these two settings advance the state of the art on differentially private large-scale query answering.

4.1 Overview

Many data analysis and machine learning algorithms, at their core, involve answering *statistical queries*. Statistical queries are the class of queries that answer the question: "What fraction of entries in a given dataset have a particular property P?" Because of their ubiquity, developing differentially private mechanisms to answer statistical queries effectively has been one of the most well-studied problems in DP [DN03; Blu+05; Dwo+06b; BLR08; Dwo+09; DRV10; RR10; HR10; HLM12; GRU12]. Early DP research primarily focused on designing mechanisms to answer specific, individual statistical queries in an interactive setting. In that setting, queries are posed and answered one at a time with the goal of answering each query with minimal error while ensuring privacy. However, most practical data-driven algorithms do not pose only a single query. Instead, they pose a large number of queries, referred to as a *query workload*. When a query workload is available in advance (i.e., prespecified), it is possible to design DP mechanisms that take advantage of the relationships between the queries to achieve higher utility relative to answering the individual queries independently. In this chapter, we address the problem of privately answering a large number of queries by answering the following high-level research question.

> *In the two following settings, to what extent are differentially private mechanisms able to answer a large number of statistical queries efficiently and with low error?*
>
> *Setting 1: All queries are prespecified; i.e., known in advance.*

Setting 2: Only partial knowledge of the queries is available in advance.

Motivating Example

A motivating data analysis example for this chapter is the American Community Survey (ACS), a demographics survey program conducted by the U.S. Census Bureau [Bur16]. The ACS regularly gathers information such as ancestry, citizenship, educational attainment, income, language proficiency, migration, disability, employment, and housing characteristics. The Census Bureau aggregates the individual ACS responses (microdata), then generates population estimates which are available to the public via online data tools. The most popular tool, Public Use Microdata Sample (PUMS), enables researchers to generate custom cross-tabulations of responses to the ACS questions. To protect the privacy of the ACS respondents, PUMS data are sampled, anonymized, and only available for sufficiently populous geographic regions. However, studies have found that the ad hoc anonymization techniques used are not entirely sufficient to protect the privacy of individual respondents (e.g., via re-identification attacks) [Abo18b; CRB22]. As a result, the Census Bureau has announced plans to incorporate differential privacy into the American Community Survey and declared that it is researching "a new fully-synthetic data product" with a development period ending in 2025 [Rod21; Dai22].

One promising and active direction within DP research is synthetic data generation [MSM19; Vie+20; LVW21]. The hope is that once a synthetic dataset is generated via a differentially private mechanism, researchers and analysts can pose an arbitrary number of queries against the synthetic dataset without increasing the privacy risk to those who contributed the original underlying data. DP synthetic data generation mechanisms seek to strike a balance between distilling the information in the underlying dataset most useful to analysts while ensuring the underlying dataset's privacy. Thus, to maximize the eventual usefulness of the synthetic dataset, synthetic data generation mechanisms must tailor the generated dataset to the specific class of downstream

tasks (e.g., a particular class of queries) that analysts are most likely interested in. This is typically done by providing a set of queries (the query workload) to the DP mechanism so that the mechanism can tailor the synthetic dataset to answer these queries (and ideally, to other similar queries). Much of DP synthetic data research has focused on designing mechanisms to generate synthetic data which can provide accurate answers (under a variety of metrics, most commonly ℓ_∞ error) to the subset of statistical queries known as k-way marginal queries [Bar+07; TUV12; Gup+13; Cha+14; CKS18; MSM19; Vie+20; Nix+22]. Informally, a k-way marginal query answers the question: "What fraction of people in the private dataset have all of the following k attributes: ...?" In this chapter, we focus on a strict generalization of k-way marginal queries known as r-of-k threshold queries [Kea+87; Lit88; HW04; TUV12; Ull13; Ayd+21] under the ℓ_∞ error metric. Informally, r-of-k threshold queries answer the question: "What fraction of people in the private dataset have at least r of the following k attributes: ...?".

As a simplified example of where such queries can be used, we consider the scenario where a social scientist is interested in using ACS data to determine what portion of a community has a substandard quality of living. Suppose the scientist wants to examine the four following attributes for each person in the community: is their income level below the poverty line, are they unemployed, are they homeless, and do they have a low net worth? Clearly, a person having any single attribute does not necessarily mean they have a substandard quality of living. Similarly, a person does not need to have all four attributes to have a substandard quality of living. Thus, the social scientist can formulate this as an r-of-k threshold query with $r = 3$, $k = 4$; i.e., a person has a substandard quality of living if they have at least three of the four attributes.

This social scientist may have many such queries, and other researchers may have sets of queries of their own that they wish to pose. Thus, a natural algorithm design question is: how should the U.S. Census Bureau answer everyone's queries with low error while still ensuring the ACS respondents' privacy? The simplest option is to use a portion of the DP budget to individually answer each query, independent of all other queries. This would likely be unsatisfactory

utility-wise since it both limits how many queries can be answered and ignores any relationships between queries (which would likely lead to a large ℓ_∞ error over the set of answers). We posit two potentially superior alternatives whose performance we will investigate.

1. One alternative is to collect a large group of queries and then use a state-of-the-art DP query answering mechanism to answer them all simultaneously. This is an example of answering queries in the "prespecified queries" setting (studied in Sections 4.3 and 4.4). With careful DP mechanism design or selection, this alternative typically leads to lower ℓ_∞ error over the set of answers than answering each query independently.

2. A separate alternative is along the lines of synthetic data generation and is applicable to the Census Bureau if queries that have been posed in the past are in some sense similar to queries that analysts will likely pose in the future. Concretely, we hypothesize that the Census Bureau can leverage those past queries in conjunction with a state-of-the-art DP synthetic data generation mechanism to generate a synthetic dataset privately. Researchers can then pose their own queries directly against the synthetic dataset without needing to go through the Census Bureau or worry about the original ACS respondents' privacy. This is an example of answering queries in the "partial knowledge" setting (studied in Section 4.5), as knowledge from the past is used to inform the future. If the queries posed in the past are indeed similar to the queries posed in the future, then a synthetic dataset generated using the past queries has the potential to answer the future queries with low ℓ_∞ error.

4.1.1 Prior Work on Large-Scale Query Answering

To address answering a large number of queries under differential privacy in an improved manner over the naive interactive approach, two separate lines of research previously emerged: synthetic data generation and workload evaluation. We describe both lines of research, then briefly introduce the state-of-the-art mechanism we build upon in the thesis.

Synthetic Data Generation: One line of research studies the problem of answering a large number of queries via private synthetic dataset generation. In differentially private synthetic dataset generation, a DP mechanism is applied to the original, sensitive data in order to generate a synthetic dataset. The synthetic dataset's purpose is then to directly answer arbitrary queries posed in the future without the further need to account for potential privacy leakage or manage differential privacy budgets. In this setting, aside from knowing the general query class, *no knowledge is typically assumed about which specific queries will be posed in the future.* The proven advantage of this approach is that DP synthetic datasets are theoretically capable of accurately answering an exponentially larger number of queries relative to the aforementioned interactive approach [BLR08; GRU12; Che+12; HRS12; Gup+13]. However, actually generating a synthetic dataset that accurately answers exponentially many queries has been proven intractable [Dwo+09; UV11; Ull16], even for simple subclasses of statistical queries (e.g., 2-way marginals). Thus, a significant recent research focus has been on designing efficient mechanisms for privately generating synthetic datasets which accurately answer increasingly large numbers of queries [Gab+14; MSM19; Vie+20; LVW21].

Workload Evaluation: A separate line of research focuses on the problem of answering a large number of queries when the concrete query workload is prespecified, i.e., *when all queries are known in advance.* Pre-specifying the query workload enables researchers to design DP mechanisms to take advantage of the workload's structure in order to answer the queries with lower error relative to the interactive approach or the private synthetic dataset approach. Early research in this setting produced mechanisms with optimal or near-optimal error guarantees but with impractical (typically exponential) running times for even modestly sized real-world problems [HR10; HLM12; GRU12; Li+15]. As a result, recent research has focused on designing computationally efficient mechanisms to answer prespecified workloads with low error on real-world datasets [McK+18; SS18; Ayd+21], at the cost of losing the strong theoretical utility guarantees

of prior works and thus necessitating thorough empirical utility evaluations to demonstrate their value.

***Relaxed Adaptive Projection* Mechanism:** Our approach for evaluating suitable (i.e., efficient and accurate) mechanisms in both our settings of interest builds on Aydore et al.'s [Ayd+21] recently introduced *Relaxed Adaptive Projection* (RAP) mechanism. RAP is the current state-of-the-art mechanism for answering large sets of statistical queries in the setting where the query workload is prespecified. At a high level, RAP works by:

1. Initializing a synthetic dataset D' in a relaxed data space (e.g., by relaxing a binary feature in the original dataset to the interval $[0, 1]$ in the synthetic dataset).

2. For each original prespecified query, specifying a surrogate query that is equivalent to the original in the unrelaxed data space but that is differentiable everywhere in the relaxed space.

3. Iteratively applying an *Adaptive Selection* (AS) step followed by a *Relaxed Projection* (RP) step. In the AS step, adaptivity is introduced to allow the subset of queries with the highest error on D' to be privately selected. In the RP step, these selected queries' surrogates are used to optimize D' using standard gradient-based optimization techniques.

4. Finally, answering the original set of queries using the optimized synthetic dataset D'.

For k-way marginals, a canonical subclass of statistical queries [Bar+07; TUV12; Gup+13; Cha+14; CKS18] (formally defined in Section 4.2), Aydore et al. theoretically and empirically demonstrate that RAP outperforms prior state-of-the-art mechanisms. Theoretically, they provide an "oracle efficient" (i.e., assuming the optimization procedure achieves a global minimum) utility result characterizing RAP's error, showing that RAP achieves strictly lower error than the previous practical state-of-the-art mechanism [Vie+20]. Experimentally, they compare the RAP mechanism

with prior state-of-the-art mechanisms [MSM19; Vie+20], demonstrating that RAP answers pre-specified sets of queries with lower error.

4.1.2 Our Contributions

To answer this chapter's high-level research question, we make the following contributions in both settings of interest. In the classic setting where all queries are known in advance, our contributions are as follows.

- We overcome memory hurdles in RAP's initial implementation by reimplementing RAP in a memory-efficient way, thus enabling the evaluation of significantly larger query spaces than previously considered.
- We utilize the new implementation to enhance RAP's evaluation, evaluating RAP on larger query spaces (answering approximately 50x more queries) than in its initial evaluation and conclusively determining the role that adaptivity from the AS step plays in RAP's utility.
- We extend RAP's applicability by expanding the class of queries it evaluates, finding that it can efficiently and effectively answer more complex query classes than previously considered.

As a realistic intermediate setting between the two classic extremes of no-knowledge vs. full-knowledge of which queries will be posed, we propose a new setting where partial knowledge of the future queries is available. In this new setting, our contributions are as follows.

- We concretely define this setting and how to measure utility within it. Specifically, we assume that a set of historical queries was independently drawn from some unknown distribution $\mathcal{T}_H$ and that the mechanism has access to these historical queries. In the future, the mechanism will be posed an arbitrary number of queries sampled from a distribution

$\mathcal{T}_F$, which may be related to $\mathcal{T}_H$. We define the utility of the mechanism in terms of its generalization error, i.e., its expected error across these future queries drawn from $\mathcal{T}_F$ having been given access to the historical queries from $\mathcal{T}_H$.

- We assess how suitable RAP is for this new setting by formulating query distributions according to real-world phenomena, then empirically evaluating RAP's generalization error on these distributions. When future queries are drawn from the same distribution as the historical queries that RAP used to learn its synthetic dataset (i.e., $\mathcal{T}_H = \mathcal{T}_F$), we find that regardless of the distribution, RAP is able to achieve high utility. When the future queries distribution diverges from the historical queries distribution, we find that RAP's utility slowly and gracefully declines.

These contributions, in both the prespecified queries setting and the partial knowledge setting, demonstrate the practical value of RAP and improve RAP's adoptability for real-world uses.

The remainder of this chapter is structured as follows. Beginning in Section 4.2, we provide a comprehensive overview of the relevant technical terminology and definitions and detail the RAP mechanism that we build upon. In Section 4.3, we perform a focused but thorough reproducibility study on Aydore et al.'s [Ayd+21] evaluation of the RAP mechanism. To accomplish this, we first improve RAP's implementation from the ground up and then leverage the new implementation to enhance RAP's initial evaluation in order to strengthen our comprehension of its utility. Building on the improved RAP implementation, in Section 4.4 we expand the class of queries that RAP is able to accommodate. We then empirically evaluate RAP on this new class of queries, finding that it is able to efficiently answer large numbers of queries from this class while maintaining high utility. In Section 4.5, we concretely define our newly proposed setting where a mechanism is given partial knowledge of the queries that will be posed in the future. We define how we assess a mechanism's performance in this setting and detail the distinct new ways that RAP's performance may be affected in this new setting. We then empirically evaluate RAP in this setting,

finding that even with only partial knowledge of which queries will be posed in the future, RAP is able to efficiently and effectively achieve high utility. Finally, in Section 4.6, in addition to the related works already discussed in this section, we describe other important, relevant works and the future directions they motivate pertaining to our work in this chapter.

4.2 Technical Preliminaries

In this section, we define the requisite technical terminology. The fundamental concepts introduced here were primarily presented in prior works [Gab+14; Vie+20; Ayd+21]. We restate them to aid in understanding and contextualizing Aydore et al.'s RAP mechanism, which we use to answer this chapter's research questions. Towards this, we first define statistical queries and their subclasses that are relevant to this work. We then define what it means to be a "surrogate" query for one of these statistical queries. Next, we describe what workloads are and how we use them. Finally, we detail the RAP mechanism that we build on in this work. Because this chapter is notationally dense, Table 4.1 serves as a reference for the various symbols we define.

4.2.1 Statistical Queries and their Subclasses

The general class of queries we are interested in (which the RAP mechanism can, in theory, be used to answer) are statistical queries.

Definition 4.2.1 (Statistical query)**.** A *statistical query* q_ϕ is parameterized by a predicate $\phi : \mathcal{X} \to \{0, 1\}$; i.e., the predicate takes as input a record x of a dataset D and outputs a boolean value. The statistical query is then defined as the normalized count of the predicate over all n records of the input dataset, i.e.,

$$q_\phi(D) = \frac{\sum_{x \in D} \phi(x)}{n}.$$

Given a vector of m statistical queries Q, we define $Q(D) = (a_1, \ldots, a_m)$ to be the answers to each of the queries on D; i.e., $a_i = q_{\phi_i}(D)$ for all $i \in [m]$.

	Symbol	Usage
	ϵ, δ	Differential privacy parameters.
	$\mathcal{X},\ d,\ \mathcal{X}_i$	Data space $\mathcal{X}$ for any possible record consisting of d features. $\mathcal{X}_i$ is the domain of feature i.
	$D,\ n$	Dataset D containing n records from $\mathcal{X}$.
	q_ϕ	Statistical query q defined by the mean of the predicate ϕ over a set of records from $\mathcal{X}$.
	$Q,\ m,\ a$	Q is a vector of m queries, and a represents the answers to the vector of queries over the dataset D such that $Q(D) = a = (a_1, \ldots, a_m)$.
	W	Threshold workload W which defines the concrete query vector Q.
	$q_{\phi_{S,y,k}}$	k-way marginal query specified by set S of k features and values y for each feature.
	$q_{\phi_{S,y,1}}$	1-of-k threshold query specified by set S of k features and values y for each feature.
⋆	$q_{\phi_{S,y,r}}$	r-of-k threshold query specified by set S of k features and values y for each feature, and threshold r.
	$\mathcal{Y},\ d'$	Data space $\mathcal{Y}$ consisting of d' features, which is a relaxation of the one-hot encoded $\mathcal{X}$ data space.
	$D',\ n'$	Synthetic dataset D' containing n' features from $\mathcal{Y}$.
	$\hat{q}_{\hat{\phi}}$	Surrogate query $\hat{q}$ defined by the mean of the function $\hat{\phi}$ over a set of records from $\mathcal{Y}$.
	$\hat{Q}$	Vector of surrogate queries.
	$\hat{q}_{\hat{\phi}_T}$	Product query, specified by a set of features T.
⋆	$\hat{q}_{\hat{\phi}_{T_+,T_-}}$	Generalized product query, specified by a set of positive and negated features T_+ and T_-.
⋆	$\hat{q}_{\hat{\phi}_{T,r}}$	Polynomial threshold query, specified by a set of features T and integer r.
⋆	err_P	Measure of a mechanism's present error, used when all queries are known in advance.
⋆	err_F	Measure of mechanism's future error, used when only partial knowledge of queries is available in advance.
⋆	$\mathcal{F}, \mathcal{T}$	Distribution $\mathcal{T}$ from which thresholds in a random workload are sampled i.i.d. to form a corresponding vector of consistent queries. The threshold distribution may be formed by a distribution over features $\mathcal{F}$.
	`RAP`, `AS`, `RP`	Relaxed Adaptive Projection mechanism, with its primary subcomponents: the Adaptive Selection and Relaxed Projection mechanisms.
	`RNM`	Report Noisy Max mechanism, used by the `AS` mechanism to select high-error queries.
	`GM`	Gaussian noise-addition mechanism, used as both a baseline mechanism and a subcomponent of `RAP` to privately answer queries directly.
⋆	`OSAS`	Oneshot Adaptive Selection mechanism, introduced as more efficient a drop-in replacement for `RAP`'s `AS` mechanism.
	`All-0`	Baseline mechanism that returns only 0 for all queries.

Table 4.1: Comprehensive list of notation. Lines marked with a ⋆ indicate new concepts not found in [Ayd+21].

We now formally define the specific subclasses of statistical queries that we reference throughout this chapter. Let the space for each record in the dataset consist of d categorical features $\mathcal{X} = (\mathcal{X}_1 \times \cdots \times \mathcal{X}_d)$, where each $\mathcal{X}_i$ is the discrete domain of feature i, and let $x_i \in \mathcal{X}_i$ denote the value of feature i of record $x \in \mathcal{X}$. Prior works have primarily evaluated the subclass of statistical queries known as k-way marginals (also known as k-way contingency tables or k-way conjunctions) [Bar+07; TUV12; Gup+13; Cha+14; CKS18; MSM19; Vie+20], and typically focused specifically on 3-way and 5-way marginals.

Definition 4.2.2 (k-way marginal)**.** A *k-way marginal query* $q_{\phi_{S,y,k}}$ is a statistical query whose predicate $\phi_{S,y,k}$ is specified by a set S of k features $f_1 \neq \cdots \neq f_k \in [d]$ and a target $y \in (\mathcal{X}_{f_1} \times \cdots \times \mathcal{X}_{f_k})$, given by

$$\phi_{S,y,k}(x) = \begin{cases} 1 & \text{if } x_{f_1} = y_1 \wedge \cdots \wedge x_{f_k} = y_k \\ 0 & \text{otherwise.} \end{cases}$$

Informally, a row satisfies the predicate if *all* of its values match the target on the specified features. A *k-way marginal* is then specified by a set S of k features, and consists of all $(\Pi_{i=1}^{k} |\mathcal{X}_{f_i}|)$ k-way marginal queries with feature set S.

1-of-k thresholds (also known as k-way disjunctions) were briefly evaluated in [Ayd+21], and are defined similarly.

Definition 4.2.3 (1-of-k threshold)**.** A *1-of-k threshold query* $q_{\phi_{S,y,1}}$ is a statistical query whose predicate $\phi_{S,y,1}$ is specified by a set S of k features $f_1 \neq \cdots \neq f_k \in [d]$ and a target $y \in (\mathcal{X}_{f_1} \times \cdots \times \mathcal{X}_{f_k})$, given by

$$\phi_{S,y,1}(x) = \begin{cases} 1 & \text{if } x_{f_1} = y_1 \vee \cdots \vee x_{f_k} = y_k \\ 0 & \text{otherwise.} \end{cases}$$

Informally, a row satisfies the predicate if *any* of its values match the target on the specified features. A *1-of-k threshold* is then specified by a set S of k features, and consists of all $(\Pi_{i=1}^{k}|\mathcal{X}_{f_i}|)$ 1-of-k threshold queries with feature set S.

In this work, we evaluate a generalization of both of these subclasses of statistical queries: r-of-k thresholds [Kea+87; Lit88; HW04; TUV12; Ull13; Ayd+21].

Definition 4.2.4 (r-of-k threshold)**.** An *r-of-k threshold query* $q_{\phi_{S,y,r}}$ is a statistical query whose predicate $\phi_{S,y,r}$ is specified by a positive integer $r \leq k$, a set S of k features $f_1 \neq \cdots \neq f_k \in [d]$[27], and a target $y \in (\mathcal{X}_{f_1} \times \cdots \times \mathcal{X}_{f_k})$. The predicate is then given by

$$\phi_{S,y,r}(x) = \mathbb{1}\left[\sum_{i=1}^{k} \mathbb{1}[x_{f_i} = y_i] \geq r\right].$$

Informally, a row satisfies the predicate if *at least* r of its values match the target on the specified features. An *r-of-k threshold* is then specified by positive integer $r \leq k$ and a set S of k features, and consists of all $(\Pi_{i=1}^{k}|\mathcal{X}_{f_i}|)$ r-of-k threshold queries with feature set S. This class generalizes k-way marginals when $r = k$ and generalizes 1-of-k thresholds when $r = 1$.

The expressiveness of r-of-k thresholds makes them more useful than k-way marginals, as they enable more nuanced queries to be easily and intuitively posed. This is particularly useful when the implications behind categories of distinct features in a dataset have some overlap. For instance, in the motivating U.S. Census example, there were several features with categories indicative of a substandard quality of living. Requiring someone to belong to *all* of the categories (as a k-way marginal requires) is overly restrictive, and r-of-k thresholds allow this restrictiveness to be relaxed.

Remark. We say that any r-of-k threshold query (and, by extension, any k-way marginal query or 1-of-k threshold query) specified by r, k, S, and y is *consistent* with the r-of-k threshold specified by r, k, and S. That is, we often refer to an r-of-k threshold simply as the features it specifies,

[27] Although requiring each f_i to be distinct is not strictly necessary for the techniques in this chapter to hold, we subsequently discuss in Section 4.2.2 why this is a reasonable requirement in this setting.

whereas a query *consistent with* that r-of-k threshold is one which specifies concrete target values corresponding to those features.

4.2.2 Threshold Workloads

It was standard in prior works to evaluate *workloads* of k-way marginals [Li+15; McK+18; MSM19; Vie+20; Liu+21; LVW21]. A k-way marginal workload W is specified by a set of k-way marginals, $W = \{S_1, \ldots, S_{|W|}\}$ such that each $S_i \in W$ is a set of k features. This workload W defines a concrete query vector Q, which consists of all queries consistent with each marginal in W. Q is therefore commonly referred to as the *query workload*. For example, a workload may be specified by the following two 3-way marginals, $W = \{(1, 2, 5), (2, 3, 7)\}$, and would therefore define the query vector Q containing all marginal queries consistent with those feature sets. The number of queries in this query vector would then be $|Q| = |\mathcal{X}_1||\mathcal{X}_2||\mathcal{X}_5| + |\mathcal{X}_2||\mathcal{X}_3||\mathcal{X}_7|$.

Since our work extends the class of queries from marginals to r-of-k thresholds, rather than a workload being specified by a set of marginals, we say that a set of r-of-k thresholds specifies a workload W. W similarly defines the concrete query vector Q which consists of all r-of-k threshold queries consistent with each r-of-k threshold in W. For example, when $r = 1$ and $k = 3$, we can specify a similar workload as before $W = \{(1, 2, 5), (2, 3, 7)\}$ which defines a query workload Q containing the same number of consistent queries as before ($|Q| = |\mathcal{X}_1||\mathcal{X}_2||\mathcal{X}_5| + |\mathcal{X}_2||\mathcal{X}_3||\mathcal{X}_7|$) — however, here each $q \in Q$ is a 1-of-3 threshold query instead of a 3-way marginal query. We use threshold workloads (and their corresponding vector of all consistent queries) for the empirical evaluations of our mechanisms.

In Definition 4.2.4 for r-of-k thresholds, each f_i was required to be distinct. For individual queries, removing this requirement does not significantly change the mechanisms or techniques in this chapter. However, since we answer *all* consistent queries of a given threshold in this chapter, this requirement does not diminish the richness of the query class. Specifically, if one seeks to pose an r-of-k threshold query containing non-distinct features, its answer can be computed

directly from the answers to all consistent queries of the same threshold but with distinct features. This holds true in the opposite direction as well. Thus, both definitions are equivalent in this sense when answering all consistent queries over a given set of features.

4.2.3 Surrogate Queries

Aydore et al. [Ayd+21] introduce surrogate queries to replace the original statistical queries with queries that are similar but that are amenable to first-order optimization methods. Thanks to significant recent advances in hardware and software tooling, these first-order optimization methods can enable highly efficient learning of synthetic datasets.

Definition 4.2.5 (Surrogate Query)**.** A *surrogate query* $\hat{q}_{\hat{\phi}}$ is parameterized by function $\hat{\phi} : \mathcal{Y} \to \mathbb{R}$; i.e., the function takes as input a record $x \in \mathcal{Y}$ from a dataset D', and outputs a real value. The surrogate query is then defined as the mean of the function over all n' records of the input dataset, i.e.,

$$\hat{q}_{\hat{\phi}}(D') = \frac{\sum_{x \in D'} \hat{\phi}(x)}{n'}.$$

The only distinctions between the definitions of a surrogate query with $\hat{\phi}$ and a statistical query with ϕ are that $\hat{\phi}$'s domain may be different than ϕ's, and $\hat{\phi}$'s codomain is the entire real line instead of $\{0, 1\}$.

We let $\hat{Q}$ denote a corresponding vector of surrogate queries for a given query vector Q.

Notation of Feature Spaces: Recall the original feature space $\mathcal{X} = (\mathcal{X}_1 \times \cdots \times \mathcal{X}_d)$, where each $\mathcal{X}_i$ is the discrete domain of feature i, and let t_i be the number of distinct values/categories that $\mathcal{X}_i$ can attain. A one-hot encoding[28] $h(x)$ of any record x results in a binary vector $\{0, 1\}^{d'}$, where $d' = \sum_{i=1}^{d} t_i$. Just as in [Ayd+21], we are interested in constructing a synthetic dataset that lies in a continuous relaxation of this binary feature space. A natural relaxation of $\{0, 1\}^{d'}$

[28] A one-hot encoding of a categorical feature $\mathcal{X}_i$ with t_i categories is a mapping from each category to a unique $1 \times t_i$ binary vector that has exactly one non-zero coordinate.

is $[0,1]^{d'}$, so we adopt $\mathcal{Y} = [0,1]^{d'}$ as the relaxed space for the remainder of this chapter.

We are interested in surrogate queries that are *equivalent extended differentiable queries* (EEDQs) as defined in [Ayd+21].

Definition 4.2.6 (Equivalent Extended Differentiable Query)**.** Let q_ϕ be an arbitrary statistical query parameterized by $\phi : \mathcal{X} \to \{0,1\}$, fix a one-hot encoding over each of the features, and let $\hat{q}_{\hat{\phi}}$ be a surrogate query parameterized by $\hat{\phi} : \mathcal{Y} \to \mathbb{R}$. We say that $\hat{q}_{\hat{\phi}}$ is an *equivalent extended differentiable query* to q_ϕ if it satisfies the following properties:

1. $\hat{\phi}$ is differentiable over $\mathcal{Y}$. I.e., for every $x \in \mathcal{Y}$, $\nabla\hat{\phi}(x)$ is defined.
2. $\hat{\phi}$ agrees with ϕ on every possible database record that results from the fixed one-hot encoding. I.e., for every $x \in \mathcal{X}$ where $h(x)$ represents the one-hot encoding of x: $\phi(x) = \hat{\phi}(h(x))$.

As an illustrative example of an EEDQ, we define the class of EEDQs used by Aydore et al. for k-way marginals. Concretely, [Ayd+21] defines the class of surrogate queries known as *product queries* and shows how to construct an EEDQ product query for any given k-way marginal.

Definition 4.2.7 (Product Query)**.** Given a subset of features $T \subseteq [d']$, the *product query* $\hat{q}_{\hat{\phi}_T}$ is a surrogate query parameterized by the function $\hat{\phi}_T$ which is defined as $\hat{\phi}_T(x) = \prod_{i \in T} x_i$.

Lemma 4.2.8 ([Ayd+21], Lemma 3.3)**.** Every k-way marginal query $q_{\phi_{S,y,k}}$ has an EEDQ in the class of product queries. By construction, every $\hat{\phi}_T$ satisfies the requirement that it is defined over the entire relaxed space $\mathcal{Y}$ and is differentiable. Additionally, for every $q_{\phi_{S,y,k}}$, there is a corresponding product query $\hat{q}_{\hat{\phi}_T}$ with $|T| = k$ such that for every $x \in \mathcal{X} : \phi_{S,y,k}(x) = \hat{\phi}_T(h(x))$. We construct this T in the following straightforward way: for every $i \in S$, we include in T the coordinate corresponding to $y_i \in \mathcal{X}_{f_i}$.

4.2.4 *Relaxed Adaptive Projection* (RAP) Mechanism

We now describe the details of the RAP mechanism, including how it works and its DP guarantee.

Algorithm 4.1 formally defines the RAP mechanism. The input to the mechanism is the dataset D of sensitive user data, the desired size of the synthetic dataset n', privacy parameters (ϵ, δ), a vector of m statistical queries Q and their corresponding surrogate queries $\hat{Q}$, adaptiveness parameters $T, K \in [m]$. The final outputs are (1) an n'-row synthetic dataset, and (2) estimates to the original queries Q obtained by evaluating their surrogate queries on the synthetic dataset; i.e., RAP outputs (1) D' and (2) $\hat{Q}(D')$.

Algorithm 4.1 Relaxed Adaptive Projection (RAP) Mechanism

Input

- D: Dataset of n records from space $\mathcal{X}$.
- $Q, \hat{Q}$: A vector of m statistical queries and their corresponding surrogate queries.
- $n', \mathcal{Y}$: Desired size of synthetic dataset with records from relaxed space $\mathcal{Y}$.
- T: Number of rounds of adaptiveness.
- K: Number of queries to select per round of adaptiveness.
- ϵ, δ: Differential privacy parameters.

Body

Let $\rho = \epsilon + 2\left(\log(\frac{1}{\delta}) - \sqrt{\log(\frac{1}{\delta})(\epsilon + \log(\frac{1}{\delta}))}\right)$.
Independently uniformly randomly initialize $D' \in \mathcal{Y}^{n'}$.
if $T = 1,\ K = m$ **then**
  for $i = 1, 2, \ldots, m$ **do**
    Let $\tilde{a}_i = \texttt{GM}(D, q_i, \rho/m)$.
  end for
  Let $D' = \texttt{RP}(\hat{Q}, \tilde{a}, D')$.
else
  Let $Q_s = \emptyset$.
  for $t = 1, 2, \ldots, T$ **do**
    Let $Q_s, \tilde{a} = \texttt{AS}(D, D', Q, \hat{Q}, Q_s, K, \frac{\rho}{T})$.
    Let $\hat{Q}_s = (\hat{q}_i : q_i \in Q_s)$.
    Let $D' = \texttt{RP}(D', \hat{Q}_s, \tilde{a})$.
  end for
end if
Return: Final synthetic dataset D' and answers $\hat{Q}(D')$.

Non-Adaptive Case: In its most basic form ($T = 1, K = m$), RAP employs no adaptivity. Here, the vector of m queries is first privately answered directly on the sensitive dataset D using

Algorithm 4.2 Adaptive Selection (AS) Mechanism

Input

- D, D': Dataset of n records from space $\mathcal{X}$, and synthetic dataset of n' records from relaxed space $\mathcal{Y}$.
- $Q, \hat{Q}$: Vector of all statistical queries and their corresponding surrogate queries.
- Q_s: Set of already selected queries.
- K: Number of new queries to select.
- ρ: Differential privacy parameter.

Body

1: **for** $j = 1, 2, \ldots, K$ **do**
2: Let $\Delta = (|\hat{q}_i(D) - \hat{q}_i(D')| : q_i \in Q \setminus Q_s)$.
3: Let $i = \texttt{RNM}(\Delta, \frac{\rho}{2K})$
4: Add q_i into Q_s.
5: Let $\tilde{a}_i = \texttt{GM}(D, q_i, \frac{\rho}{2K})$.
6: **end for**
7: **Return:** Q_s and $\tilde{a} = (\tilde{a}_i : q_i \in Q_s)$.

Algorithm 4.3 Relaxed Projection (RP) Mechanism

Input

- D': Synthetic dataset of n' records from relaxed space $\mathcal{Y}$.
- $\hat{Q}$: Vector of surrogate queries.
- $\tilde{a}$: Vector of "true" privatized answers corresponding to each surrogate query.

Body

1: Use any iterative differentiable optimization technique (SGD, Adam, etc.) to attempt to find:

$$D' = \underset{D' \in \mathcal{Y}^{n'}}{\arg\min} ||\hat{Q}(D') - \tilde{a}||_2^2,$$

applying the Sparsemax transformation to every feature encoding in each row of D' between each iteration.
2: **Return:** D'.

the *Gaussian Mechanism* (GM) (Section 1.1.2). These answers, along with the vector of surrogate queries $\hat{Q}$ and a uniformly randomly initialized n'-row synthetic dataset D', are passed to the *Relaxed Projection* mechanism (RP, Algorithm 4.3). The RP subcomponent utilizes an iterative gradient-based optimization procedure (such as SGD) to update D' by minimizing the disparity between the surrogate queries' answers on D' and the privatized answers on the sensitive dataset D. After each iterative update, the Sparsemax transformation is applied to every feature encoding in each row of D'. Once the procedure reaches a stopping condition (e.g., $\hat{Q}(D')$ is within a certain tolerance of $\tilde{a}$, or a certain number of iterations have occurred), RP returns the final D'. RAP then returns D' along with estimated answers to the query workload $\hat{Q}(D')$.

Adaptive Case: In the more general case where $K < m$ and $T \cdot K \leq m$, RAP proceeds in T rounds. In each round t, RAP uses the *Adaptive Selection* (AS) mechanism to select K new queries to add to the set Q_s. AS iteratively uses the Gumbel noise *Report Noisy Max* (RNM) [Che+16; DR19] and GM mechanisms together to privately choose the K queries that have the largest disparity between their current answers on the synthetic dataset D' and their answers on the true dataset D. The RP mechanism is then applied only to this subset Q_s containing $t \cdot K$ queries in each round, rather than applying RP in 1 round on the full vector of privately answered queries Q (as in the non-adaptive case). Aydore et al. claim that the aim of incorporating this adaptivity is to expend the privacy budget more wisely by selectively answering only the $T \cdot K \ll m$ total worst-performing queries.

Concentrated Differential Privacy

To state and understand RAP's DP guarantee, we must briefly discuss *zero-concentrated differential privacy* (zCDP) [BS16].

Although RAP is given ϵ and δ values as input and, in turn, guarantees (ϵ, δ)-DP, its DP sub-mechanisms and corresponding privacy proof are in terms of ρ-zCDP. Zero-concentrated differential privacy is a different definition of DP that provides a weaker guarantee than pure DP but a stronger guarantee than approximate DP. It is formally defined as follows.

Definition 4.2.9 ([BS16])**.** A randomized mechanism $\mathcal{M}$ is ρ-zCDP if and only if for all neighboring input datasets D and D' that differ in precisely one individual's data and for all $\alpha \in (1, \infty)$, the following inequality is satisfied:

$$\mathbb{D}_\alpha(\mathcal{M}(D)||\mathcal{M}(D')) \leq \rho\alpha,$$

where $\mathbb{D}_\alpha(\cdot||\cdot)$ is the α-Rényi divergence.

We omit a detailed discussion of zCDP in this thesis, referring an interested reader to Bun and Steinke's work [BS16] for more details. However, its value for RAP comes from the fact that zCDP has better composition properties than approximate DP, yet RAP's final composed zCDP guarantee (parameterized by ρ) can be converted back into an (ϵ, δ)-DP guarantee. This converted (ϵ, δ)-DP guarantee is better than if standard composition results of approximate DP had been directly applied.

We now informally state these composition and conversion properties. zCDP's composition property ensures that if two mechanisms satisfy ρ_1-zCDP and ρ_2-zCDP, then a mechanism that sequentially composes them satisfies ρ-zCDP with $\rho = \rho_1 + \rho_2$. zCDP's conversion property ensures that if a mechanism satisfies ρ-zCDP, then for any $\delta > 0$, the mechanism also satisfies (ϵ, δ)-DP with $\epsilon = \rho + 2\sqrt{\rho \log(1/\delta)}$.

Finally, we define the two fundamental DP mechanisms used in RAP — GM and RNM — and state their DP guarantees in terms of zCDP. The first mechanism is the Gaussian mechanism, previously discussed in Section 1.1.2. For convenience, we redefine it here in terms of zCDP and for the particular use case of answering a single statistical query.

Definition 4.2.10. The Gaussian mechanism $\texttt{GM}(D, q_i, \rho)$ takes as input a dataset $D \in \mathcal{X}^n$, a statistical query q_i, and a zCDP parameter ρ. It outputs $a_i = q_i(D) + Z$, where $Z \sim \text{Normal}(0, \sigma^2)$ and $\sigma^2 = \frac{1}{2n^2\rho}$.

Lemma 4.2.11 ([BS16])**.** For any query q_i and $\rho > 0$, the $\texttt{GM}(D, q_i, \rho)$ satisfies ρ-zCDP.

The second fundamental mechanism that RAP uses is the Gumbel noise Report Noisy Max (RNM) mechanism.

Definition 4.2.12. The Report Noisy Max mechanism $\texttt{RNM}(D, \Delta, \rho)$ takes as input a dataset $D \in \mathcal{X}^n$, a vector of real values Δ, and a zCDP parameter ρ. It outputs the index of the highest noisy value in Δ; i.e., $i^* = \arg\max_i \Delta_i + Z_i$, where each $Z_i \sim \text{Gumbel}\left(\frac{1}{\sqrt{2\rho|D|^2}}\right)$.

Lemma 4.2.13 ([DR19])**.** For any real vector Δ and $\rho > 0$, the $\texttt{RNM}(D, \Delta, \rho)$ satisfies ρ-zCDP.

With these fundamental mechanisms and their zCDP guarantees defined, we are now able to formally reproduce Aydore et al.'s original theorem and proof of RAP's DP guarantee.

Theorem 4.2.14 ([Ayd+21])**.** For any class of queries and surrogate queries Q and $\hat{Q}$, and for any set of parameters n', T, and K, the RAP mechanism satisfies (ϵ, δ)-DP.

Proof. First, consider the non-adaptive case where $T = 1, K = m$. Here, the sensitive dataset D is only accessed via m invocations of the Gaussian mechanism, each with privacy ρ/m. Therefore, by the composition property of zCDP, RAP satisfies ρ-zCDP. Thus, by our choice of ρ in line 1, we conclude that RAP satisfies (ϵ, δ)-DP.

Next, assume $T > 1$. RAP executes T iterations of its loop, only accessing the sensitive dataset D via the Adaptive Selection (AS) mechanism each iteration. Thus, we seek to prove that the AS mechanism satisfies ρ/T-zCDP. Each invocation of the AS mechanism receives the privacy parameter $\rho' = \rho/T$ as input and accesses the sensitive dataset via K invocations of RNM and K invocations of GM. Each invocation of either mechanism ensures $\frac{\rho'}{2K}$-zCDP, and therefore by

the composition property of zCDP, the total $2K$ mechanism invocations ensure ρ'-zCDP. Thus, the AS mechanism satisfies ρ/T-zCDP. Leveraging zCDP's composition property again, because RAP invokes AS T times, RAP therefore satisfies ρ-zCDP. Finally, by our choice of ρ in line 1, we conclude that RAP satisfies (ϵ, δ)-DP. □

Remark (Restriction to r-of-k thresholds)**.** r-of-k threshold queries are a subclass of a more general class of statistical queries: arbitrary Boolean formulas over k atoms of the form $x_{f_i} = y_i$. As RAP can conceptually be used to answer arbitrary statistical queries, it follows that RAP can be used to answer these k-restricted arbitrary Boolean formulas as well. In fact, we hypothesize that RAP would likely be able to answer queries from this class with high utility. However, in this work, we restrict ourselves from generalizing beyond r-of-k thresholds to this more general class for two reasons. The first reason is regarding our aforementioned evaluation of workloads. In this more general query class, it is not entirely clear what the value is of evaluating all consistent queries corresponding to the k features of an arbitrary Boolean formula, or whether that would even be the most meaningful evaluation methodology for the query class. The second reason is regarding performance. For a mechanism to be capable of evaluating this query class at a large scale, particularly careful attention would need to be paid to designing a system that automatically generates *efficient* EEDQs for any given Boolean formula. Thus, we leave generalizing beyond r-of-k thresholds as an interesting direction for future work.

4.3 Enhancing RAP's Evaluation

In this section, we address our first two contributions in the setting where all queries are pre-specified: we strengthen and clarify our understanding of RAP's utility by performing a thorough reproducibility study on two important aspects of Aydore et al.'s evaluation of RAP. These two aspects are:

1. The benefit of RAP's adaptive component relative to its non-adaptive component was unknown. We determine and quantify this component's utility benefit, finding that it is crucial for enabling RAP to achieve high utility.

2. RAP was only ever evaluated on highly reduced portions of the query space. We instead evaluate RAP's utility across the entire query space, answering up to 50x more queries than in its initial evaluation.

The first aspect is significant because it improves our understanding of how RAP's adaptivity parameters affect its utility and establishes whether RAP's adaptive component is necessary in order to achieve high utility. The second aspect is important because RAP's initial evaluation on highly reduced portions of the query space yielded potentially biased utility results. By instead evaluating RAP across the entire query space, we establish RAP's unbiased utility and determine what impact reducing the query space has on RAP's utility. In order to evaluate both aspects, we had to reimplement RAP from the ground up to improve its efficiency for evaluating large sets of prespecified queries. We then use the new implementation to evaluate both aspects, clarifying the value of the RAP mechanism and thus improving its adoptability for practical uses.

To make the description of our improved evaluation precise, in Section 4.3.1, we define the utility metric used by Aydore et al. and by the prior state-of-the-art mechanisms for answering prespecified queries, which we also use in our evaluations. We then discuss in Section 4.3.2 the details and implications of the two aspects of Aydore et al.'s initial evaluation of RAP that we are improving upon. In Section 4.3.3, we detail the particular obstacle in RAP's initial implementation that prevents its use for our improved evaluation. To overcome this obstacle, we reimplemented RAP from the ground up and make its implementation publicly available[29]. Finally, in Section 4.3.4, we describe how we use our improved implementation to perform our enhanced evaluation of RAP.

[29] https://github.com/bavent/large-scale-query-answering.

Regarding the role of adaptivity in RAP, we not only find that it is crucial to achieving high utility, we quantitatively measure how RAP's adaptivity parameters (T and K) affect its utility. This motivates new, more efficient search strategies to find optimal T and K values, thus reducing RAP's computational burden and privacy cost in practice. Regarding evaluating RAP on the full query space, we find that Aydore et al.'s initial evaluation of RAP on a reduced portion of the query space likely *underestimated* RAP's utility. This was due to their reduced query space having less "sparsity" in the query answers (i.e., a larger portion of the queries they evaluated had non-0 answers). This finding motivates a new line of research on mechanisms for the separate cases of when query answers are and are not sparse. Together, the improved RAP implementation combined with the enhanced evaluation clarifies the value of the RAP mechanism and thus improves RAP's adoptability and usability in practice.

4.3.1 Measuring Utility of Prespecified Queries

We define the concrete utility measure used in prior works to evaluate DP mechanisms that answer prespecified sets of statistical queries. Prior works in this setting measured the utility of DP mechanisms in terms of a mechanism's maximum error over the answers to all queries in the prespecified query set [MSM19; Vie+20; LVW21; Ayd+21]. We refer to this measure of utility as *present utility*, since it is the error on the set of presently available queries, and measure it in terms of the negative of *present error*; i.e., a mechanism with low present error has high present utility and vice versa. This error measure is formally defined as follows.

Definition 4.3.1 (Present error)**.** Let $a = Q(D) = (a_1, \ldots, a_m)$ be the true answers to a given query vector Q on dataset D, and let $\tilde{a} = (\tilde{a}_1, \ldots, \tilde{a}_m)$ be mechanism M's corresponding answers to the query vector. Then err_P is the present error of the mechanism, defined as $\mathrm{err}_P(M, D, Q) = \mathbb{E}_{M(D)} \|a - \tilde{a}\|_\infty$, where the expectation is over the randomness of the mechanism.

We choose the ℓ_∞ norm as the base metric for present error because of its use in Aydore et al.'s evaluation of RAP and because it is the most popular norm utilized in the most closely

related literature [MSM19; Vie+20; LVW21; Ayd+21]. However, other norms (e.g., ℓ_1 and ℓ_2) and even error definitions may be equally valid in the prespecified queries setting depending on the practical use case [Tao+21]. Although we do not empirically evaluate RAP on such alternative definitions, investigating how the findings in this work change based on the error definition is an excellent direction for future work.

To put this utility measure for RAP into context, we compare its present error against the present error of the (ϵ, δ)-DP Gaussian mechanism GM mechanism (Section 1.1.2). This baseline mechanism was chosen because it is able to efficiently (i.e., with low memory and runtime overhead) answer queries at the large scale we evaluate in this work. For alternative state-of-the-art DP query answering mechanisms, we leave their reimplementation, scaling, and reevaluation as future work. Additionally, as in Aydore et al. [Ayd+21], we provide further context for RAP's present error by comparing it against the All-0 mechanism, which trivially answers 0 to every query.

4.3.2 Focus of RAP's Reevaluation

We now detail the two primary aspects of Aydore et al.'s evaluation of RAP that we enhance in this chapter and how their origins trace back to a particular challenge in RAP's initial implementation.

Adaptivity Evaluation: The first aspect that we address in RAP's reevaluation is how RAP's adaptive component affects its utility. To provide context, we briefly describe the non-adaptive form of RAP. We then describe the adaptive form of RAP and the motivation behind its design. Finally, we detail how Aydore et al.'s evaluation of RAP omitted studying the adaptive component's effect on utility, and we describe why that is an issue.

In its non-adaptive form, the RAP mechanism essentially reduces to privately answering the full query vector Q with the Gaussian Mechanism, then applying the RP mechanism to generate a synthetic dataset. This non-adaptive form of the RAP mechanism is a novel reimagining of

the classic *Projection Mechanism* [NTZ13], a near-optimal but computationally intractable mechanism for answering prespecified queries. By leveraging a relaxation of the query space and utilizing EEDQs, Aydore et al. describe how their non-adaptive RAP mechanism can use modern tools (e.g., GPU-accelerated optimization) to efficiently generate a relaxed synthetic dataset that can hypothetically answer the prespecified queries with low (albeit non-optimal) error. Moreover, they prove a theoretical result (Theorem 4.1, [Ayd+21]) which confirms the power of the non-adaptive RAP mechanism, achieving a $\sqrt{d'}$ factor of utility improvement over the prior state-of-the-art mechanism.

Aydore et al. go on to describe the full adaptive form of RAP parameterized by T and K. This adaptive form of RAP optimizes the synthetic dataset iteratively over T separate rounds, in each round adaptively selecting K new queries to incorporate into the optimization procedure. Their stated motivation for introducing adaptivity into RAP was to more wisely expend the privacy budget by adaptively optimizing over a small number of "hard" queries. They conjecture (without a result similar to their Theorem 4.1) that such adaptivity will result in higher utility than that achieved by the non-adaptive form of RAP.

Aydore et al. then perform an empirical evaluation of RAP across a range of parameters and datasets and establish that it achieves state-of-the-art utility — however, the utility benefits of RAP's adaptivity are left unanalyzed. Specifically, in all evaluations, they report the best utility of RAP across $2 \leq T \leq 50$ and $5 \leq K \leq 100$. There are two issues related to this.

1. The values of T and K that achieved the maximum utility are not reported, only what that maximum utility was. Thus, it is unclear how these parameters affect utility. This is problematic in practice because evaluating RAP on multiple choices of T and K is computationally expensive and consumes a portion of the overall differential privacy budget.

2. The non-adaptive form of RAP is not empirically evaluated. Without evaluating the non-adaptive RAP mechanism as a baseline, there is no meaningful way to understand or measure the benefit of adaptivity.

Combined, these two issues leave open the question of how valuable the adaptive component of RAP is and to what extent its adaptivity affects utility.

Query Space Evaluation: The second aspect we address in RAP's reevaluation is how reducing the query space affects RAP's utility for answering k-way marginals. First, we describe the motivation behind evaluating this aspect: that for computational ease, Aydore et al. only evaluated RAP on a reduced portion of the query space. We then detail how this reduction may have biased their evaluation's results.

Aydore et al.'s empirical evaluation focuses on RAP's utility for answering k-way marginals, specifically 3-way and 5-way marginals. Reviewing the code of their published RAP implementation, we determined that a heuristic filtering criterion of the query space was being applied to remove any "large" marginals from possible evaluation. Specifically, any marginal with more consistent queries than the number of records in the dataset (n) was not considered for evaluation. The impact that filtering had on the evaluated workloads varied depending on k and n. For instance, with 3-way marginals on the ADULT dataset, the filtering criterion removed the top 24% largest 3-way marginals, which accounted for over 90% of all consistent queries. With 5-way marginals on the ADULT dataset, this filtering criterion removed the top 92% largest 5-way marginals, which accounted for over 99.99% of all consistent queries.

Discussing this discrepancy directly with the authors [AS21] revealed that the filtering criterion was an intentional choice meant to reduce the computational burden during experimentation, and they conjectured that removing this criterion and rerunning all experiments would yield results comparable to those obtained by increasing the workload size. Since all baseline mechanisms were evaluated on the same query vectors, the filtering criterion does not result in favorable utility for RAP relative to the prior state-of-the-art mechanisms that serve as their baselines. However, for marginals with a significantly larger number of consistent queries than n, most queries will evaluate to 0 by a Pigeonhole principle argument. Thus, the filtering criterion may result in favorable utility for RAP relative to the naive baseline mechanism that they consider

in their work: `All-0`, the mechanism which outputs 0 as the answer to every query. This leaves open the question of `RAP`'s utility on large, unfiltered query spaces, both in absolute terms and relative to the baseline `All-0` mechanism.

4.3.3 Reimplementing `RAP`

We now describe why these two aspects cannot be evaluated using Aydore et al.'s initial `RAP` implementation: Briefly, the amount of memory required by the implementation is inordinate. We then detail how we overcome this challenge by reimplementing `RAP` in a way that trades off a significant amount of memory usage for a potential increase in runtime.

Conceptually, both aspects could be evaluated using Aydore et al.'s published code. However, evaluating either the non-adaptive form of `RAP` or a larger portion of the query space both lead to the same obstacle: Aydore et al.'s `RAP` implementation requires an inordinate amount of memory to answer the corresponding large number of queries. We have identified several portions of their code where this memory bottleneck occurs, all of which fail to execute either when the total number of consistent queries is "too large" or when any marginal has "too many" consistent queries. Consequently, Aydore et al. were unable to evaluate either the non-adaptive form of `RAP` or a significant portion of the k-way marginals' consistent query space.

The high-level idea behind our approach for overcoming this implementation challenge is to trade off some of `RAP`'s required memory for a potential increase in its runtime. Our motivation for this approach is inspired by recent advances in the differentially private deep learning literature. In particular, the canonical DP-SGD mechanism [Aba+16] for training machine learning models with differential privacy had been plagued by poor computational performance due to several of its underlying operations (e.g., per-example gradient clipping, uniformly random batch sampling without replacement, etc.) not being natively supported by modern machine learning frameworks. More recently, however, several highly performant DP-SGD implementations [Pap19; You+21; SVK21] have been deployed that dramatically decrease the mechanism's

runtime in exchange for a mild increase in its memory usage. To our knowledge, our high-level approach is the first in the DP literature to make practical use of this trade-off in the opposite direction: decreasing the mechanism's memory requirement by increasing its runtime.

Concretely, we overcome this implementation challenge by reimplementing RAP via the following high-level steps. First, we reduce the maximal memory requirement in RAP's original implementation caused by the original implementation's implicit evaluation of all marginals (or, more generally, all thresholds) in parallel. We accomplish this by evaluating each marginal (or threshold) sequentially, thus distributing the computational burden. To further reduce the overall memory requirement, rather than explicitly enumerating and storing every query consistent with each marginal (threshold), we represent the queries implicitly and only convert a query to its explicit representation when it is needed for evaluation. To evaluate arbitrary sets of such individual queries, we implement the core EEDQ evaluation function from the ground up by designing a simple, direct function to evaluate arbitrary predicates efficiently. With such a function implemented, we then leverage a combination of powerful language features — namely vectorizing maps and just-in-time compilation in JAX [Bra+18] — to enable efficient evaluation, summation, and differentiation of large sets of predicates without exceeding memory constraints.

In addition to these implementation improvements, which primarily serve to reduce RAP's memory requirement, we additionally incorporate an algorithmic improvement based on recent theoretical findings to help offset the increased runtime from our aforementioned deparallelization step. Specifically, by trivially adapting the *Oneshot Top-K Selection with Gumbel Noise* mechanism [DR19; CR21] to our setting, we replace RAP's iterative Adaptive Selection (AS) mechanism with the more efficient Oneshot Adaptive Selection (OSAS) mechanism in Alg. 4.4. The results of [DR19] prove that the OSAS mechanism is probabilistically equivalent to AS (i.e., both mechanisms have identical output distributions and thus achieve identical privacy and utility), but OSAS requires only one pass over a set of values to select the top-K instead of the K passes that AS requires.

Algorithm 4.4 Oneshot Adaptive Selection (OSAS) Mechanism

Input

- D, D': The dataset and synthetic dataset.
- $Q, \hat{Q}$: A vector of all statistical queries and their corresponding surrogate queries.
- Q_s: A set of already selected queries.
- K: The number of new queries to select K.
- ρ: Differential privacy parameter.

Body

1: Let $\Delta = (|\hat{q}_i(D) - \hat{q}_i(D')| : q_i \in Q \setminus Q_s)$.
2: Let I denote the indices of the top-K values of: $\Delta_i + Z_i$, where $Z_i \overset{\text{iid}}{\sim} \texttt{Gumbel}\left(\sqrt{\frac{K}{2\rho|D|^2}}\right)$.
3: Let $\tilde{a}_i = \texttt{GM}(D, q_i, \frac{\rho}{2K}) \quad \forall i \in I$.
4: Let $Q_s = Q_s \cup \{q_i\}_{i \in I}$.
5: **Return:** Q_s and $\tilde{a} = (\tilde{a}_i : q_i \in Q_s)$.

Figure 4.1 compares our new implementation to Aydore et al.'s original implementation without filtering out any large marginals. Specifically, this figure shows the runtimes of both implementations executing the non-adaptive and adaptive variants of RAP given the same amount of GPU memory on two datasets across a range of workload sizes[30]. We find that for the non-adaptive variant of RAP, the original implementation was only able to evaluate tiny workloads, while our new reimplementation was able to evaluate massive workloads (albeit with a very high runtime); this represents a 500x improvement in memory efficiency for our reimplementation. For the adaptive variant of RAP (specifically, with T=16 and K=4), we find that our reimplementation's runtime is comparable to the original implementation's — outperforming it slightly on one dataset while being outperformed slightly on the other. On the ADULT dataset, both implementations were able to exhaustively evaluate the complete space of marginals. On the LOANS dataset, the original implementation was able to evaluate marginal workloads of size 256 consistently but was unable to evaluate the largest workload size of 1024 consistently; this represents up to a 4x improvement in memory efficiency for our reimplementation.

[30]The runtimes for both implementations (and all subsequent evaluations in this chapter) were measured on an Nvidia RTX 3090 consumer GPU with 24 GB VRAM.

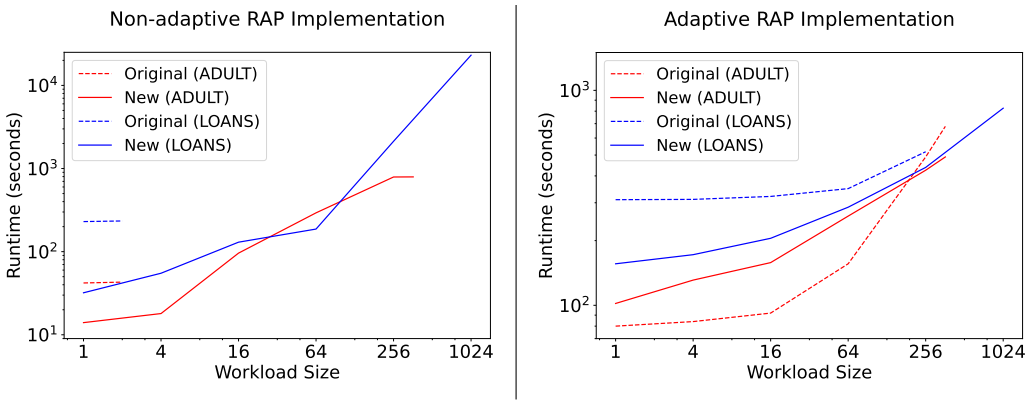


Figure 4.1: Runtime evaluations of non-adaptive and adaptive RAP variants on the original implementation and reimplementation, on both ADULT and LOANS datasets.

4.3.4 Reevaluating RAP

Using our new implementation, we reevaluate both the adaptivity and query space aspects of RAP, enabling new findings. We start by simply establishing RAP's present utility for answering k-way marginals on unbiased random samples of the full marginal space (i.e., without filtering out any "large" marginals). This results in RAP answering approximately 50x more queries at its peak than in Aydore et al.'s initial evaluation on filtered marginals. We then use these results to analyze the role that adaptivity plays in RAP's utility. Finally, we address the question of whether filtering the large marginals out of RAP's evaluation significantly impacts its utility in order to determine if the filtering criterion is a reasonable heuristic to apply to reduce RAP's computational burden in future evaluations. This improved implementation and reevaluation, taken together, demonstrate that RAP is a feasible and valuable mechanism for practical, real-world use cases. Furthermore, in conjunction with our improved implementation, our findings enable new capabilities, such as more efficient search strategies for optimal T and K parameters.

Evaluation Datasets

As in prior works on evaluating DP mechanisms that answer statistical queries [Ayd+21; Vie+20; MSM19], all empirical evaluations use the ADULT [Fra10] and LOANS [Vie+20] datasets with the same preprocessing. Table 4.2 contains a high-level description of each dataset.

Dataset	Records	Features	Binarized Features
ADULT	48,842	14	588
LOANS	42,535	48	4,427

Table 4.2: Datasets for empirical evaluations. Binarized features represent the features after a transformation via one-hot encoding.

4.3.4.1 k-way Marginal Evaluation of RAP

To begin RAP's reevaluation, we concretely establish its utility on a larger portion of the query space than previously considered by Aydore et al. Specifically, we evaluate RAP's present error for answering uniformly random workloads of 3-way marginals across a range of parameters on both the ADULT and LOANS datasets, and we do so *without any* thresholding criterion to filter out "large" marginals. This results in RAP answering approximately 50x as many queries as in its original evaluation by Aydore et al. — a contribution that was not possible utilizing RAP's initial implementation. Table 4.3 provides a reference for the parameter ranges in this experiment. For each setting of parameters, we evaluate the adaptive variant of RAP across a range of T and K values and report the combinations that achieve minimal present error[31]. We separately evaluate the non-adaptive ($T = 1, K = m$) variant of RAP across the same range of parameters to answer the question of whether or not there is any benefit to RAP's adaptivity. Additionally, as baselines, we evaluate the present utility of the All-0 and GM mechanisms, enabling us to put the utility of RAP into context. The results of this experiment are visualized in Figure 4.2.

Several immediate conclusions can be drawn from these results. The first is that while the non-adaptive variant of RAP achieves lower error than the GM baseline, its utility is nearly identical to the All-0 baseline for all but the smallest workload sizes. This result likely stems from the fact that the answers to the large majority of a marginal's consistent queries are 0 or nearly 0, with only a small percentage of answers having larger values. Since the non-adaptive variant

[31]We note that evaluating multiple hyperparameter settings to select the optimal one has an impact on privacy which, in practice, must be accounted for. Such accounting is an active area of research [LT19; PS22]. Thus, like other works on DP workload answering mechanisms, we consider it out of the scope of this work. Alternatively, a similar but public dataset may be used to perform hyperparameter selection without any impact on the final privacy guarantee.

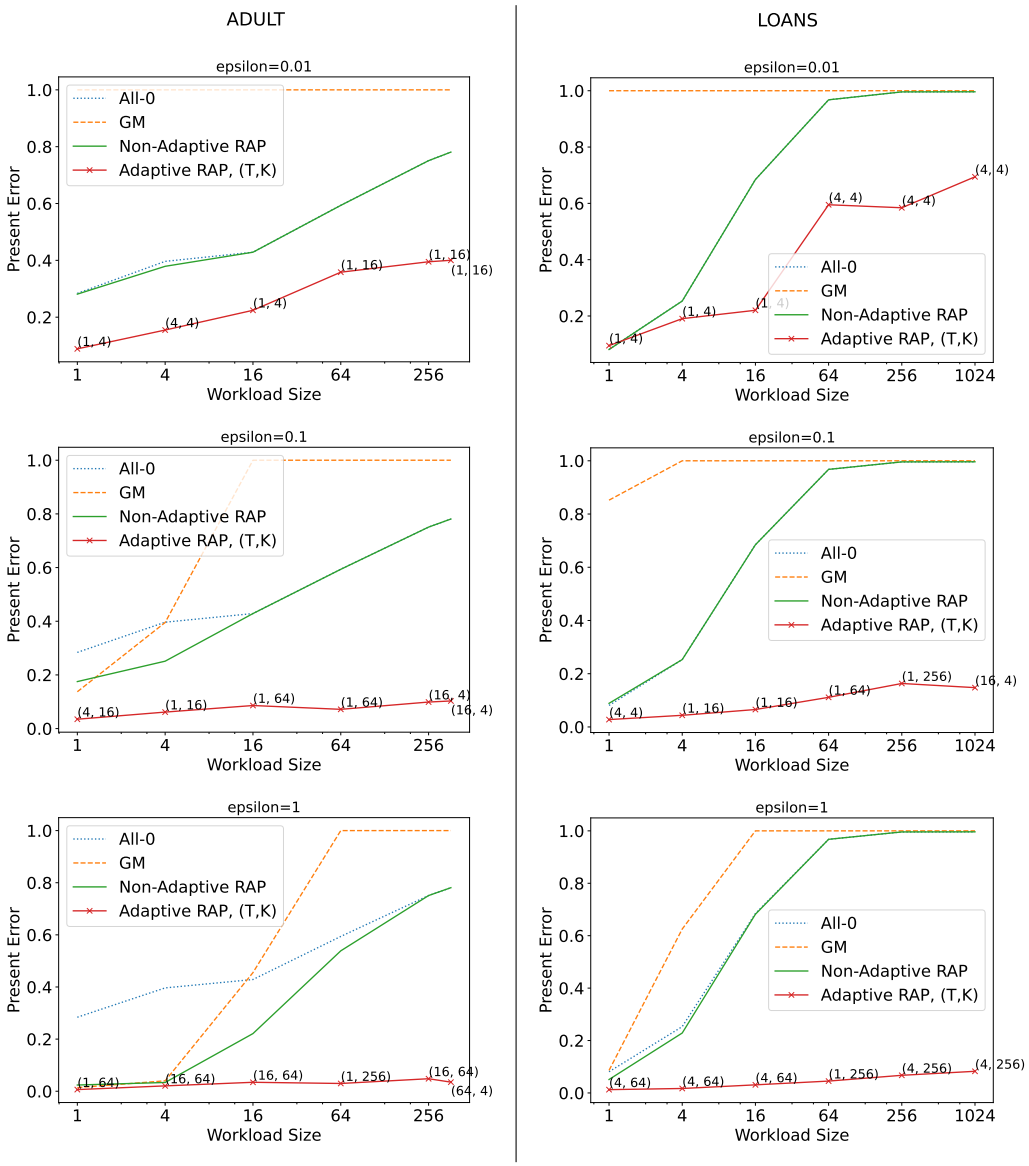


Figure 4.2: Present error across a range of parameters and datasets for the adaptive and non-adaptive variants of `RAP`, the `GM` baseline, and the `All-0` baseline. Present error for the adaptive variant of `RAP` is computed as the minimal error across the range of T and K values (with the specific (T, K) pair that achieved the minimum reported at each point).

Primary Mechanism	RAP
Baseline Mechanisms	All-0, GM
Utility Measure	err_P
D	ADULT, LOANS
ϵ	$0.01, 0.1, 1$
δ	$1/\lvert D\rvert^2$
$\lvert W\rvert$	$1, 4, 16, 64, 256$
n'	10^3
T	$1, 4, 16, 64$
K	$4, 16, 64, 256, m$
k	3

Table 4.3: Experimental reference table for reevaluating RAP's utility on k-way marginals.

of RAP first privatizes the answers to all queries, in the synthetic dataset optimization procedure, it is likely unable to distinguish between the few answers that are genuinely larger than 0 vs. the outliers that are only large due to random chance. The second conclusion is that the adaptive variant of RAP achieves significantly lower present error than the non-adaptive RAP variant and the baselines. This implies that RAP's adaptivity is critical for achieving low error and thus warrants a more thorough investigation into T and K's precise impact on utility. Because of this finding, taken together with our findings in Figure 4.1 that the adaptive variant is significantly faster than the non-adaptive variant, we can safely omit the non-adaptive variant of RAP from further consideration in this work.

4.3.4.2 Role of Adaptivity

In this next experiment, we seek to understand the precise impact that T and K have on RAP's utility. From Figure 4.2, we are only able to glean that RAP typically achieves minimal error via smaller values of T in conjunction with relatively larger values of K. However, these values of T and K vary dramatically across parameter settings and datasets. Moreover, Figure 4.2 provides no information about RAP's utility for T and K combinations that did not achieve minimal error. To better understand the role these parameters play in RAP's utility, we examine the present error

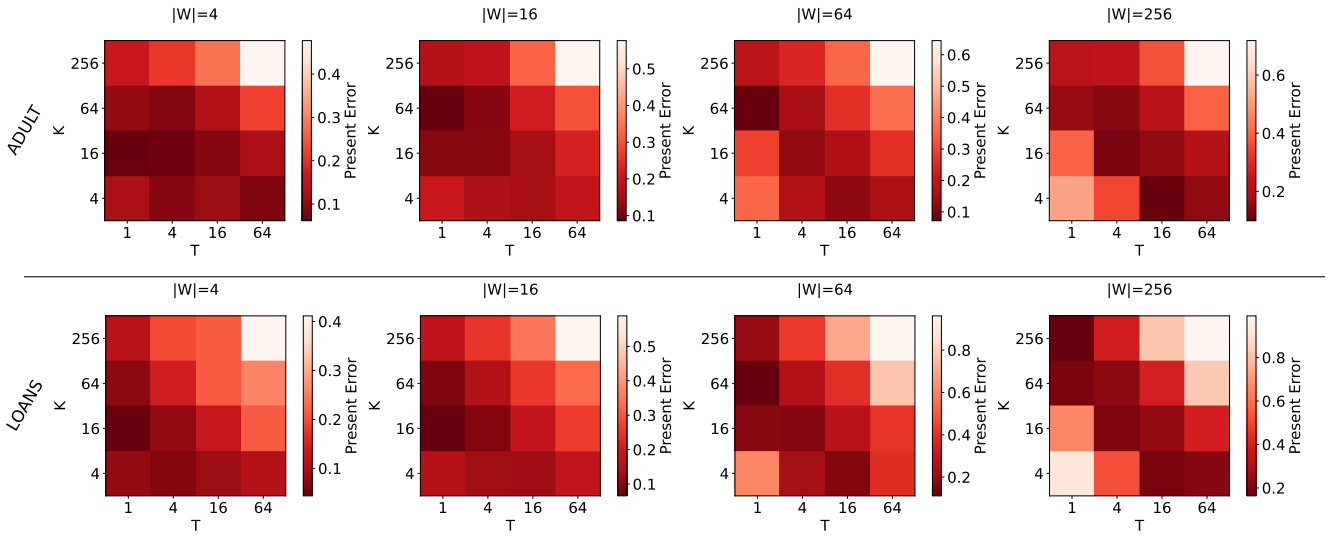


Figure 4.3: Present error across a range of workload sizes with $\epsilon = 0.1$ for the adaptive variant of RAP at every combination of T and K value considered.

of the adaptive variant of RAP for every (T, K) pair across the same parameter settings from Table 4.3. The results of this experiment are shown in Figures 4.3 and 4.4.

The heatmaps in both figures provide interesting insight into RAP's adaptivity. In Figure 4.3, with ϵ fixed at 0.1, we see no single (T, K) value or region that consistently achieves minimal error across all workload sizes. Instead, we notice that at each workload size, there is some diagonal banding at around a fixed region of $T \cdot K$ that achieves approximately minimal error. That is, for any particular workload size, let (T^*, K^*) denote the T and K value that induces minimal error for RAP across our considered range of T, K values, and let $c^* := T^* \cdot K^*$. We see that for other (T, K) pairs such that $T \cdot K \approx c^*$, the corresponding error is typically comparable to the minimal error. Moreover, we see that as $T \cdot K$ diverges from c^*, RAP's error increases essentially monotonically. We hypothesize that for $T \cdot K \ll c^*$, RAP's error is relatively high because RAP had not answered and optimized over a sufficient number of queries. For $T \cdot K \gg c^*$, we hypothesize that RAP's error is relatively high because the privacy budget is spread too thin across answering a large number of queries, resulting in RAP utilizing overly noisy queries to optimize its underlying synthetic dataset.

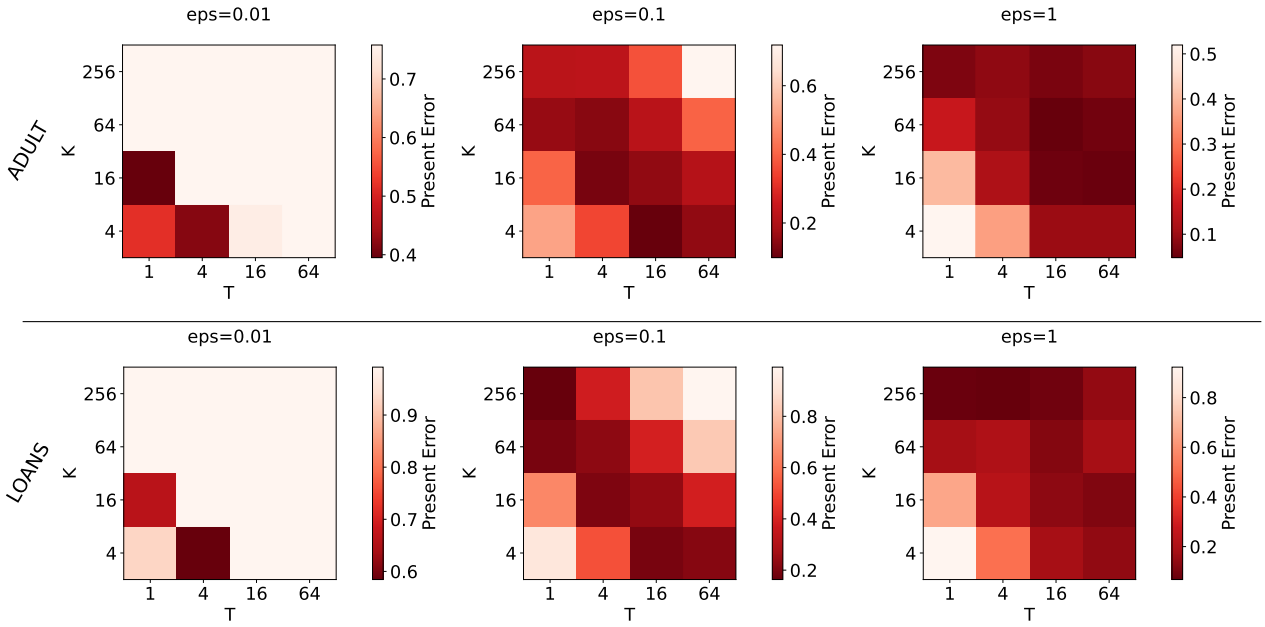


Figure 4.4: Present error across a range of ϵ values with $|W| = 256$ for the adaptive variant of RAP at every combination of T and K value considered.

These hypotheses are supported by the results in Figure 4.4. Specifically, as ϵ becomes larger, not only does the minimal error of RAP decrease but the T and K values that achieve the minimal error (along with their corresponding diagonal bands) are pushed to increasingly large values. Taken together, these results imply that to achieve low error, RAP primarily requires answering and optimizing over a *specific number* of queries — it is less important whether those queries are answered in small batches over a large number of adaptive rounds or in large batches over a small number of adaptive rounds.

This finding is important to RAP's usefulness in practice, as it motivates improved search strategies for optimal (T, K) values. Improved search strategies (beyond the naive $N \times N$ grid search that we performed) are important for two reasons.

1. Evaluating RAP across a range of T and K values can be computationally expensive. Thus, improved search strategies would decrease the computational cost. Alternatively, at a fixed computational cost, improved search strategies would allow RAP to be evaluated across a larger set of T and K values.

2. In practice, each evaluation of RAP on any (T, K) setting consumes a portion of the privacy budget, even though only the optimal setting is ultimately chosen. Thus, reducing the total number of evaluated (T, K) settings enables more efficient use of the overall privacy budget.

We provide one example of an improved search strategy over the naive $N \times N$ grid search strategy as follows. First, the observed monotonicity of present error about c^* could be leveraged to binary search for a $c \coloneqq T \cdot K$ setting along the positive diagonal that achieves approximately minimal error. Then, a linear search across all (T', K') settings such that $T' \cdot K' = c$ could be performed to compute the setting that achieves minimal error. Relative to the grid search, this strategy would yield an $\mathcal{O}(N)$ factor improvement both in the portion of the privacy budget consumed as well as in the computational cost.

4.3.4.3 Utility Impact of Filtering Marginals

In the final experiment, we analyze what impact filtering out marginals with "too many" consistent queries has on RAP's utility. Recall that in Aydore et al.'s evaluation, as a heuristic to reduce the computational burden of experimentally evaluating RAP, any marginal was removed from consideration if it contained more consistent queries than the number of records in the underlying dataset. Here, we compare how RAP's utility is affected by this marginal filtering criterion. We initiate this comparison by reevaluating RAP with and without the filtering criterion. We do so across the range of parameters in Table 4.3, and we record the minimal present error of RAP at each parameter setting across all (T, K) pairs. We then perform two analyses on these results, one focusing on how the workload size affects RAP's present error with and without marginal filtering and another analyzing how the total number of queries affects RAP's present error. We determine that RAP's present error is impacted by filtering large marginals. More specifically, we find that when holding the number of queries that RAP evaluates constant, filtering large marginals *increases* RAP's present error.

Influence of Workload Size on Utility Aydore et al. hypothesized that removing the marginal filtering criterion would cause RAP's present error to increase comparably to the error increase induced by increasing the workload size. To test this hypothesis, we perform a standard nested regression analysis [GH06] on the RAP evaluation results. For brevity, we state the steps of this analysis and then immediately jump to the results, deferring the regression details to the end-of-chapter Appendix 4.A.

At a high level, the steps for this analysis are as follows. For the ADULT and LOANS datasets separately, we define a full regression model to account for the following three variables' (and their interactions') impact on RAP's present error: the DP level ϵ, the workload size $|W|$, and whether the marginal filtering criterion was applied. We also define a restricted regression model that accounts for ϵ and $|W|$ but does not distinguish whether or not a result had the marginal filtering criterion applied. Following the standard approach for a nested regression analysis, we first determine whether the full regression model is a good fit for the RAP evaluation results (based on the fitted model's adjusted R^2 value, F-statistic p-value, and omnibus p-value). We then compare the full model's fit to the restricted model's fit by performing a likelihood ratio test, analyzing the p-value of the resulting χ^2 statistic. Since the full model only differs from the restricted model in that it accounts for whether the marginal filtering criterion was applied, we can conclude that if the fit of the full model is both statistically sound and statistically significantly better than that of the restricted model, then the marginal filtering criterion impacts RAP's present error.

From this analysis, Figure 4.5 shows the fitted full regression model on both datasets with ϵ fixed at 0.1. We find that the full regression models for both datasets fit the RAP evaluation results well. Thus, we perform the aforementioned likelihood ratio test against the restricted models for

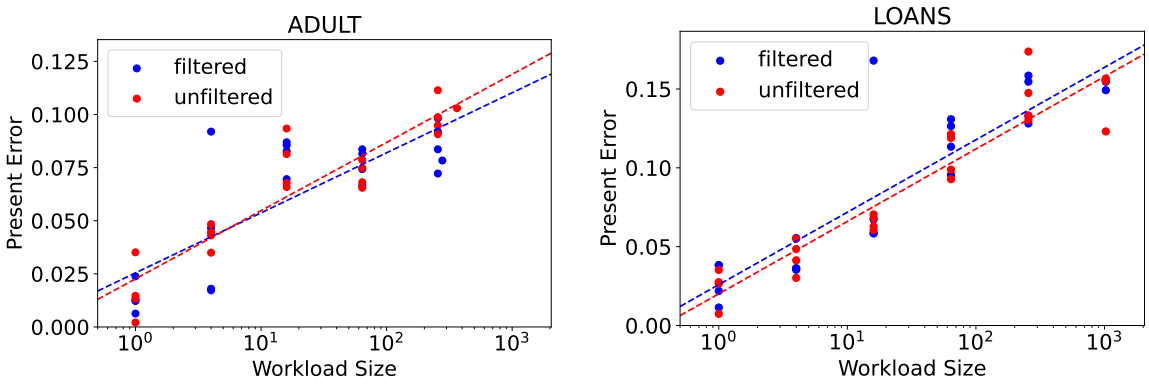


Figure 4.5: Regression models for each dataset of RAP's present error vs. workload size for results from filtered and unfiltered marginals, at $\epsilon = 0.1$.

each dataset. The corresponding p-values for the models on the ADULT and LOANS RAP evaluations were 0.026 and 0.623, respectively[32]. The small p-value for the model corresponding to the RAP evaluations on the ADULT dataset enables us to conclude that the marginal filtering criterion does have an impact on RAP's present error. However, the coefficients (and their corresponding p-values) in the full regression model do not indicate any clear, statistically significant trend for how the workload size impacts the present error when comparing the filtered vs. unfiltered RAP evaluations. Moreover, regardless of the workload size, due to the lack of significance in many of the coefficients' p-values, we are unable to use this model to confidently determine the marginal filtering criterion's impact on RAP's present error. Thus, although we are able to conclude that incorporating the marginal filtering criterion into RAP's evaluation does impact its present error, we are unable to confirm Aydore et al.'s hypothesis on the precise nature of this impact.

Influence of Number of Queries on Utility We now perform a more direct analysis of the marginal filtering criterion's impact on RAP's utility. Our previous regression analysis assessed Aydore et al.'s hypothesis regarding the filtering criterion's influence on RAP's present error as a

[32]We report the individual p-values for all statistical hypotheses tested. However, we control the family-wise error rate α (i.e., the probability α that at least one "false positive" finding will occur) using the Holm–Bonferroni method [Hol79]. At the $\alpha = 0.05$ level, no conclusions based on the individual p-values change when the Holm–Bonferroni method is applied.

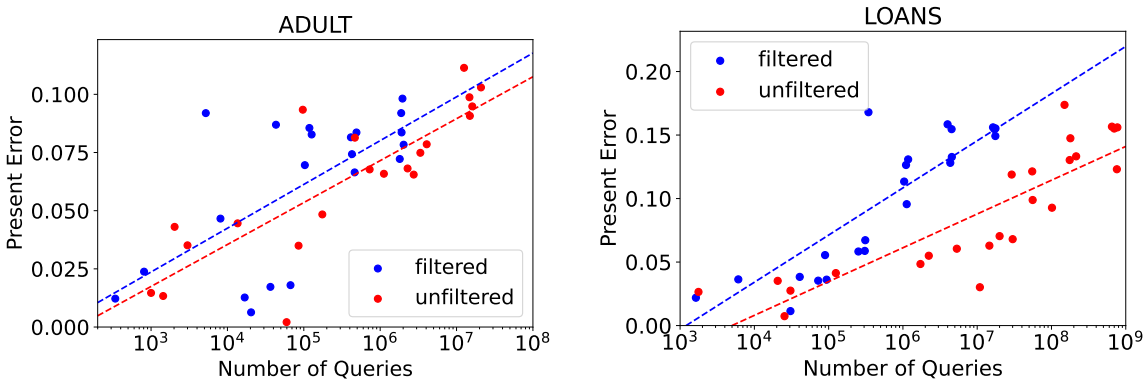


Figure 4.6: Regression models for each dataset of RAP's present error vs. number of queries for results from filtered and unfiltered marginals, at $\epsilon = 0.1$.

function of workload size. However, the filtering criterion does not affect workload size directly — it only affects the total number of queries consistent with the marginals in the workload. As such, we believe that a more informative assessment would be to analyze the marginal filtering criterion's influence on RAP's present error as a function of the total number of consistent queries it evaluates.

We perform this assessment using precisely the same statistical analysis and regression models as before, only now having the full and restricted models account for the total number of queries rather than workload size. Figure 4.6 shows the fitted full regression models on both datasets with ϵ fixed at 0.1. Again, the full regression models for both datasets fit the RAP evaluation results well, allowing us to then test these full models against their corresponding restricted models. The corresponding p-values of the likelihood ratio tests for the models on both the ADULT and LOANS RAP evaluations were less than 0.0001, indicating that the filtering criterion has a statistically significant impact on RAP's present error (for both datasets this time). The results from the figure for both datasets visually imply that including the filtering criterion increases RAP's present error for any given number of queries and that this increase worsens as the total number of queries grows. By examining the coefficients (and their corresponding p-values) of the full regression models on both datasets, we confirm that this visual trend holds statistically as well.

These results match intuition: for a result with the filtering criterion to have approximately the same number of queries as a result without, the result with filtering would likely have corresponded to a larger size workload. A larger size workload with the same number of queries implies a more diverse set of queries, whereas a smaller workload with the same number of queries implies a less diverse set of queries with sparser support (i.e., more of the queries evaluate to 0). Thus, we conclude that Aydore et al.'s initial evaluation of RAP — especially for the highly filtered 5-way marginals — likely overestimates RAP's present error. Moreover, this finding motivates a new branch of work on large-scale query answering for the separate cases of when the queries have dense support vs. sparse.

4.4 Extending RAP's Applicability

Having confirmed that RAP is indeed a useful mechanism for efficiently evaluating k-way marginals on a large scale in the prior section, we now address our third contribution for the setting where queries are prespecified: extending RAP's applicability by expanding the class of queries that it is able to evaluate. We begin by discussing the motivation behind this contribution. We then describe what we expand the query class to (r-of-k thresholds) and how we accomplish it. Finally, we detail the empirical evaluations we perform on RAP within this expanded query class to quantify its utility and feasibility, finding that RAP efficiently evaluates r-of-k thresholds with high utility.

4.4.1 Motivation

We contextualize the motivation for this contribution by considering the contributions of prior works. Prior work on answering statistical queries in practical settings has been focused on relatively simple classes of statistical queries — most popularly, k-way marginals (Definition 4.2.2),

as these are a useful query class which is evaluable within a reasonable computational budget [Bar+07; TUV12; Gup+13; Cha+14]. Aydore et al. claim that their gradient-based RAP mechanism [Ayd+21] is able to answer queries from richer classes. In addition to evaluating k-way marginals, they demonstrated this claim by briefly evaluating a new class of queries, 1-of-k thresholds (Definition 4.2.3). However, 1-of-k thresholds are essentially a negation of k-way marginals. As such, Aydore et al. were able to evaluate RAP on 1-of-k thresholds by reusing virtually the same class of EEDQs and the same underlying implementation as they used for k-way marginals. Thus, although their evaluation demonstrated that RAP attains high utility on both query classes, these choices of query classes were not fully convincing in demonstrating that RAP is effective for answering truly richer classes of queries. Therefore, it remained an open question whether RAP is able to answer richer, more general query classes.

4.4.2 Expanding the Query Class

To extend RAP's applicability, we develop the mathematical and computational machinery necessary for RAP to evaluate a class of queries that generalizes both k-way marginals and 1-of-k thresholds: r-of-k thresholds (Definition 4.2.4). We first describe this query class in detail, then derive its corresponding EEDQs. Finally, we show how we optimize the derived EEDQs to be more efficiently evaluable, greatly reducing RAP's per-query evaluation time.

4.4.2.1 Generalizing to r-of-k Thresholds

Informally, an r-of-k threshold query counts what fraction of datapoints in the dataset have at least r out of the k specified attributes. Thus, it strictly generalizes both k-way marginals (when $r = k$) and 1-of-k thresholds (when $r = 1$). r-of-k thresholds are a useful generalization because they allow for more expressive, dynamic queries beyond the rigid "everything" ($r = k$) or "anything" ($r = 1$) queries that were previously studied.

The challenge when expanding RAP's evaluation to r-of-k thresholds is deriving corresponding EEDQs. r-of-k thresholds cannot trivially reuse the EEDQs relied upon by Aydore et al. to evaluate k-way marginals and 1-of-k thresholds. Thus, we must derive new EEDQs for r-of-k thresholds, and we accomplish this by generalizing the EEDQs of k-way marginals and 1-of-k thresholds. Towards this, we first reframe the standard definition of r-of-k thresholds to enable explicit accounting of all possible combinations of matching and non-matching terms.

Definition 4.4.1 (r-of-k thresholds, Alternative)**.** An r-of-k threshold query $q_{\phi_{S,y,r}}$ is a statistical query whose predicate is specified by a positive integer $r \leq k$, a set S of k features $f_1 \neq \cdots \neq f_k \in [d]$, and a target $y \in (\mathcal{X}_{f_1} \times \cdots \times \mathcal{X}_{f_k})$. Let $\mathcal{R}$ denote the set of all partitions (R_+, R_-) of the k features in S, such that each $|R_+| \geq r$ and each corresponding $R_- = S - R_+$. The predicate $\phi_{S,y,r}$ is then given by

$$\phi_{S,y,r}(x) = \begin{cases} 1 & \text{if } \bigvee_{(R_+,R_-)\in\mathcal{R}} \left(\bigwedge_{i\in R_+}(x_{f_i} = y_i) \bigwedge_{i\in R_-}(x_{f_i} \neq y_i) \right) \\ 0 & \text{otherwise.} \end{cases}$$

Note that at most one partition in $\mathcal{R}$ will satisfy the predicate.

We now use this equivalent definition of r-of-k thresholds queries to design corresponding EEDQs. For k-way marginals, Aydore et al. used *product queries* (Definition 4.2.7) as EEDQs, which simply compute the product of a datapoint's values at the k specified indices. For r-of-k threshold queries, we generalize product queries in the following ways. First, we expand the product queries to explicitly include both positive and negated terms, which we refer to as *generalized product queries*.

Definition 4.4.2 (Generalized Product Query)**.** Given two disjoint subsets of features $T_+, T_- \subseteq [d']$, the generalized product query $\hat{q}_{\hat{\phi}_{T_+,T_-}}$ is a surrogate query parameterized by $\hat{\phi}_{T_+,T_-}$ which is defined as

$$\hat{\phi}_{T_+,T_-}(x) = \prod_{i\in T_+} x_i \prod_{i\in T_-} (1 - x_i).$$

Informally, a generalized product query effectively serves as a "sub"-EEDQ for the conjunction portion of a single partition of $\phi_{S,y,r}(x)$ in Definition 4.4.1.

Then, leveraging this alternative definition of r-of-k thresholds together with generalized product queries, we define a new class of EEDQs in Definition 4.4.3: *polynomial threshold queries*.

Definition 4.4.3 (Polynomial Threshold Query)**.** Given a subset of features $T \subseteq [d']$ and integer r, let Υ denote the set of all partitions (T_+, T_-) of T such that each $|T_+| \geq r$ and each corresponding $T_- = T - T_+$. The *polynomial threshold query* $\hat{q}_{\hat{\phi}_{T,r}}$ is a surrogate query parameterized by $\hat{\phi}_{T,r}$ which is defined in terms of the generalized product query predicates as

$$\hat{\phi}_{T,r}(x) = \sum_{(T_+,T_-)\in\Upsilon} \hat{\phi}_{T_+,T_-}(x).$$

Informally, a polynomial threshold query computes the sum of generalized product queries across all $\sum_{t=r}^{k} \binom{k}{t}$ partitions of T, where T is constructed identically as in Lemma 4.2.8; i.e., for every $i \in S$, we include in T the coordinate corresponding to $y_i \in \mathcal{X}_{f_i}$.

4.4.2.2 Optimizing the Evaluation of Polynomial Threshold Queries

Evaluating polynomial threshold queries can be computationally expensive due to their combinatorial expansion and summation of generalized product query predicates. Therefore, optimizing their definition to be efficiently evaluable is important for enabling RAP to evaluate large sets of r-of-k thresholds. Towards this, we present two optimizations that can be used together, which significantly improve the practical runtime of RAP.

The first optimization is inspired by Aydore et al.'s implicit reduction of 1-of-k threshold queries to k-way marginal queries. They accomplished this by recognizing that a 1-of-k threshold predicate is the negation of a k-way marginal predicate on a negated datapoint; i.e., $\phi_{S,y,1}(x) = 1 - \phi_{S,y,k}(1-x)$. This equivalence enabled them to efficiently reuse the k-way marginals' EEDQs (product queries) in RAP's evaluation. Applying this concept more generally to computing an r-of-k threshold predicate $\phi_{S,y,r}(x)$, the idea is that when $r \leq k/2$, it is logically equivalent to compute

the negation of a corresponding predicate (with $r' = k - r + 1$) on the negated datapoint; i.e., $\phi_{S,y,r}(x) = 1 - \phi_{S,y,r'}(1 - x)$. The benefit of utilizing this equivalence when using a polynomial threshold query as the EEDQ to evaluate $\phi_{S,y,r}(x)$ is that *at most* $\lceil k/2 \rceil$ different partition sizes now need to be computed over, compared to at most k when not utilizing this equivalence. The computational savings from utilizing the equivalence are especially apparent when r is small, as it leads to an exponential (in k) reduction in the required number of predicate evaluations.

For the second optimization, the goal is to eliminate the need to explicitly account for the negated terms in our alternative definition of r-of-k thresholds (Definition 4.4.1), as this necessitates the computation of the product of negated values in generalized product queries (Definition 4.4.2). Removing the conjunction over negated terms from Definition 4.4.1 yields a logically equivalent predicate, i.e.,

$$\phi_{S,y,r}(x) = \begin{cases} 1 & \text{if } \bigvee_{(R_+,R_-)\in\mathcal{R}} \bigwedge_{i\in R_+} (x_{f_i} = y_i) \\ 0 & \text{otherwise.} \end{cases}$$

However, more than one partition of $\mathcal{R}$ may now satisfy the predicate. As a result, analogously eliminating the product of negated values from the generalized product query definition (reducing it to a standard product query) would cause the summation in the polynomial threshold query's definition (Def 4.4.3) to overcount. To eliminate computing the product of negated values while simultaneously remedying this overcount, we utilize the principle of inclusion-exclusion to equivalently redefine polynomial threshold queries purely in terms of standard product queries (Definition 4.2.7).

Definition 4.4.4 (Polynomial Threshold Query, Inclusion-Exclusion)**.** Given a subset of features $T \subseteq [d']$ and integer r, let $\Upsilon(i)$ denote the set of all i-size combinations of features in T for

Primary Mechanism	RAP
Baseline Mechanisms	All-0, GM
Utility Measure	err_P
D	ADULT, LOANS
ϵ	$0.1, 1$
δ	$1/\lvert D\rvert^2$
$\lvert W\rvert$	$1, 4, 16, 64, 256$
n'	$500, 1000, 2000$
T	$1, 4, 16, 64$
K	$4, 16, 64, 256$
r	$1, 2, 3, 4$
k	4

Table 4.4: Experimental reference table for evaluating r-of-k thresholds with RAP.

$i = r, \ldots, k$; i.e., each $T_i \in \Upsilon(i)$ is such that $|T_i| = i$ and $T_i \subseteq T$. The polynomial threshold query $\hat{q}_{\hat{\phi}_{T,r}}$ parameterized by $\hat{\phi}_{T,r}$ can be defined in terms of product query predicates $\hat{\phi}_{T_i}$ as

$$\hat{\phi}_{T,r}(x) = \sum_{i=r}^{k} (-1)^{i-r} \binom{i-1}{i-r} \sum_{T_i \in \Upsilon(i)} \hat{\phi}_{T_i}(x).$$

Utilizing this redefinition of polynomial threshold queries reduces the number of arithmetic operations by nearly half relative to the original definition (when $r > k/2$, which we assume without loss of generality by simultaneously utilizing the first optimization in this section). In our subsequent experiments with r-of-4 thresholds (Section 4.4.3), this reduction in operations results in a maximal runtime improvement of approximately 40% for evaluating the polynomial threshold queries.

4.4.3 Evaluating RAP on r-of-k Thresholds

With the class of EEDQs derived, the primary question is how well RAP is able to utilize the EEDQs to answer prespecified sets of r-of-k thresholds. We investigate this question by evaluating how the various inputs to RAP affect its present utility and runtime.

4.4.3.1 Utility on r-of-k Thresholds

To begin, we evaluate the present utility of RAP on r-of-k thresholds, with k fixed at 4. As in our prior experiments in Section 4.3, we contextualize RAP's utility by comparing against the utilities of the All-0 and GM baseline mechanisms. We then evaluate the utility of each mechanism across a range of r values, ϵ values, datasets D, workload sizes $|W|$, and synthetic dataset sizes n', and across the same T, K values for RAP as before. Table 4.4 summarizes the precise parameter values.

Figure 4.7 displays the results of this experiment for $n' = 1000$, showing the minimal present error of RAP across all T, K values considered alongside the present error of the baseline mechanisms. The present errors of both baseline mechanisms are as expected, with the All-0 mechanism's present error having a clear and straightforward dependence on r, whereas the GM mechanism's present error is independent of r. Immediately, we see that RAP significantly outperforms the baseline mechanisms in all settings. Across the r values, we find that RAP achieves its minimal present error at $r = 4$ (i.e., 4-way marginals). Although RAP's present error for $r < 4$ is not much greater than for $r = 4$, we find no further apparent relationship between RAP's present error and r.

To understand the role that RAP's adaptivity plays in this experiment, in Figure 4.8, we visualize RAP's present error for each combination of T, K settings considered. Just as with 3-way marginals in Section 4.3.4.2, we find that the same adaptivity behavior emerges with 4-way marginals ($r = 4$); i.e., RAP primarily needs to evaluate a specific number of queries to achieve low present error, regardless of whether those queries are evaluated jointly in a small number of adaptive rounds or individually across a large number of adaptive rounds. However, we find that this behavior no longer holds for $r < 4$. Instead, the only consistent pattern that we find for $r < 4$ in this figure (which holds across other workload sizes and ϵ values as well) is that RAP achieves its minimal present error when the number of adaptive rounds is relatively large and the number of selected queries per round of adaptivity is relatively small. Since executing RAP for a large number of adaptive rounds is computationally expensive, this finding motivates

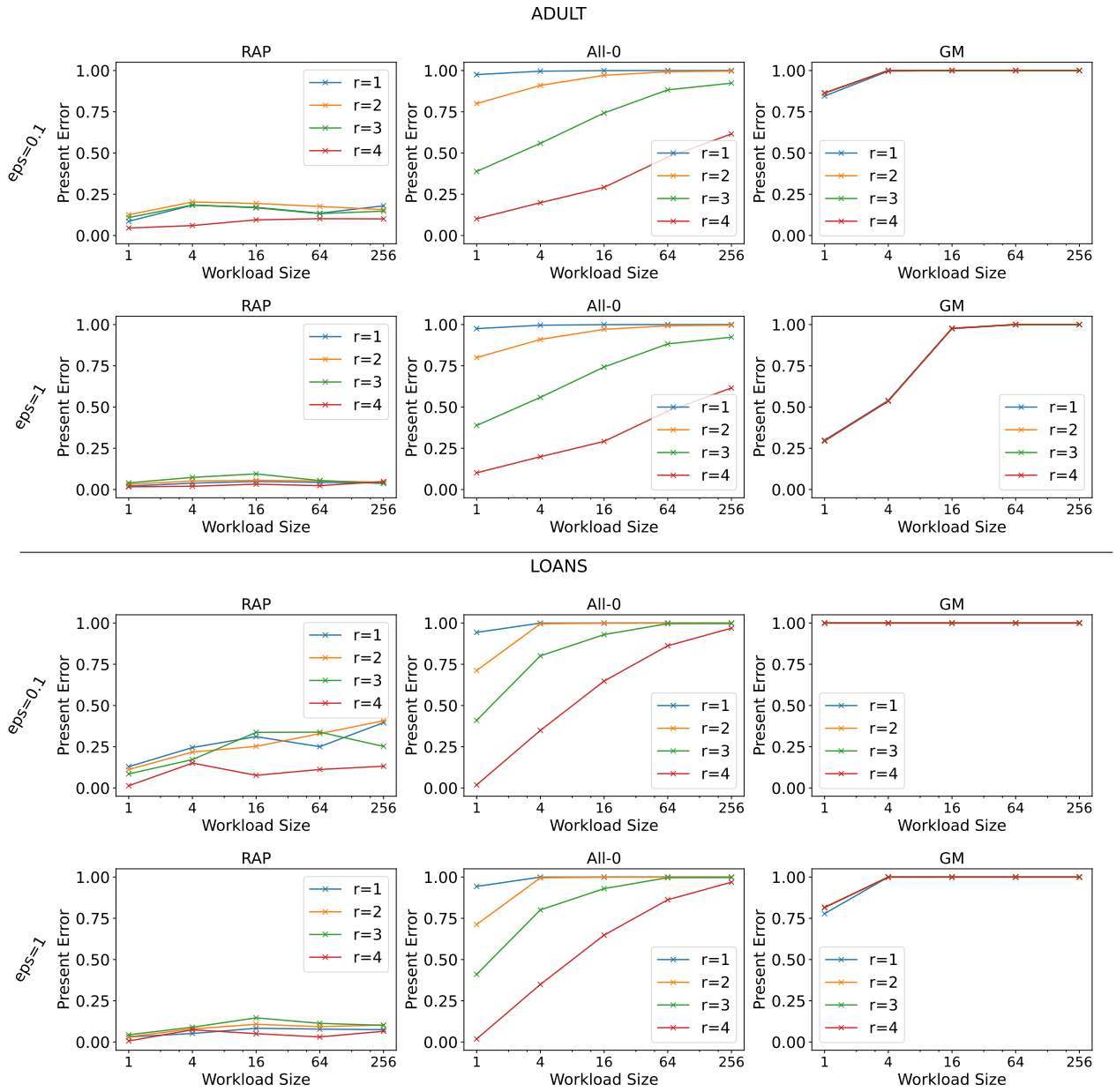


Figure 4.7: RAP's minimal present error across all T, K values considered alongside present error of the baseline mechanisms.

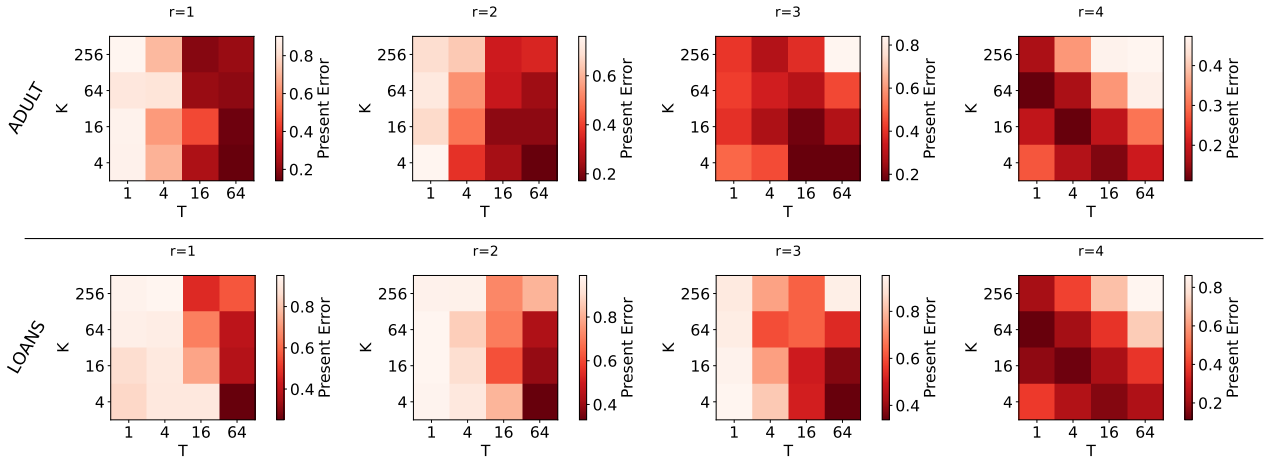


Figure 4.8: RAP's present error at each T, K value considered on a workload of 64 r-of-k thresholds with $\epsilon = 0.1$.

future work on reducing the necessary number of rounds of adaptivity. This could be done by more strategically selecting the set of queries in each round — for instance, by considering their expected joint impact on RAP's present error in the next optimization step, rather than selecting the individual queries with the highest present error independently.

4.4.3.2 Effect of Synthetic Dataset Size

Lastly, we investigate how RAP's synthetic dataset size n' affects its present error and runtime. Conceptually, n' controls RAP's learning capacity — the larger n', the better the answers to the queries should be. However, since optimizing large synthetic datasets is computationally expensive, n' cannot be taken arbitrarily large. Similarly, when the synthetic dataset size is too small, the optimization problem becomes underparameterized, which also results in a computationally expensive optimization process. Aydore et al. empirically confirmed this utility–computation trade-off for RAP with k-way marginals, where they found that setting $n' = 1000$ struck a good balance between utility and runtime for (filtered) 3-way and 5-way marginals.

We evaluate this trade-off on (unfiltered) r-of-4 thresholds, with the results shown in Figure 4.9. For each setting of r, we find that increasing n' generally results in a mild reduction

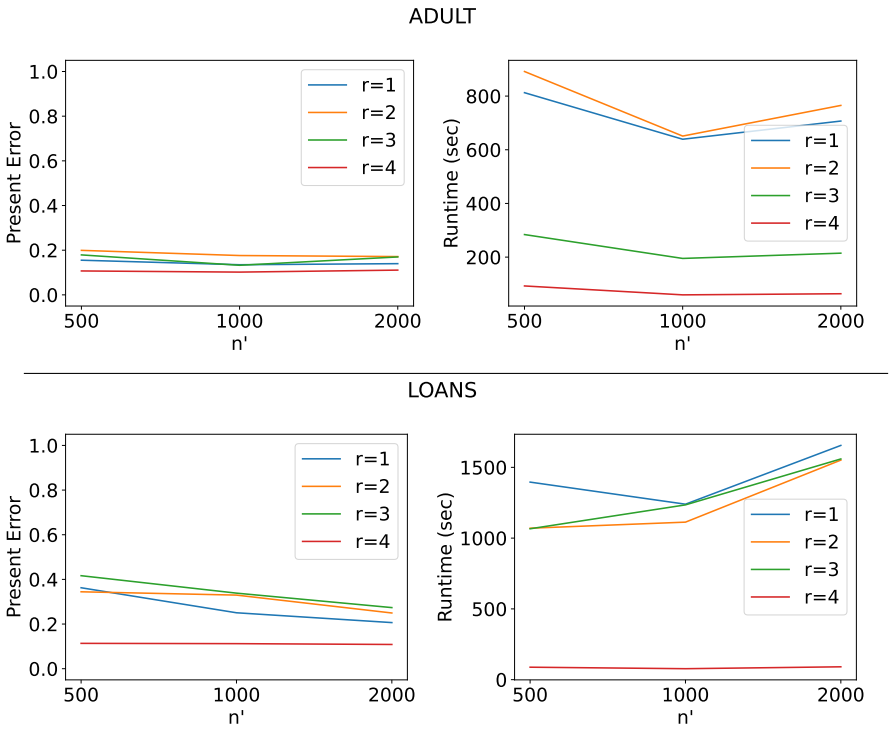


Figure 4.9: RAP's present error and runtime as a function of the synthetic dataset size on a workload of 64 r-of-k thresholds with $\epsilon = 0.1$.

of RAP's present error, but that at $n' = 1000$, RAP often attains minimal or near-minimal runtime. This mirrors Aydore et al.'s results and thus supports their findings regarding RAP's utility–computation trade-off. However, one interesting new finding is the effect that r has on RAP's runtime. A priori, we expected that RAP would have the shortest runtime when evaluating r-of-4 thresholds with $r \in \{1, 4\}$ and that their runtimes would be comparable. This is because at $r \in \{1, 4\}$, RAP has the least arithmetic operations to perform in order to evaluate each predicate (compared to $r \in \{2, 3\}$; refer to Section 4.4.2.2 for details on predicate evaluation). At $r = 4$, we confirm that RAP achieves minimal runtime. However, we find that $r = 1$ induces up to a 20x longer runtime. This increase in runtime is primarily explained by our prior observation that for $r < 4$, RAP achieves its maximal utility via a larger number of adaptive rounds (where RAP's runtime approximately linearly increases with the number of rounds).

However, even with this jump in runtime taken into consideration, we find that RAP is a highly performant mechanism for evaluating large sets of queries. For instance, consider the

worst-case runtime at $n' = 1000$ in Figure 4.9, which occurs where RAP answered a workload of 64 1-of-4 thresholds on the LOANS dataset. Here, RAP answered approximately 3.5×10^7 individual consistent queries in 1,240 seconds — a rate of over 28,000 queries per second. Based on these findings, we conclude that RAP is highly efficient for answering large sets of r-of-k thresholds.

4.5 Understanding RAP's Generalizability

In this final section, we propose a new and realistic intermediate setting that lies between the classic settings of having full knowledge of all queries in advance (i.e., the prespecified queries setting) vs. having no knowledge of which queries will be posed. We begin by concretely defining this new partial knowledge setting along with a generalization-based measure of utility for mechanisms operating within it. We then address our final contribution by empirically evaluating RAP's utility to determine its suitability in the new setting.

Motivation

In statistical modeling, and especially in the subfield of synthetic data generation, the primary goal is not to generate a model or a synthetic dataset that answers a prespecified set of queries well. Instead, the goal is to generate a model or synthetic dataset that *generalizes* well to future queries [Vap99; MRT12]. When it comes to differentially private mechanisms for answering statistical queries through a synthetic dataset, prior utility analyses have focused on either: (a) how well those mechanisms answer the prespecified set of queries, or (b) theoretically bounding how well the mechanisms can answer any class of queries in the worst case. For example, the utility of RAP (and the related practical mechanisms which preceded it) had previously been based solely on the answers to the prespecified workload, e.g., present utility. Experimentally evaluating a mechanism's present utility is straightforward: simply report the error of the highest error query from the prespecified query set. However, in some settings, it may be more beneficial to understand how well the mechanism can answer future queries. Towards this, theoretical

bounds can provide strong guarantees for the mechanism's worst-case utility across an entire query class [BLR08; Dwo+09; DRV10; HLM12; TUV12]. The drawback to using these theoretical bounds in practical settings is that they may be overly pessimistic, especially if the queries posed in the future are highly similar to those used to generate the synthetic dataset. This apparent disparity between the utility suggested by theoretical analyses and the actual utility that may be observed in practice is nearly identical to the disparity that famously exists between utility analyses in theoretical vs. empirical machine learning research [Vap98; BM06; SSBD14; NTS14; Zha+21]. However, for answering statistical queries with DP, the theoretical worst-case bounds are currently the best tool available without introducing additional information or assumptions.

4.5.1 Defining the Partial Knowledge Setting

We now motivate the design of a particular partial knowledge setting, then formally define it.

Much like in the machine learning research literature, we motivate a new partial knowledge setting for the context of differential privacy based on the rationale that in some realistic settings, future queries may be similar to queries posed in the past, i.e., historical queries. For instance, the U.S. Census Bureau periodically collects sensitive data for the decennial census and routinely allows researchers to securely pose queries directly on the collected data. Because similar data is being collected each decennial census, it is very likely that some of the queries analysts pose on one census dataset will be similar to those that analysts pose on the next census dataset. This intuition is illustrated in Figure 4.10.

We formalize this intuition on partial query repeatability for r-of-k thresholds in a general manner in Definition 4.5.1. For ease of exposition, we first introduce the following notation. Let $\mathcal{T}$ be an arbitrary distribution over thresholds, and let $Q \leftarrow \mathcal{T}$ denote the vector of all consistent queries Q of a threshold randomly drawn from distribution $\mathcal{T}$. Similarly, we let $Q \xleftarrow{|W|} \mathcal{T}$ denote the vector of all consistent queries Q from a $|W|$ size workload of thresholds sampled i.i.d. from $\mathcal{T}$.

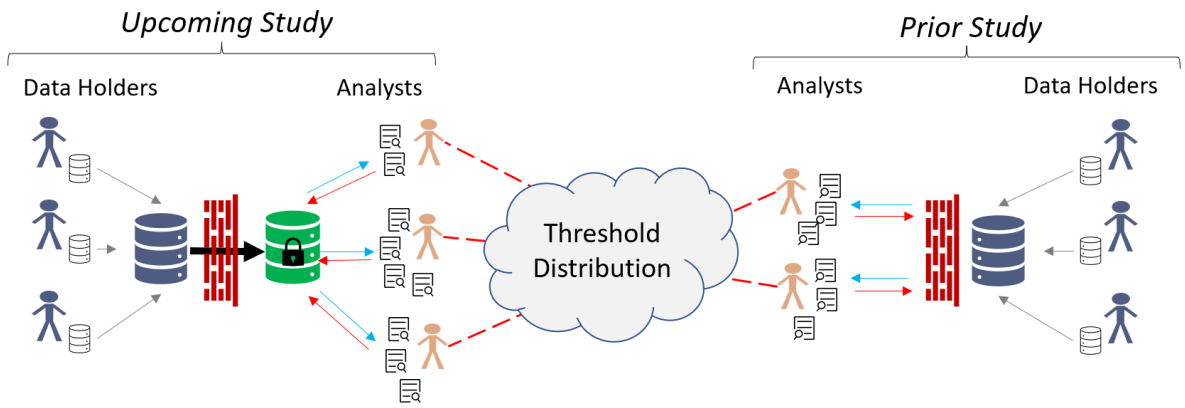


Figure 4.10: Visualization of the intuition behind how prior studies can provide partial knowledge of which future thresholds (or other query classes) may be posed by analysts.

Definition 4.5.1 (Partial Knowledge Setting, General)**.** Let $\mathcal{T}_H$ and $\mathcal{T}_F$ be arbitrarily related distributions over thresholds. In this setting, DP mechanisms are expected to answer arbitrary future thresholds drawn i.i.d. from $\mathcal{T}_F$. However, the DP mechanisms are not provided $\mathcal{T}_F$ explicitly. Instead, DP mechanisms are provided access to partial knowledge of $\mathcal{T}_F$ via a workload W_H of "historical" thresholds sampled i.i.d. from $\mathcal{T}_H$; i.e., the mechanisms are given access to $Q_H \xleftarrow{|W_H|} \mathcal{T}_H$.

Intuitively, in this partial knowledge setting, mechanisms can utilize Q_H to learn about the underlying threshold distribution $\mathcal{T}_H$, and if $\mathcal{T}_H$ is similar to $\mathcal{T}_F$, this will, in turn, inform what areas of the threshold space future thresholds are most likely to be sampled from. The role of Q_H in this setting is analogous to the role that training data plays in machine learning; i.e., it is the concrete sample of data provided to the mechanism that the mechanism can use to attempt to generalize.

For the historical queries Q_H to convey useful information about $\mathcal{T}_F$ to the DP mechanism, $\mathcal{T}_H$ and $\mathcal{T}_F$ should be related. Towards this, in Definition 4.5.2, we specify two concrete instantiations of the partial knowledge setting that make the relationship between $\mathcal{T}_H$ and $\mathcal{T}_F$ explicit.

1. Informally, the first concrete instantiation is the *exact* partial knowledge setting, where historical thresholds are drawn from the same distribution as future thresholds.

2. The second concrete instantiation is the *drifting* partial knowledge setting, which extends the exact partial knowledge setting. The drifting partial knowledge setting is inspired by the practical consideration that even if the historical and future thresholds distributions are initially the same, they may gradually drift apart over time.

In both settings, we ground the historical and future thresholds distributions in the observation that, in practice, certain features (or combinations of features) are likely to be more relevant to analysts than other features. For instance, in the ADULT dataset, "Age" and "Years of education" might be more relevant and valuable for analyses than "Capital loss amount" and "Relationship status". We model this relevance as a historical probability distribution $\mathcal{F}_H$ over the *features*, such that the probability mass corresponding to any r-of-k threshold in $\mathcal{T}_H$ corresponds to the (normalized) product of the k features' probabilities; i.e., $\mathcal{T}_H$ is the sampling distribution of k features from $\mathcal{F}_H$ without replacement. Our definition of the drifting partial knowledge setting specifically attempts to capture the practical phenomenon that if (for instance) analysts' interests are concentrated primarily in a small subset of features, then even if their interests drift over time, the analysts' new interests may still be concentrated in a small subset of different features. Based on this, we now formally define both concrete instantiations of the partial knowledge setting.

Definition 4.5.2 (Partial Knowledge Setting, Exact & Drifting)**.** Let $\mathcal{F}_H$ be an arbitrary historical distribution over features with $\mathcal{T}_H$ as its corresponding historical thresholds distribution. Without loss of generality, assume the features are sorted in descending order of their probability masses under $\mathcal{F}_H$; i.e., for each feature f_i with probability p_i, we have that $p_i \geq p_{i+1}$. Let $\gamma \in [0, 1]$ be a drift parameter, which defines the distributional similarity of the future distribution over features $\mathcal{F}_F$ (and correspondingly the future thresholds distribution $\mathcal{T}_F$) as follows. For each probability p_i, associate the corresponding key

$$k_i = \underbrace{(1-2\gamma)}_{\substack{\text{ordering}\\ \text{weighting}}} \cdot \underbrace{\frac{d-i}{d-1}}_{\substack{\text{relative order,}\\ \text{normalized}}} + \underbrace{(1-|1-2\gamma|)}_{\substack{\text{shuffling}\\ \text{weighting}}} \cdot \underbrace{u_i}_{\substack{\text{random}\\ \text{shuffling}\\ \text{amount}}},$$

where $u_i \overset{\text{iid}}{\sim} \text{Uniform}[0, 1]$. The feature distribution $\mathcal{F}_F$ is defined by leaving the features fixed in their original ordering, but reordering the probability masses in descending order of their keys. This results in a distribution of the same concentration but with probability masses re-assigned to potentially different features. Therefore, the future thresholds distribution $\mathcal{T}_F$ is the sampling distribution of k features without replacement from $\mathcal{F}_F$. When $\gamma = 0$, this procedure yields $\mathcal{T}_F = \mathcal{T}_H$, and we refer to this as the *exact* partial knowledge setting. When $\gamma > 0$, we refer to this as the *drifting* partial knowledge setting.

This model of drift is designed to maintain the concentration of the initial feature distribution $\mathcal{F}_H$ while interpolating between the exact partial knowledge setting ($\gamma = 0$) and a uniformly random reshuffling of the features' probabilities ($\gamma = 1/2$). For $0 < \gamma < 1/2$, this model induces a weighted amount of random reshuffling of probabilities in conjunction with simultaneously encouraging features' probabilities to remain "similar" to what they initially were, e.g., features with large probability masses under $\mathcal{F}_H$ are likely to retain large probability masses under $\mathcal{F}_F$. On the other end of the spectrum is the $\gamma > 1/2$ setting, where the relative orderings of probabilities become more likely to be reversed; e.g., features with large probability masses under $\mathcal{F}_H$ are likely to be assigned small probability masses under $\mathcal{F}_F$. At the extreme of this setting is $\gamma = 1$, which induces $\mathcal{F}_F$ of maximal total variation distance to $\mathcal{F}_H$ by deterministically reversing the relative ordering of the features' probabilities. Figures 4.11 and 4.12 concretely illustrate how the drift amount γ affects the distribution of future features.

We note that there are other well studied non-uniform probability measures on permutations that can be used in place of our proposed drift model (e.g., Mallows [Mal57], Plackett-Luce [Pla75; Luc12], etc.). We opted for our defined model because it is more straightforward while capturing the necessary components of threshold drift in an intuitive and easily controllable way. However, significant research has been conducted on various theoretical and applied aspects of alternative models. For a theoretical treatment of threshold drift, which we leave to future work, leveraging the significant body of results for these alternative models could prove invaluable.

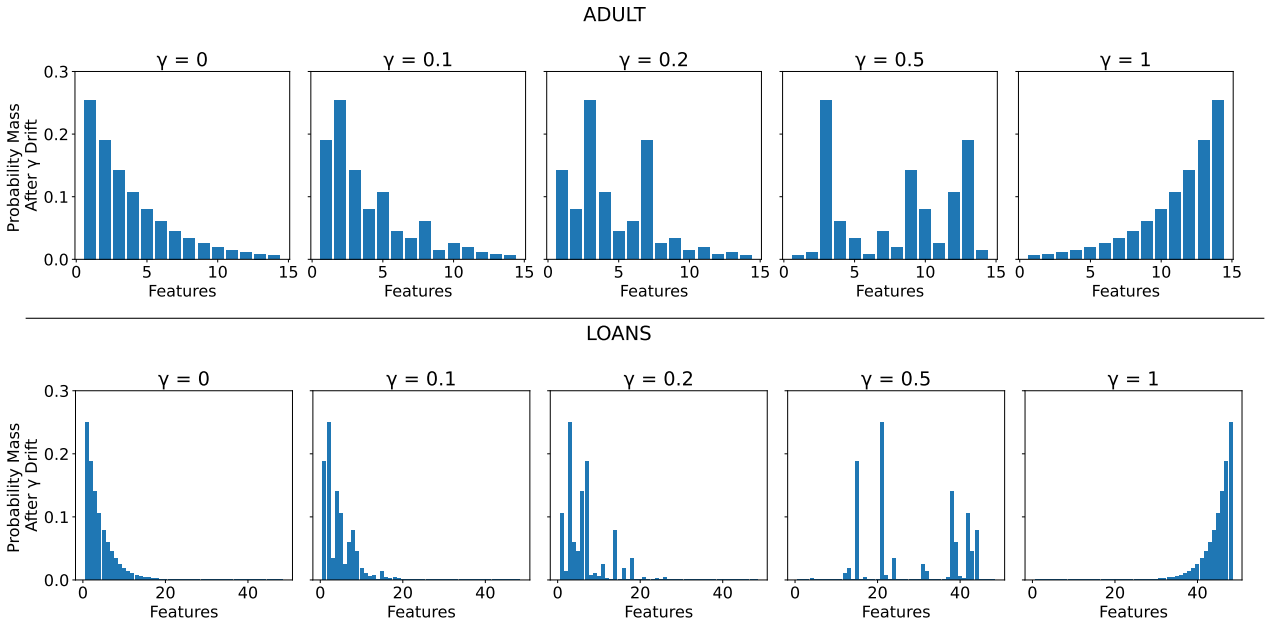


Figure 4.11: Examples of drifted feature distributions $\mathcal{F}_F$ across a range of drift parameters γ, with an initial Geometric distribution for $\mathcal{F}_H$ on the ADULT and LOANS datasets. Categorical features are numbered (rather than named) along the x-axis.

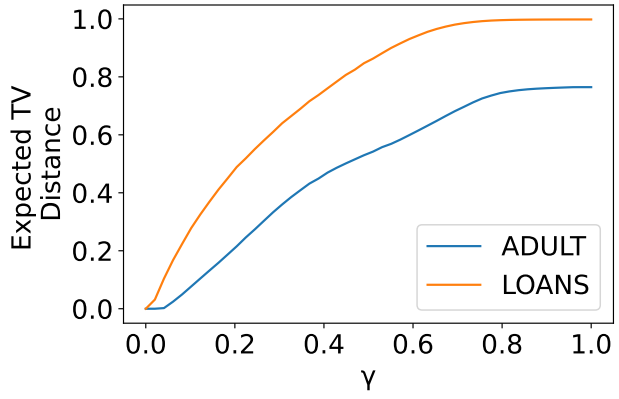


Figure 4.12: Effect of drift parameter γ on the total variation distance between the historical features distribution $\mathcal{F}_H$ and the future features distribution $\mathcal{F}_F$, with an initial Geometric distribution for $\mathcal{F}_H$ on the ADULT and LOANS datasets.

4.5.2 Measuring and Computing Utility

Having concretely defined the partial knowledge setting, we formally define a utility measure to quantify how well a mechanism can answer future thresholds based on the historical thresholds it was given access to. In other words, we define a measure quantifying how well the mechanism generalizes. We then describe how to empirically evaluate this defined utility measure efficiently.

In this setting, we are interested in the mechanism's error across its answers to the consistent queries of r-of-k thresholds drawn from $\mathcal{T}_F$. This new utility measure is based on the classic utility measure used in the prespecified queries setting (Definition 4.3.1), with the only difference being that the randomness of the future thresholds distribution $\mathcal{T}_F$ is now explicitly taken into account. We thus define *future utility*, which we measure in terms of the negative of *future error*; i.e., a mechanism with low future error has high future utility and vice versa. Specifically, future error is the expected absolute error taken over the randomness of both M and $\mathcal{T}_F$, formally defined as follows.

Definition 4.5.3 (Future error)**.** Let $a = Q(D)$ be the true answers to all queries in Q on D, and let $\tilde{a}$ be mechanism M's corresponding answers. Then err_F is the future error of mechanism M, defined as $\mathrm{err}_F(M, D, \mathcal{T}_F) = \mathbb{E}_{M(D), Q \leftarrow \mathcal{T}_F} \|a - \tilde{a}\|_\infty$, where the expectation is over the randomness of both the mechanism and future threshold distribution.

Theoretically evaluating err_F of a mechanism on *a priori* unknown threshold distributions without resorting to worst-case bounds is a challenging problem. Experimentally, however, we are able to efficiently and accurately estimate err_F for the `RAP` mechanism as follows:

1. Construct the feature distributions $\mathcal{F}_H$ and $\mathcal{F}_F$ according to real-world phenomena, which in turn define the threshold distributions $\mathcal{T}_H$ and $\mathcal{T}_F$.

2. Generate a workload W_H of historical thresholds, yielding query vector $Q_H \xleftarrow{W_H} \mathcal{T}_H$. Independently, generate a workload W_F of future thresholds, yielding query vector $Q_F \xleftarrow{W_F} \mathcal{T}_F$.

3. Provide Q_H as the input queries to RAP in order to generate a synthetic dataset.

4. Use the synthetic dataset to answer Q_F, recording the mean error (and optionally, the corresponding confidence intervals to quantify how faithfully err_F was approximated).

This evaluation approach is analogous to standard practice in empirical machine learning research where data is split into "training" and "test" sets randomly (to ensure distributional similarity) [Has+09]. The model is then learned on the training set and subsequently evaluated on the test set to measure how well it generalizes.

4.5.3 Evaluating RAP's Future Utility

As our final contribution, we empirically evaluate RAP's future utility for answering r-of-k thresholds. The experiments that we perform on RAP to understand its suitability in this new partial knowledge setting are as follows:

- Evaluating the effects that the threshold distribution concentration and the historical threshold workload size $|W_H|$ have on RAP's future utility.
- Evaluating the effect that "overfitting" in the synthetic data optimization step has on RAP's future utility.
- Evaluating the effect that the distribution drift amount γ has on RAP's future utility.

These experiments are designed to assess the distinct new ways (beyond those in the previous prespecified queries setting) in which RAP's inputs may influence its future utility.

4.5.3.1 Effect of Threshold Distribution Concentration & Historical Workload Size

To empirically evaluate RAP's future utility in the exact partial knowledge setting, we must specify the particular threshold distribution $\mathcal{T}_H = \mathcal{T}_F$ from which we generate both the input queries

Q_H and future queries Q_F used to evaluate err_F. As previously discussed, we do so by specifying feature distributions $\mathcal{F}_H$ and $\mathcal{F}_F$ that, in turn, define the threshold distributions. As a baseline, we choose what is intuitively the most challenging extreme: setting $\mathcal{F}_H$ and $\mathcal{F}_F$ to be the Uniform distribution. We expect the future utility of this baseline to be the lowest among all possible distributions since it is the least concentrated, implying that it provides the least amount of information possible to the mechanism about any particular region of the threshold space.

In an effort to model the real-world phenomenon that certain features are likely to be more relevant to analysts than other features, we utilize the following two feature distributions. For a highly concentrated distribution, we use the exponentially-tailed Geometric distribution. For a mildly concentrated distribution, we use the heavy-tailed Zipfian distribution. Both distributions are commonly used in practice when modeling real-world phenomena, e.g., [MC89; Yu+04; Zen+12; OVL18]. We hypothesize that the highly concentrated Geometric distribution will induce high-utility results since many of the same features in Q_H will also appear in Q_F. Analogously, we hypothesize that the mildly concentrated Zipfian distribution will induce lower-utility results (although still higher than the Uniform distribution baseline).

With a fixed threshold distribution $\mathcal{T}_H$ defined by the feature distribution $\mathcal{F}_H$, we must specify how many thresholds will be randomly sampled to form the historical threshold workload W_H (and corresponding vector of all consistent queries Q_H) that RAP takes as input. Obtaining a clear understanding of what impact the historical workload size $|W_H|$ has on RAP's future utility is important because there may be a tension between the number of historical r-of-k thresholds and RAP's future utility. On the one hand, the more sampled thresholds there are, the more information RAP has about the underlying distribution $\mathcal{T}_F$ from which future thresholds will be generated. This suggests that the more historical r-of-k thresholds there are, the higher RAP's future utility should be. On the other hand, to optimize RAP's underlying synthetic dataset, its privacy budget is split between all queries consistent with the historical thresholds. This implies that the more historical thresholds there are, the more noise will be added to each consistent

Primary Mechanism	RAP
Baseline Mechanism	All-0
Utility Measure	err_F
D	ADULT, LOANS
ϵ	0.1
δ	$1/\lvert D\rvert^2$
$\lvert W_H\rvert$	$1, 4, 16, 64, 256, 1024$
n'	1000
T	$1, 4, 16, 64$
K	$4, 16, 64, 256$
r	1
k	3
$\mathcal{T}_H, \mathcal{T}_F$	Uniform, Zipf, Geometric
γ	$0, 0.05, 0.1, 0.2, 0.5, 1$

Table 4.5: Experimental reference table for evaluating the future utility of RAP on r-of-k thresholds.

query's answer, which seems to suggest that this will cause the future utility to be lower. Thus, we seek to understand whether one of these two possibilities is correct or whether there is a "sweet spot" where a certain number of historical thresholds is just enough for the mechanism to implicitly learn $\mathcal{T}_F$ but does not result in the privacy budget being spread too thin.

To empirically quantify the effect of both the threshold distribution concentration and historical workload size on RAP's future utility, we evaluate RAP across a range of workload sizes using the three specified distributions over features in both the ADULT and LOANS datasets. To put RAP's future utility into context, we also evaluate the future utility of the All-0 baseline mechanism. Refer to Table 4.5 for a summary of this experiment.

Figure 4.13 shows the results of this experiment. As in our prior experiments, each point of the RAP line is taken to be where RAP achieves minimal *present error* across all combinations of T, K evaluated. The future error at this minimizing T, K pair is then evaluated and plotted, along with a corresponding 95% confidence interval to account for randomness both between independent repetitions and across sampling future thresholds from the threshold distribution. For real-world applications, this reflects what a practitioner using RAP would be able to do; i.e., choose the best-performing instance of RAP across T, K values on the present error metric (since they would not

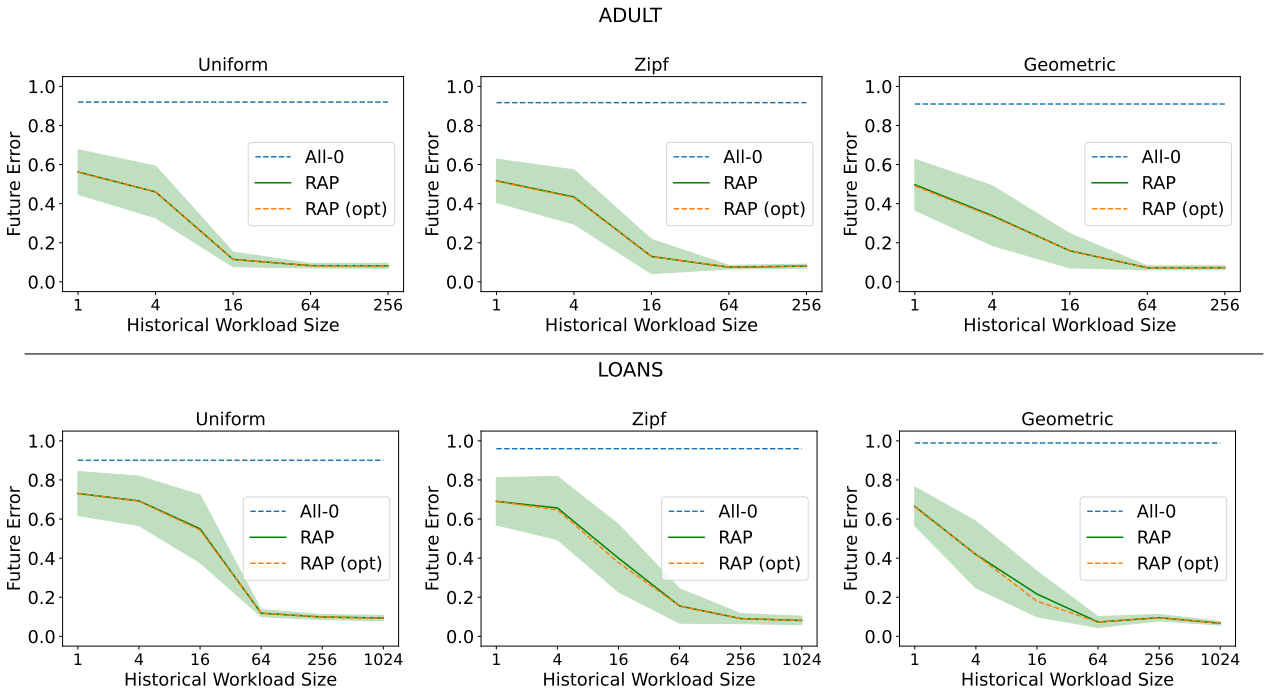


Figure 4.13: RAP's future error (and 95% confidence intervals) across all T, K values considered where RAP achieves minimal present error, plotted across a range of workload sizes and historical threshold distributions. "RAP (opt)" represents RAP's future error across all T, K values considered where RAP achieves minimal future error. Future error of All-0 included as a baseline.

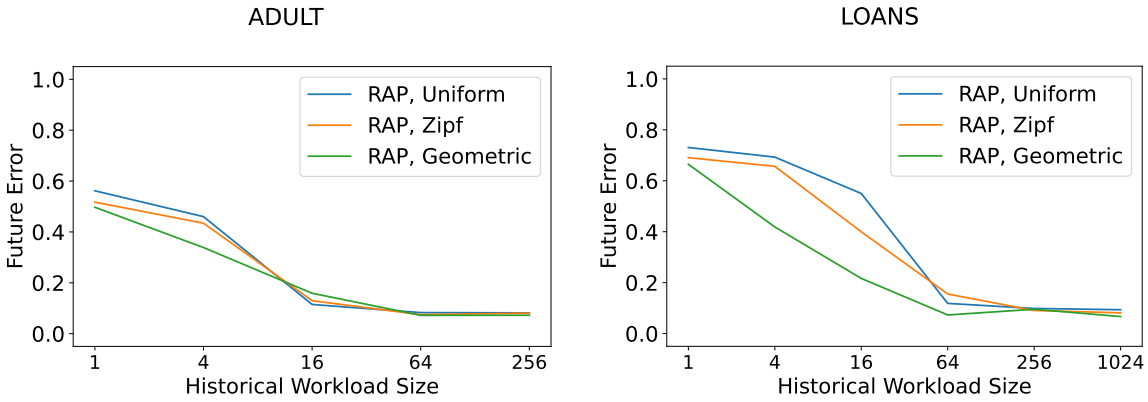


Figure 4.14: RAP's future utility on each threshold distribution across a range of workload sizes.

be able to evaluate future error), and then use that instance to answer future queries. Ideally, however, the practitioner would have omnisciently been able to choose the best-performing instance of RAP across T, K values on the future error metric directly, as this approach will never have larger future error than the former (feasible) approach. To understand whether there is a significant difference in the future error between these two scenarios, we additionally plot the latter as "RAP (opt)". For each distribution individually, we find the results are as expected. Namely, RAP's future error is always lower than the baseline mechanism All-0's future error, and RAP's future error decreases as the number of historical thresholds that it is given increases. Interestingly, we find no evidence that there is any point at which the number of historical thresholds given to RAP becomes "too large" and causes RAP's future error to begin increasing. Instead, we find that RAP benefits from being provided more historical thresholds when the historical workload size is small, then eventually reaches a point of diminishing returns. Additionally, we find that the future error corresponding to the RAP instance that attains minimal present error across T, K values is nearly identical to the future error corresponding to the RAP instance that attains minimal future error across T, K values. This indicates that in practice, answering future queries using the RAP instance that achieved minimal present error across T, K values will likely also yield the minimal future error.

To better visualize the differences across distributions, RAP's future error lines are overlayed in Figure 4.14 for both the ADULT and LOANS datasets. From this, we see that the differences between RAP's future error across all three distributions are not as striking as one may expect. For small historical workload sizes (less than 16 and 64 on the ADULT and LOANS datasets, respectively), we find that the results roughly align with our intuition: the least concentrated (Uniform) distribution induces the highest future error, while the most concentrated (Geometric) distribution induces the lowest future error. These findings, taken together with those of Figure 4.13, yield a simple, useful insight into how to achieve low future error with RAP in practice. Specifically, if the size of the historical workload is small, a practitioner can simply augment it by adding

uniformly randomly sampled thresholds from the space of all possible thresholds (regardless of what the underlying threshold distribution $\mathcal{T}_H$ is). In the worst case, RAP's future error will be essentially unaffected (if $|W_H|$ was already in the region where returns are diminishing); in the best case, RAP's future error will be reduced significantly.

4.5.3.2 Effect of "Overfitting" the Synthetic Dataset

When answering a prespecified set of queries using RAP, the goal in the relaxed projection step is to achieve as close to a global minimum as possible. In fact, although such an achievement is unlikely in practice, Aydore et al.'s theoretical utility result relies on a global minimum having been reached. However, when the goal is to learn a model that generalizes to unseen data, it is well known that optimizing the loss function to a global minimum will lead to an extremely overfit model. In the exact partial knowledge setting where future utility is the metric of choice, we seek to determine whether a conceptually similar "overfitting" phenomenon may be occurring when RAP uses the historical threshold workload to generalize to future thresholds.

Towards this, we recall our finding from Figure 4.13. Specifically, that RAP does not seem to noticeably overfit to the historical queries when selecting the adaptivity parameters T and K based on the instance of RAP that had minimal present error. However, this finding does not eliminate the possibility that RAP is overfitting to the historical queries during the synthetic dataset optimization procedure itself. For instance, in Figure 4.14 on the LOANS dataset at a historical workload size of 4, there is a significant difference between RAP's future errors on the Uniform vs. Geometric distributions. This could be explained either by RAP overfitting to the historical workload generated from the Uniform distribution (which is relatively less informative regarding which thresholds are likely to be sampled in the future), or it could simply indicate that the historical workload does not contain enough information about the relevant space of thresholds that RAP needs to generalize well. To analyze this possibility, we perform the same experiment as above while simultaneously evaluating RAP's future utility not just at the end of the

optimization procedure but after each iteration. Figure 4.15 displays the results, along with RAP's training loss and present error after each iteration of the optimization procedure. In classic ML, a canonical symptom of overfitting is observing a point in the training progress where the training error continues decreasing but where the test error begins steadily increasing. In our setting, the analog would be observing a point where the present error continues decreasing but where the future error begins increasing. However, we do not observe such behavior in either graph, as the future error steadily decreases throughout the entire training procedure. The primary difference between the two graphs is that RAP's decrease in future error under the Uniform distribution is much smaller than under the Geometric distribution. This indicates that, as expected, RAP is able to take advantage of the significantly more informative (with respect to the relevant portions of the space that future thresholds will be drawn from) historical workload from the Geometric distribution. Viewed differently, in the case of the Uniform distribution, RAP did not "overfit" to the historical workload — rather, the historical workload did not provide enough information to RAP about the relevant remainder of the query space.

The takeaway from these findings is that while RAP would have benefited from having a larger historical threshold workload, it would not have benefited from introducing analogs to other classic overfitting remedies. For example, a practitioner may be tempted to reserve a held-out set of thresholds from the historical workload with the intention of using them between iterations as a proxy to estimate future utility, stopping the training early when the error on the held-out set begins increasing. Not only do these findings indicate that such a strategy would not be beneficial, but combined with the findings from the previous experiment, we conclude that such a strategy would result in relatively *greater* future error due to the reduced historical workload that RAP is given.

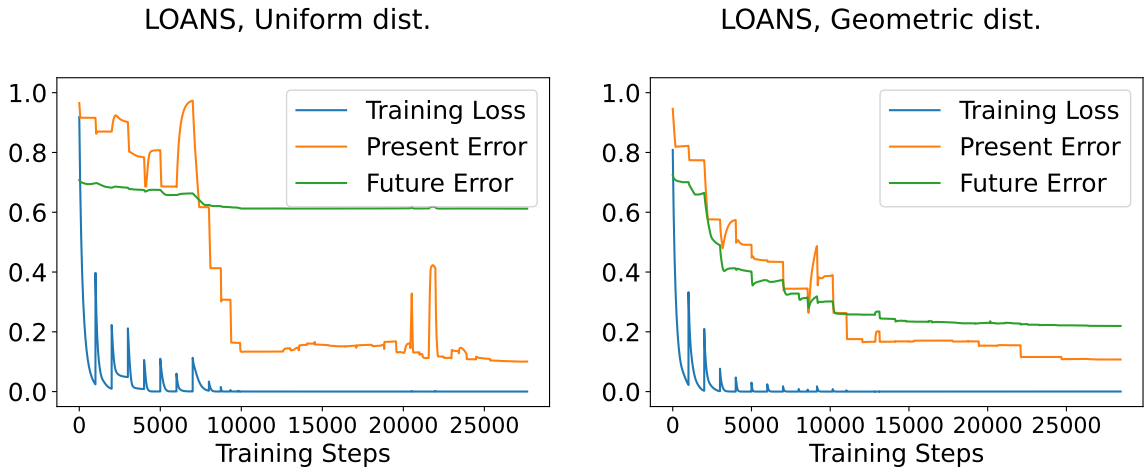


Figure 4.15: Training progress across iterations for RAP on Uniform vs. Geometric distributions over features in LOANS dataset, both with a small historical workload size of 4.

4.5.3.3 Effect of Threshold Distribution Drift

In the drifting partial knowledge setting, as the future features distribution $\mathcal{F}_F$ drifts further from the historical features distribution $\mathcal{F}_H$, it is clear that RAP's future utility should decrease. However, it is unclear how *sensitive* RAP's future utility is to such drift. Thus, we seek to quantify the extent to which RAP can tolerate distributional drift while maintaining high future utility.

To achieve this, we evaluate RAP's future utility in the following experiment. We first define the historical features distribution $\mathcal{F}_H$ using the aforementioned highly concentrated Geometric distribution over features in both the ADULT and LOANS datasets. We then measure RAP's future error across a range of drift amounts. Because RAP achieved low future error in the exact partial knowledge setting on all distributions when the workload size was large enough, we anticipate that distributional drift will similarly not have a significant impact when the historical workload size is large. Thus, in Figure 4.16, we evaluate the impact of distributional drift specifically with small historical workload sizes of 4 and 16 on the ADULT and LOANS datasets, respectively.

The results of this experiment reveal that on both datasets, RAP is fairly impervious to distributional drift. RAP's future error only begins to exhibit a significant increase at approximately $\gamma = 0.4$ on the ADULT dataset and $\gamma = 0.1$ on the LOANS dataset. Compared with Figure 4.12,

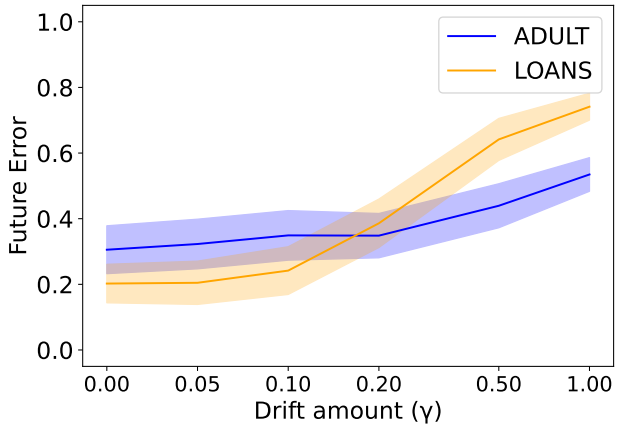


Figure 4.16: Future error of RAP across a range of distributional drift amounts on the ADULT and LOANS datasets, given small historical workload sizes of 4 and 16, respectively.

these points correspond to an expected total variation distance between the historical and future features distributions of approximately 0.5 on their respective datasets. Thus, we are able to conclude that even if the future features distribution drifts from the historical features distribution by a moderate amount, RAP can still be expected to maintain high utility.

4.6 Additional Related Works

In this section, in addition to the prior works on large-scale query answering previously discussed (Section 4.1.1), we detail other important works related to differentially private query answering. We begin by discussing some works (concurrent with and subsequent to our work presented in this chapter) related to answering large sets of prespecified queries. For the mechanisms defined in these works, a prime direction for future research would be to evaluate them analogously to our evaluation of RAP in this chapter. For instance, evaluating their scalability to larger query spaces and their generalizability for answering queries posed in the future, perhaps in a manner similar to Tao et al. [Tao+21]). We then briefly discuss some lines of research related to the general problem and settings explored in this chapter.

Answering Many Queries

One closely related work to the goals of this chapter is that of Liu et al. [LVW21], which studies the problem of constructing an algorithmic framework for privately answering a prespecified set of statistical queries — our first setting of interest. Concretely, their framework unifies several DP mechanisms that specifically answer queries by building a synthetic dataset through iterative, adaptive updates. These mechanisms include the previous practical state-of-the-art mechanisms [Gab+14; Vie+20], as well as a modified variant of a *preliminary* version of the RAP mechanism (where a softmax transformation [Bri90] is applied to each row of the synthetic dataset D after each iteration of RAP's optimization procedure). Liu et al. then leverage their framework to design two new mechanisms for answering prespecified sets of queries and empirically show that both achieve high utility. However, in their empirical evaluations, Liu et al. find that the modified RAP mechanism's utility is comparable to the utility of their two newly proposed mechanisms and that RAP is computationally cheaper to execute. Thus, we leave large-scale evaluations of their two new mechanisms as future work. Moreover, Aydore et al. have subsequently updated the RAP mechanism to incorporate a similar modification (applying the Sparsemax transformation [MA16], and optionally finishing with randomized rounding) and showed that it further improves utility — in turn, further justifying our focus on the RAP mechanism.

Along similar lines, another closely related work is the recently introduced *Adaptive and Iterative Mechanism* (AIM) by McKenna et al. [McK+22]. AIM is a mechanism for DP synthetic data generation to specifically answer workloads of marginal queries. The high-level idea of their approach is similar to that of RAP and Liu et al.'s work [LVW21], adaptively selecting marginals to use to optimize the synthetic dataset. However, their work takes this further by designing a method to perform the selection more intelligently. Moreover, they develop new techniques to quantify the uncertainty of answers derived from the generated synthetic data. Empirically evaluating AIM, they show that it generally outperforms prior state-of-the-art mechanisms, including RAP. However, their evaluation setting was somewhat different; specifically, they reduced

the domain size of the datasets by discretizing numerical features into 32 equal-width bins. This makes the optimization problem significantly easier for all mechanisms they evaluate, which is highly useful when running a wide range of experiments across many random trials. However, it leaves AIM's utility unclear when the data is unbinned and sparse (e.g., for a numerical attribute with 100 possible values). Moreover, since the source code of AIM's implementation was not released, we consider a ground-up reimplementation of AIM amenable to large-scale evaluations on large and sparse data spaces to be out of the scope of this work. Performing such evaluations, especially in connection to the computational resources required by each method (AIM, `RAP`, and others), is a prime direction for future work.

Another closely related work is the concurrent theoretical work of Nikolov [Nik22], which proposes and analyzes a new mechanism for answering sets of prespecified queries with differential privacy. Their new mechanism is based on randomly projecting the queries to a lower-dimensional space, answering the projected queries with a simple DP additive-noise mechanism, then lifting the answers back into their original dimension. Their work's primary focus and contribution is the thorough mathematical analysis of the mechanism's utility, showing that it achieves optimal worst-case sample complexity under an average error metric. Such results are less directly relevant to our work, as we focus on different error metrics for fixed real-world datasets (rather than in the worst case across all possible datasets). However, conceptually, Nikolov's newly proposed mechanism could be used to tackle the same problem as our work. Practically though, the runtime of Nikolov's mechanism (although polynomial) would prevent it from being used to answer the large number of queries that we answer with `RAP` in this chapter. An intriguing direction for future work would be adapting Nikolov's new mechanism for practical query answering and determining ways to scale it up to accurately answer queries on a truly large scale.

A final line of closely related work is the subsequent work of Vietri et al. [Vie+22]. Their work focuses explicitly on enhancing the `RAP` mechanism, creating a new mechanism they call

RAP++. Their goal is orthogonal to the goal of this chapter in that they seek to extend the original RAP mechanism so that it can support numerical features natively. Prior to their work, RAP required one-hot discretization of any numerical features in the dataset. For features with wide numerical ranges, one-hot discretization greatly increases the dimensionality of RAP's optimization problem, increasing the computational burden and simultaneously decreasing the mechanism's overall utility. In RAP++, Vietri et al. incorporate tempered sigmoid annealing and random linear projection queries into RAP in order to handle a mixture of categorical and numerical features without any discretization. They perform several empirical evaluations on RAP++, finding that it achieves state-of-the-art utility and runtime. Despite their goal being orthogonal to this chapter's goal, the findings could be used to further improve the RAP++ mechanism and its evaluation.

Related Lines of Research

One related (but disjoint) line of research is on the *public/private* model of differential privacy, where some data must be protected with differential privacy while the remaining "public" data requires no privacy protections [BNS13; JE13; HCB16; Pap+17; Bas+20a; ABM19; Liu+21; TBM21]. These works have shown that mechanisms can be designed which make use of a small amount of public data in order to boost utility significantly. Our work differs from this model in that it does not use any public data. In our newly defined partial knowledge setting, we instead assume that the entire set of user data D is private but that there exist publicly known historically posed queries Q_H which are not privacy sensitive. Assuming that Q_H was generated from a random distribution $\mathcal{T}_H$, we seek to understand the extent to which the RAP mechanism is able to take advantage of Q_H using D to accurately answer future queries generated from a distribution $\mathcal{T}_F$ related to $\mathcal{T}_H$.

The final related line of work is on reconstruction attacks, which studies how accurately sets of queries can be answered before private information in the dataset can be recovered. The high-level results of this research can be summarized through the Fundamental Law of Information

Recovery [DR+14]: "overly accurate answers to too many questions will destroy privacy in a spectacular way." Initial work on reconstruction attacks [DN03] inspired the conception of DP, and subsequent works have improved the computational efficiency of attacks, improved the theoretical analyses of attacks, or crafted highly effective attacks to specific cases [DMT07; DY08; MN12; Dwo+17; GAM19]. Although somewhat related, this line of work's focus significantly differs from our work's focus. In research on reconstruction attacks, the basic goal is to find worst-case sets of queries (or the minimal sizes thereof) such that it is impossible to answer them all accurately while simultaneously maintaining privacy. In this work, our focus is not on generic worst-case queries but instead on efficiently and accurately answering practical sets of prespecified or randomly sampled queries with privacy. Thus, the works on reconstruction attacks are not directly relevant to our problem in either of the two settings we consider.

4.7 Future Directions

As evidenced by the discussion of related works in Section 4.6, there is significant current research progress on DP query workload answering and DP synthetic data generation. Specifically related to the goals of this chapter, we believe the most promising future directions lie in the *scalability* and *generalizability* of query workload answering mechanisms.

Regarding scalability, practical mechanisms for answering query workloads have been evaluated across a range of datasets and workloads to quantify their utility. However, less attention has been devoted to quantifying these mechanisms' scalability and understanding when and why they may fail to scale. Some DP mechanisms may excel at accurately answering a small query workload, but whose runtime becomes intractable as the workload grows. Others may excel at answering moderately large workloads quickly and accurately, but their implementations suddenly fail due to resource constraints when faced with truly massive workloads. Conversely, some mechanisms may answer exceedingly large workloads quickly by exploiting a simpler design or natural parallelization, but their accuracy on smaller workloads is relatively poor compared

to the other approaches that are unable to scale. Thus, we posit that a critical direction for future work lies in benchmarking DP query answering mechanisms while controlling for important computational resources. For instance, DP query answering mechanisms should be benchmarked not only across a range of workload sizes but also across a range of RAM allowances (or VRAM allowances for GPU-bound mechanism implementations), runtime limits, and distributed computing resources. In particular, designing a system to perform such benchmarks in an automated way would remove a tremendous barrier for anyone looking to leverage a DP query answering mechanism for their particular use case.

Regarding generalizability, a straightforward direction for future work is understanding how well alternative DP query answering mechanisms are able to generalize for answering queries posed in the future. A separate direction, however, is in designing DP query answering mechanisms that generalize in a different sense. In this chapter and related prior works, mechanisms were evaluated on single specific subclasses of statistical queries, e.g., k-way marginals. But, in practice, when a synthetic dataset is released, analysts may desire to pose queries from classes other than the class the synthetic dataset was explicitly generated from. As a simple example, an analyst may want to pose both k-way marginal queries and 1-of-k queries. Designing DP mechanisms capable of producing useful synthetic datasets for answering these *mixed-class* queries is a particularly valuable line of future research with important practical implications.

4.A Chapter Appendix

Deferred Regression Analysis Details

In this portion, we present the details of the setup and results for the regression analysis on the utility impact of filtering "large" marginals out of RAP's evaluation.

Present Error vs. Workload Size

For this regression analysis on each dataset, we define the following regression variables:

- x_1, x_2: dummy variable encodings for the three levels of ϵ evaluated. I.e.,
  - $x_1 = x_2 = 0$ represents $\epsilon = 0.01$.
  - $x_1 = 1, x_2 = 0$ represents $\epsilon = 0.1$.
  - $x_1 = 0, x_2 = 1$ represents $\epsilon = 1$.
- x_3: logarithm of the workload size.
- x_4: indicator variable representing whether thresholding was applied. I.e., $x_4 = 0$ if thresholding was not applied, $x_4 = 1$ if it was.
- ζ: stochasticity in the process (e.g., from randomness in the RAP mechanism due to privacy, from randomness in the marginal selection process across independent trials, etc.).

With these variables defined, we state the full regression model with interactions as

$$\text{err}_P = \beta_0+\beta_1 x_1+\beta_2 x_2+(\beta_3+\beta_4 x_1+\beta_5 x_2)x_3+(\beta_6+\beta_7 x_1+\beta_8 x_2+(\beta_9+\beta_{10} x_1+\beta_{11} x_2)x_3)x_4+\zeta,$$

and the restricted regression model as

$$\text{err}_P = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + (\beta_3 + \beta_4 x_1 + \beta_5 x_2)x_3 + \zeta.$$

We then fit both the full and restricted regression models to the results of the RAP evaluations for the ADULT and LOANS datasets (separately). Regression results for the full models (ADULT on left and LOANS on right) are stated below.

Dep. Variable:	present_err	**R-squared:**	0.963
Model:	OLS	**Adj. R-squared:**	0.959
Method:	Least Squares	**F-statistic:**	266.6
Covariance Type:	nonrobust	**Prob (F-statistic):**	7.40e-76
No. Observations:	126	**Log-Likelihood:**	295.45
Df Residuals:	114	**AIC:**	-566.9
Df Model:	11	**BIC:**	-532.9

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
β_0	0.0320	0.009	3.415	0.001	0.013	0.051
β_1	-0.0066	0.013	-0.495	0.621	-0.033	0.020
β_2	-0.0248	0.013	-1.869	0.064	-0.051	0.001
β_3	0.0650	0.003	24.536	0.000	0.060	0.070
β_4	-0.0528	0.004	-14.075	0.000	-0.060	-0.045
β_5	-0.0600	0.004	-16.015	0.000	-0.067	-0.053
β_6	0.0280	0.013	2.120	0.036	0.002	0.054
β_7	-0.0309	0.019	-1.649	0.102	-0.068	0.006
β_8	-0.0277	0.019	-1.482	0.141	-0.065	0.009
β_9	-0.0036	0.004	-0.952	0.343	-0.011	0.004
β_{10}	0.0052	0.005	0.988	0.325	-0.005	0.016
β_{11}	0.0040	0.005	0.762	0.448	-0.006	0.014

Omnibus:	24.270	**Durbin-Watson:**	1.693
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	114.122
Skew:	0.434	**Prob(JB):**	1.65e-25
Kurtosis:	7.581	**Cond. No.**	64.4

Dep. Variable:	present_err	**R-squared:**	0.942
Model:	OLS	**Adj. R-squared:**	0.937
Method:	Least Squares	**F-statistic:**	193.4
Covariance Type:	nonrobust	**Prob (F-statistic):**	1.05e-75
No. Observations:	144	**Log-Likelihood:**	228.17
Df Residuals:	132	**AIC:**	-432.3
Df Model:	11	**BIC:**	-396.7

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
β_0	0.0372	0.019	1.982	0.050	7.32e-05	0.074
β_1	-0.0113	0.027	-0.425	0.671	-0.064	0.041
β_2	-0.0282	0.027	-1.062	0.290	-0.081	0.024
β_3	0.0966	0.004	21.626	0.000	0.088	0.105
β_4	-0.0767	0.006	-12.134	0.000	-0.089	-0.064
β_5	-0.0882	0.006	-13.953	0.000	-0.101	-0.076
β_6	0.0215	0.027	0.812	0.418	-0.031	0.074
β_7	-0.0273	0.038	-0.729	0.467	-0.102	0.047
β_8	-0.0275	0.038	-0.733	0.465	-0.102	0.047
β_9	-0.0039	0.006	-0.619	0.537	-0.016	0.009
β_{10}	0.0039	0.009	0.437	0.663	-0.014	0.022
β_{11}	0.0051	0.009	0.574	0.567	-0.013	0.023

Omnibus:	29.738	**Durbin-Watson:**	2.677
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	208.504
Skew:	0.355	**Prob(JB):**	5.29e-46
Kurtosis:	8.852	**Cond. No.**	75.6

Present Error vs. Number of Queries

For this regression analysis on each dataset, we define the same variables as in before, with the only change being that x_3 now represents the logarithm of the total number of consistent queries that RAP evaluates (rather than the size of the workload that RAP evaluates). With these variables, we define the same full and restricted regression models as before, and we fit both to the results of the RAP evaluations. Regression results for the full models (ADULT on left and LOANS on right) are stated below.

Dep. Variable:	present_err	**R-squared:**	0.889
Model:	OLS	**Adj. R-squared:**	0.879
Method:	Least Squares	**F-statistic:**	83.19
Covariance Type:	nonrobust	**Prob (F-statistic):**	3.83e-49
No. Observations:	126	**Log-Likelihood:**	227.07
Df Residuals:	114	**AIC:**	-430.1
Df Model:	11	**BIC:**	-396.1

	coef	std err	t	P> |t|	[0.025	0.975]
β_0	-0.3210	0.043	-7.438	0.000	-0.406	-0.235
β_1	0.2882	0.061	4.722	0.000	0.167	0.409
β_2	0.3014	0.061	4.939	0.000	0.181	0.422
β_3	0.0472	0.004	12.856	0.000	0.040	0.054
β_4	-0.0390	0.005	-7.516	0.000	-0.049	-0.029
β_5	-0.0436	0.005	-8.398	0.000	-0.054	-0.033
β_6	0.1198	0.057	2.110	0.037	0.007	0.232
β_7	-0.1237	0.080	-1.540	0.126	-0.283	0.035
β_8	-0.1189	0.080	-1.480	0.142	-0.278	0.040
β_9	-0.0127	0.005	-2.742	0.007	-0.022	-0.004
β_{10}	0.0123	0.007	1.886	0.062	-0.001	0.025
β_{11}	0.0124	0.007	1.894	0.061	-0.001	0.025

Omnibus:	53.796	**Durbin-Watson:**	1.512
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	189.737
Skew:	-1.528	**Prob(JB):**	6.30e-42
Kurtosis:	8.177	**Cond. No.**	572.

Dep. Variable:	present_err	**R-squared:**	0.887
Model:	OLS	**Adj. R-squared:**	0.877
Method:	Least Squares	**F-statistic:**	93.96
Covariance Type:	nonrobust	**Prob (F-statistic):**	7.68e-57
No. Observations:	144	**Log-Likelihood:**	180.50
Df Residuals:	132	**AIC:**	-337.0
Df Model:	11	**BIC:**	-301.4

	coef	std err	t	P> |t|	[0.025	0.975]
β_0	-0.6398	0.070	-9.171	0.000	-0.778	-0.502
β_1	0.5254	0.099	5.326	0.000	0.330	0.721
β_2	0.5873	0.099	5.952	0.000	0.392	0.782
β_3	0.0779	0.005	14.839	0.000	0.068	0.088
β_4	-0.0618	0.007	-8.321	0.000	-0.076	-0.047
β_5	-0.0709	0.007	-9.550	0.000	-0.086	-0.056
β_6	0.1453	0.096	1.509	0.134	-0.045	0.336
β_7	-0.1293	0.136	-0.949	0.344	-0.399	0.140
β_8	-0.1474	0.136	-1.082	0.281	-0.417	0.122
β_9	-0.0240	0.007	-3.647	0.000	-0.037	-0.011
β_{10}	0.0195	0.009	2.088	0.039	0.001	0.038
β_{11}	0.0227	0.009	2.431	0.016	0.004	0.041

Omnibus:	18.588	**Durbin-Watson:**	1.775
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	78.893
Skew:	-0.142	**Prob(JB):**	7.39e-18
Kurtosis:	6.615	**Cond. No.**	726.

Chapter 5

Conclusions

As data analysis increasingly relies on the use of personal data, the value of practical, useful, differentially private mechanisms increases as well. Significant research effort has been devoted to designing and analyzing mechanisms that satisfy DP. However, less attention has been devoted to bridging the gap between theory and practice in order to make these DP mechanisms useful for real-world applications. In this thesis, we make differential privacy more useful in practice by removing barriers that hinder its real-world adoption. We classify these barriers into three distinct challenges, which we individually resolve in the chapters of this thesis.

We begin in Chapter 2 by addressing the first high-level challenge of this thesis: improving trust models of DP to match individuals' privacy expectations while simultaneously solving practical, data-sensitive tasks with high utility. In this chapter, we define a hybrid model of DP that, in many practical scenarios, can better match individuals' privacy expectations than either of the two classic trust models in DP. Within this hybrid model, we address the high-level research question: *how can we design DP mechanisms that achieve high utility for problems of practical interest?* To answer this question, we study two fundamental data science problems: heavy hitter discovery and estimation as well as mean estimation. For both problems, we concretely determine: 1) how mechanisms' utilities within the hybrid model can be understood and contextualized relative to the classic trust models, 2) how DP mechanisms can be designed in the hybrid model, and 3) how the privacy and utility of hybrid mechanisms depends on the computations performed and on

interactions between the individuals. Our design and analysis of high-utility hybrid mechanisms improves the state of the art for both the heavy hitter problem and the mean estimation problem. Additionally, our findings from both problems yield insights into the power of the hybrid model and provide a blueprint for practitioners to design high-utility mechanisms within it in the future.

In Chapter 3, we turn our attention to the second high-level challenge of this thesis: enabling the quantification of privacy and utility for important, real-world DP mechanisms. Specifically, we address the high-level research question: *how can we rigorously define a hyperparameterized DP mechanism's privacy–utility trade-off, and then how can we design a practical method for quantifying it?* We answer the first part of this question by rigorously defining the privacy–utility trade-off in terms of the mechanism's privacy–utility Pareto front. We then define both the problem of estimating a privacy–utility Pareto front and how the quality of an estimated Pareto front is measured. We then detail `DPareto`, our proposed method to efficiently estimate a Pareto front using techniques from multi-objective Bayesian optimization. We comprehensively empirically evaluate `DPareto` on a variety of real-world machine learning tasks involving multiple models, DP optimizers, and datasets. Comparing `DPareto`'s results to baseline methods for Pareto front estimation, we find that `DPareto` is highly efficient and effective at estimating the privacy–utility Pareto fronts of complex, hyperparameterized DP mechanisms. Taken together, these evaluation results showcase `DPareto`'s practicality, enabling decision-makers to take informed actions when balancing the privacy–utility trade-off in real-world deployments of DP mechanisms.

In Chapter 4, we address the final high-level challenge of this thesis: resolving the open question of how to improve the effectiveness and efficiency of DP mechanisms for the foundational data analysis problem of privately answering a large number of queries. Concretely, we address the high-level research question: *to what extent are differentially private mechanisms able to answer a large number of statistical queries efficiently and with low error?* We analyze this problem in two settings, the classic prespecified queries setting and a new setting that we introduced where

only partial knowledge of the queries is available to the DP mechanism in advance. In both settings, our contributions are grounded in the state-of-the-art DP mechanism for answering large numbers of queries, the RAP mechanism. In the prespecified queries setting, we perform a focused but thorough reproducibility study on Aydore et al.'s original evaluation of RAP to clarify its value and strengthen its adoptability for practical uses. We also expand the class of queries that RAP is capable of evaluating, thus extending RAP's applicability in practice. Aside from the prespecified queries setting, we concretely specify a new partial knowledge setting where a mechanism is provided with a set of historically posed queries that are similar to queries that will be posed in the future. In this setting, we define a machine learning inspired utility measure to quantify a mechanism's ability to answer such future queries. Then, leveraging this utility measure, we evaluate RAP's suitability for generating synthetic datasets to answer queries posed in the future, finding that it is both efficient and effective. Our findings in this chapter further the state of the art in differentially private large-scale query answering and open new directions for future work on other problems in differential privacy within our newly defined partial knowledge setting.

Bibliography

[Aba+16] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. "Deep learning with differential privacy". In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. 2016, pp. 308–318.

[ABM19] Noga Alon, Raef Bassily, and Shay Moran. "Limits of private learning with access to public data". In: *Advances in Neural Information Processing Systems* (2019).

[Abo18a] John M. Abowd. *Disclosure avoidance for block level data and protection of confidentiality in public tabulations*. Census Scientific Advisory Committee (Fall Meeting). 2018. URL: https://www2.census.gov/cac/sac/meetings/2018-12/abowd-disclosure-avoidance.pdf.

[Abo18b] John M Abowd. "Staring-down the database reconstruction theorem". In: *Joint Statistical Meetings, Vancouver, BC*. 2018, p. 234.

[Ach+18] Jayadev Acharya, Gautam Kamath, Ziteng Sun, and Huanyu Zhang. "INSPECTRE: Privately estimating the unseen". In: *International Conference on Machine Learning*. 2018, pp. 30–39.

[ADK20] Brendan Avent, Yatharth Dubey, and Aleksandra Korolova. "The power of the hybrid model for mean estimation". In: *Proceedings on Privacy Enhancing Technologies* 4 (2020), pp. 48–68.

[AK23] Brendan Avent and Aleksandra Korolova. "Pushing the Boundaries of Private, Large-Scale Query Answering". In: *arXiv preprint arXiv:2302.04833* (2023).

[Akt+20] Ahmet Aktay, Shailesh Bavadekar, Gwen Cossoul, John Davis, Damien Desfontaines, Alex Fabrikant, Evgeniy Gabrilovich, Krishna Gadepalli, Bryant Gipson, Miguel Guevara, et al. "Google COVID-19 community mobility reports: Anonymization process description (version 1.1)". In: *arXiv preprint arXiv:2004.04145* (2020).

[ARL12] Mauricio A. Álvarez, Lorenzo Rosasco, and Neil D. Lawrence. "Kernels for vector-valued functions: A review". In: *Foundations and Trends in Machine Learning* 4.3 (Mar. 2012), pp. 195–266. ISSN: 1935-8237.

[ARL+12] Mauricio A Alvarez, Lorenzo Rosasco, Neil D Lawrence, et al. "Kernels for vector-valued functions: A review". In: *Foundations and Trends® in Machine Learning* 4.3 (2012), pp. 195–266.

[AS19a] John M Abowd and Ian M Schmutte. "An economic analysis of privacy protection and statistical accuracy as social choices". In: *American Economic Review* (2019), pp. 171–202.

[AS19b] John M Abowd and Ian M Schmutte. "An economic analysis of privacy protection and statistical accuracy as social choices". In: *American Economic Review* 109.1 (2019), pp. 171–202.

[AS21] Amazon-Science. *K-way marginal selection methodology, issue 2, amazon-science/relaxed-adaptive-projection*. 2021. URL: https://github.com/amazon-science/relaxed-adaptive-projection/issues/2.

[AS68] M Abramowitz and IA Stegun. *Handbook of Mathematical Functions*. 7th. Washington, DC: US Government Printing Office, 1968.

[AS+94] Rakesh Agrawal, Ramakrishnan Srikant, et al. "Fast algorithms for mining association rules". In: *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*. Vol. 1215. Citeseer. 1994, pp. 487–499.

[Ave+17] Brendan Avent, Aleksandra Korolova, David Zeber, Torgeir Hovden, and Benjamin Livshits. "BLENDER: Enabling local search with a hybrid differential privacy model". In: *26th USENIX Security Symposium (USENIX Security 17)*. USENIX Association, 2017, pp. 747–764. ISBN: 978-1-931971-40-9. URL: https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/avent.

[Ave+19] Brendan Avent, Aleksandra Korolova, David Zeber, Torgeir Hovden, and Benjamin Livshits. "BLENDER: Enabling local search with a hybrid differential privacy model". In: *Journal of Privacy and Confidentiality* 9.2 (2019).

[Ave+20] Brendan Avent, Javier González, Tom Diethe, Andrei Paleyes, and Borja Balle. "Automatic discovery of privacy–utility Pareto fronts". In: *Proceedings on Privacy Enhancing Technologies* 4 (2020), pp. 5–23.

[Ayd+21] Sergul Aydore, William Brown, Michael Kearns, Krishnaram Kenthapadi, Luca Melis, Aaron Roth, and Ankit A Siva. "Differentially private query release through adaptive projection". In: *International Conference on Machine Learning*. PMLR. 2021, pp. 457–467.

[Bal+17] Maria-Florina Balcan, Travis Dick, Yingyu Liang, Wenlong Mou, and Hongyang Zhang. “Differentially private clustering in high-dimensional euclidean spaces”. In: *Proceedings of the 34th International Conference on Machine Learning (ICML)*. Vol. 70. 2017, pp. 322–331.

[Bal+19a] Borja Balle, James Bell, Adria Gascon, and Kobbi Nissim. “Differentially private summation with multi-message shuffling”. In: *arXiv preprint arXiv:1906.09116* (2019).

[Bal+19b] Borja Balle, James Bell, Adrià Gascón, and Kobbi Nissim. “The privacy blanket of the shuffle model”. In: *Annual International Cryptology Conference*. Springer. 2019, pp. 638–667.

[Bal+20] Borja Balle, Peter Kairouz, Brendan McMahan, Om Thakkar, and Abhradeep Guha Thakurta. “Privacy amplification via random check-ins”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 4623–4634.

[Bar+07] Boaz Barak, Kamalika Chaudhuri, Cynthia Dwork, Satyen Kale, Frank McSherry, and Kunal Talwar. “Privacy, accuracy, and consistency too: a holistic solution to contingency table release”. In: *Proceedings of the Twenty-sixth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*. 2007, pp. 273–282.

[Bas19] Raef Bassily. “Linear queries estimation with local differential privacy”. In: *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR. 2019, pp. 721–729.

[Bas+20a] Raef Bassily, Albert Cheu, Shay Moran, Aleksandar Nikolov, Jonathan Ullman, and Steven Wu. “Private query release assisted by public data”. In: *International Conference on Machine Learning*. PMLR. 2020, pp. 695–703.

[Bas+20b] Raef Bassily, Kobbi Nissim, Uri Stemmer, and Abhradeep Thakurta. “Practical locally private heavy hitters”. In: *Journal of Machine Learning Research* 21 (2020), pp. 16–1.

[Bav+20] Shailesh Bavadekar, Andrew Dai, John Davis, Damien Desfontaines, Ilya Eckstein, Katie Everett, Alex Fabrikant, Gerardo Flores, Evgeniy Gabrilovich, Krishna Gadepalli, et al. “Google COVID-19 search trends symptoms dataset: Anonymization process description (version 1.0)”. In: *arXiv preprint arXiv:2009.01265* (2020).

[Bav+21] Shailesh Bavadekar, Adam Boulanger, John Davis, Damien Desfontaines, Evgeniy Gabrilovich, Krishna Gadepalli, Badih Ghazi, Tague Griffith, Jai Gupta, Chaitanya Kamath, et al. “Google COVID-19 vaccination search insights: Anonymization process description”. In: *arXiv preprint arXiv:2107.01179* (2021).

[BBG18] Borja Balle, Gilles Barthe, and Marco Gaboardi. “Privacy amplification by subsampling: Tight analyses via couplings and divergences”. In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada.* 2018.

[BC20] Victor Balcer and Albert Cheu. “Separating local & shuffled differential privacy via histograms”. In: *1st Conference on Information-Theoretic Cryptography (ITC 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik. 2020.

[BDV18] Maria-Florina Balcan, Travis Dick, and Ellen Vitercik. “Dispersion for data-driven algorithm design, online learning, and private optimization”. In: *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2018, pp. 603–614.

[Bei+20] Amos Beimel, Aleksandra Korolova, Kobbi Nissim, Or Sheffet, and Uri Stemmer. “The power of synergy in differential privacy: Combining a small curator with local randomizers”. In: *1st Conference on Information-Theoretic Cryptography*. 2020.

[Ber+22] Skye Berghel, Philip Bohannon, Damien Desfontaines, Charles Estes, Sam Haney, Luke Hartman, Michael Hay, Ashwin Machanavajjhala, Tom Magerlein, Gerome Miklau, et al. “Tumult Analytics: A robust, easy-to-use, scalable, and expressive framework for differential privacy”. In: *arXiv preprint arXiv:2212.04133* (2022).

[BF16] Artem Barger and Dan Feldman. “k-Means for streaming and distributed big sparse data”. In: *Proceedings of the 2016 SIAM International Conference on Data Mining*. SIAM. 2016, pp. 342–350.

[Bha+10] Raghav Bhaskar, Srivatsan Laxman, Adam Smith, and Abhradeep Thakurta. “Discovering frequent patterns in sensitive data”. In: *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2010, pp. 503–512.

[Bil08] Patrick Billingsley. *Probability and measure*. John Wiley & Sons, 2008.

[Bis+20] Sourav Biswas, Yihe Dong, Gautam Kamath, and Jonathan Ullman. “Coinpress: Practical private mean and covariance estimation”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 14475–14485.

[Bit+17] Andrea Bittau, Ulfar Erlingsson, Petros Maniatis, Ilya Mironov, Ananth Raghunathan, David Lie, Mitch Rudominer, Ushasree Kode, Julien Tinnes, and Bernhard Seefeld. “Prochlo: Strong privacy for analytics in the crowd”. In: *Proceedings of the 26th Symposium on Operating Systems Principles*. ACM. 2017, pp. 441–459.

[BLR08] Avrim Blum, Katrina Ligett, and Aaron Roth. “A learning theory approach to non-interactive database privacy”. In: *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*. 2008, pp. 609–618.

[Blu+05] Avrim Blum, Cynthia Dwork, Frank McSherry, and Kobbi Nissim. “Practical privacy: the SuLQ framework”. In: *Proceedings of the Twenty-fourth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*. 2005, pp. 128–138.

[BM06] Peter L Bartlett and Shahar Mendelson. “Empirical minimization”. In: *Probability Theory and Related Fields* 135.3 (2006), pp. 311–334.

[BMA19] Raef Bassily, Shay Moran, and Noga Alon. “Limits of private learning with access to public data”. In: *Advances in Neural Information Processing Systems*. 2019, pp. 10342–10352.

[BNO08] Amos Beimel, Kobbi Nissim, and Eran Omri. “Distributed private data analysis: Simultaneously solving how and what”. In: *Annual International Cryptology Conference*. Springer. 2008, pp. 451–468.

[BNS13] Amos Beimel, Kobbi Nissim, and Uri Stemmer. “Private learning and sanitization: Pure vs. approximate differential privacy”. In: *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*. Springer, 2013, pp. 363–378.

[Bot10] Léon Bottou. “Large-scale machine learning with stochastic gradient descent”. In: *Proceedings of COMPSTAT’2010*. Springer, 2010, pp. 177–186.

[Bra+18] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. *JAX: Composable transformations of Python+NumPy programs*. Version 0.3.13. 2018. URL: http://github.com/google/jax.

[Bri90] John S Bridle. “Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters”. In: *Advances in Neural Information Processing Systems*. 1990, pp. 211–217.

[BS15] Raef Bassily and Adam Smith. “Local, private, efficient protocols for succinct histograms”. In: *Proceedings of the Forty-seventh Annual ACM Symposium on Theory of Computing*. 2015, pp. 127–135.

[BS16] Mark Bun and Thomas Steinke. “Concentrated differential privacy: Simplifications, extensions, and lower bounds”. In: *Theory of Cryptography Conference*. Springer. 2016, pp. 635–658.

[BS19] Mark Bun and Thomas Steinke. "Average-case averages: Private algorithms for smooth sensitivity and mean estimation". In: *Advances in Neural Information Processing Systems*. 2019, pp. 181–191.

[Bur16] UC Bureau. "American Community Survey (ACS)". In: *The United States Census Bureau nd https://www. census. gov/programs-surveys/acs (accessed May 5, 2021)* (2016).

[Bur22] US Census Bureau. *The Census Bureau's simulated reconstruction-abetted re-identification attack on the 2010 census*. 2022. URL: https://www.census.gov/data/academy/webinars/2021/disclosure-avoidance-series/simulated-reconstruction-abetted-re-identification-attack-on-the-2010-census.html.

[BW18] Borja Balle and Yu-Xiang Wang. "Improving the gaussian mechanism for differential privacy: Analytical calibration and optimal denoising". In: *International Conference on Machine Learning*. PMLR. 2018, pp. 394–403.

[BY+07] Ricardo Baeza-Yates, Aristides Gionis, Flavio Junqueira, Vanessa Murdock, Vassilis Plachouras, and Fabrizio Silvestri. "The impact of caching on search engines". In: *ACM SIGIR Conference on Research and Development in Information Retrieval*. 2007, pp. 183–190.

[BY+08] Ricardo Baeza-Yates, Aristides Gionis, Flavio P Junqueira, Vanessa Murdock, Vassilis Plachouras, and Fabrizio Silvestri. "Design trade-offs for search engine caching". In: *ACM Transactions on the Web* 2.4 (2008), p. 20.

[BZH06] Michael Barbaro, Tom Zeller, and Saul Hansell. "A face is exposed for AOL searcher no. 4417749". In: *New York Times* 9.2008 (2006), p. 8.

[Cal] University of California. *2010 annual report on emplyee compensation*. https://ucnet.universityofcalifornia.edu/compensation-and-benefits/compensation/index.html.

[Car+18] Nicholas Carlini, Chang Liu, Jernej Kos, Úlfar Erlingsson, and Dawn Song. "The secret sharer: Measuring unintended neural network memorization & Extracting secrets". In: *Proceedings of the 27th USENIX Security Symposium*. 2018.

[CCFC02] Moses Charikar, Kevin Chen, and Martin Farach-Colton. "Finding frequent items in data streams". In: *International Colloquium on Automata, Languages, and Programming*. Springer. 2002, pp. 693–703.

[CDD14a] Ivo Couckuyt, Dirk Deschrijver, and Tom Dhaene. "Fast calculation of multiobjective probability of improvement and expected improvement criteria for Pareto optimization". In: *Journal of Global Optimization* 60.3 (2014), pp. 575–594.

[CDD14b] Ivo Couckuyt, Dirk Deschrijver, and Tom Dhaene. “Fast calculation of multiobjective probability of improvement and expected improvement criteria for Pareto optimization”. In: *Journal of Global Optimization* 60.3 (2014), pp. 575–594.

[Cha+14] Karthekeyan Chandrasekaran, Justin Thaler, Jonathan Ullman, and Andrew Wan. “Faster private release of marginals on small databases”. In: *Proceedings of the 5th Conference on Innovations in Theoretical Computer Science*. 2014, pp. 387–402.

[Che+12] Mahdi Cheraghchi, Adam Klivans, Pravesh Kothari, and Homin K Lee. “Submodular functions are noise stable”. In: *Proceedings of the Twenty-third Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM. 2012, pp. 1586–1592.

[Che+16] Yiling Chen, Stephen Chong, Ian A Kash, Tal Moran, and Salil Vadhan. “Truthful mechanisms for agents that value privacy”. In: *ACM Transactions on Economics and Computation (TEAC)* 4.3 (2016), pp. 1–30.

[Che+19] Albert Cheu, Adam Smith, Jonathan Ullman, David Zeber, and Maxim Zhilyaev. “Distributed differential privacy via shuffling”. In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2019, pp. 375–403.

[CHS14] Kamalika Chaudhuri, Daniel J Hsu, and Shuang Song. “The large margin mechanism for differentially private maximization”. In: *Advances in Neural Information Processing Systems*. 2014, pp. 1287–1295.

[CKS18] Graham Cormode, Tejas Kulkarni, and Divesh Srivastava. “Marginal release under local differential privacy”. In: *Proceedings of the 2018 International Conference on Management of Data*. 2018, pp. 131–146.

[CKS20] Clément L Canonne, Gautam Kamath, and Thomas Steinke. “The discrete gaussian for differential privacy”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 15676–15688.

[Cor+03] Graham Cormode, Flip Korn, Shanmugavelayutham Muthukrishnan, and Divesh Srivastava. “Finding hierarchical heavy hitters in data streams”. In: *Proceedings 2003 VLDB Conference*. Elsevier. 2003, pp. 464–475.

[CR21] Mark Cesar and Ryan Rogers. “Bounding, concentrating, and truncating: Unifying privacy loss composition for data analytics”. In: *Algorithmic Learning Theory*. PMLR. 2021, pp. 421–457.

[CRB22] Miranda Christ, Sarah Radway, and Steven M Bellovin. “Differential privacy and swapping: Examining de-identification’s impact on minority representation and privacy preservation in the US census”. In: *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE Computer Society. 2022, pp. 1564–1564.

[Dai22] Donna Daily. *Disclosure avoidance protections for the American Community Survey*. 2022. URL: https://www.census.gov/newsroom/blogs/random-samplings/2022/12/disclosure-avoidance-protections-acs.html.

[Daj+17] Aref N Dajani, Amy D Lauger, Phyllis E Singer, Daniel Kifer, Jerome P Reiter, Ashwin Machanavajjhala, Simson L Garfinkel, Scot A Dahl, Matthew Graham, Vishesh Karwa, et al. "The modernization of statistical disclosure limitation at the US Census Bureau". In: *September 2017 meeting of the Census Scientific Advisory Committee*. 2017.

[Dek+22] Inbal Dekel, Rachel Cummings, Ori Heffetz, and Katrina Ligett. *The privacy elasticity of behavior: Conceptualization and application*. Tech. rep. National Bureau of Economic Research, 2022.

[DF19] Amit Daniely and Vitaly Feldman. "Locally private learning without interaction requires separation". In: *Advances in Neural Information Processing Systems* 32 (2019).

[Dic+22] Travis Dick, Cynthia Dwork, Michael Kearns, Terrance Liu, Aaron Roth, Giuseppe Vietri, and Zhiwei Steven Wu. "Confidence-ranked reconstruction of census microdata from published statistics". In: *arXiv preprint arXiv:2211.03128* (2022).

[DJW13] John C Duchi, Michael I Jordan, and Martin J Wainwright. "Local privacy and statistical minimax rates". In: *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*. IEEE. 2013, pp. 429–438.

[DJW18] John C. Duchi, Michael I. Jordan, and Martin J. Wainwright. "Minimax optimal procedures for locally private estimation". In: *Journal of the American Statistical Association* 113.521 (2018), pp. 182–201.

[DKY17] Bolin Ding, Janardhan Kulkarni, and Sergey Yekhanin. "Collecting telemetry data privately". In: *Advances in Neural Information Processing Systems* 30 (2017).

[DMT07] Cynthia Dwork, Frank McSherry, and Kunal Talwar. "The price of privacy and the limits of LP decoding". In: *Proceedings of the Thirty-ninth Annual ACM Symposium on Theory of Computing*. 2007, pp. 85–94.

[DN03] Irit Dinur and Kobbi Nissim. "Revealing information while preserving privacy". In: *Proceedings of the Twenty-second ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*. 2003, pp. 202–210.

[DR+14] Cynthia Dwork, Aaron Roth, et al. "The algorithmic foundations of differential privacy". In: *Foundations and Trends in Theoretical Computer Science* 9.3-4 (2014), pp. 211–407.

[DR19] David Durfee and Ryan M Rogers. “Practical differentially private top-k selection with pay-what-you-get composition”. In: *Advances in Neural Information Processing Systems* 32 (2019).

[DR82] Tore Dalenius and Steven P Reiss. “Data-swapping: A technique for disclosure control”. In: *Journal of statistical planning and inference* 6.1 (1982), pp. 73–85.

[DRV10] Cynthia Dwork, Guy N Rothblum, and Salil Vadhan. “Boosting and differential privacy”. In: *Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2010, pp. 51–60.

[Du+20] Wenxin Du, Canyon Foot, Monica Moniot, Andrew Bray, and Adam Groce. “Differentially private confidence intervals”. In: *arXiv preprint arXiv:2001.02285* (2020).

[Dwo+06a] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. “Our data, ourselves: Privacy via distributed noise generation”. In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2006, pp. 486–503.

[Dwo+06b] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. “Calibrating noise to sensitivity in private data analysis”. In: *Theory of Cryptography Conference*. Springer. 2006, pp. 265–284.

[Dwo+09] Cynthia Dwork, Moni Naor, Omer Reingold, Guy N Rothblum, and Salil Vadhan. “On the complexity of differentially private data release: Efficient algorithms and hardness results”. In: *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing*. 2009, pp. 381–390.

[Dwo11] Cynthia Dwork. “A firm foundation for private data analysis”. In: *Communications of the ACM* 54.1 (2011), pp. 86–95.

[Dwo+17] Cynthia Dwork, Adam Smith, Thomas Steinke, and Jonathan Ullman. “Exposed! A survey of attacks on private data”. In: *Annual Review of Statistics and Its Application* 4 (2017), pp. 61–84.

[DY08] Cynthia Dwork and Sergey Yekhanin. “New efficient attacks on statistical disclosure control mechanisms”. In: *Annual International Cryptology Conference*. Springer. 2008, pp. 469–480.

[EK08] Michael Emmerich and Jan-willem Klinkenberg. “The computation of the expected improvement in dominated hypervolume of Pareto front approximations”. In: *Rapport Technique, Leiden University* 34 (2008), pp. 7–3.

[EPK14] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. “RAPPOR: Randomized aggregatable privacy-preserving ordinal response”. In: *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications security*. 2014, pp. 1054–1067.

[Erl+19] 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: *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM. 2019, pp. 2468–2479.

[Fel17] Vitaly Feldman. “Dealing with range anxiety in mean estimation via statistical queries”. In: *International Conference on Algorithmic Learning Theory*. 2017, pp. 629–640.

[Fel+18] Vitaly Feldman, Ilya Mironov, Kunal Talwar, and Abhradeep Thakurta. “Privacy amplification by iteration”. In: *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2018, pp. 521–532.

[FPE16a] Giulia Fanti, Vasyl Pihur, and Úlfar Erlingsson. “Building a RAPPOR with the Unknown: Privacy-Preserving Learning of Associations and Data Dictionaries”. In: *Proceedings on Privacy Enhancing Technologies* 3 (2016), pp. 41–61.

[FPE16b] Giulia Fanti, Vasyl Pihur, and Úlfar Erlingsson. “Building a RAPPOR with the unknown: Privacy-preserving learning of associations and data dictionaries”. In: *Proceedings on Privacy Enhancing Technologies (PETS)* 2016.3 (2016), pp. 41–61.

[Fra10] Andrew Frank. “UCI machine learning repository”. In: *http://archive.ics.uci.edu/ml* (2010).

[Gab+14] Marco Gaboardi, Emilio Jesús Gallego Arias, Justin Hsu, Aaron Roth, and Zhiwei Steven Wu. “Dual query: Practical private query release for high dimensional data”. In: *International Conference on Machine Learning*. 2014.

[GAM19] Simson Garfinkel, John M Abowd, and Christian Martindale. “Understanding database reconstruction attacks on public data”. In: *Communications of the ACM* 62.3 (2019), pp. 46–53.

[GAP18] Simson L Garfinkel, John M Abowd, and Sarah Powazek. “Issues encountered deploying differential privacy”. In: *Proceedings of the 2018 Workshop on Privacy in the Electronic Society*. 2018.

[Ge+19] Chang Ge, Xi He, Ihab F Ilyas, and Ashwin Machanavajjhala. “Apex: Accuracy-aware differentially private data exploration”. In: *Proceedings of the 2019 International Conference on Management of Data*. ACM. 2019, pp. 177–194.

[Gen+20] Quan Geng, Wei Ding, Ruiqi Guo, and Sanjiv Kumar. “Tight analysis of privacy and utility tradeoff in approximate differential privacy”. In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2020, pp. 89–99.

[GH06] Andrew Gelman and Jennifer Hill. *Data analysis using regression and multilevel/hierarchical models*. Cambridge University Press, 2006.

[Gha+20a] Badih Ghazi, Noah Golowich, Ravi Kumar, Pasin Manurangsi, Rasmus Pagh, and Ameya Velingker. “Pure differentially private summation from anonymous messages”. In: *1st Conference on Information-Theoretic Cryptography*. 2020.

[Gha+20b] Badih Ghazi, Pasin Manurangsi, Rasmus Pagh, and Ameya Velingker. “Private aggregation from fewer anonymous messages”. In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2020, pp. 798–827.

[Gha+21] Badih Ghazi, Noah Golowich, Ravi Kumar, Rasmus Pagh, and Ameya Velingker. “On the power of multiple anonymous messages: Frequency estimation and selection in the shuffle model of differential privacy”. In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2021, pp. 463–488.

[GHV20] Marco Gaboardi, Michael Hay, and Salil Vadhan. “A programming framework for OpenDP”. In: *Manuscript, May* (2020).

[Gol+17] Daniel Golovin, Benjamin Solnik, Subhodeep Moitra, Greg Kochanski, John Karro, and D Sculley. “Google vizier: A service for black-box optimization”. In: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM. 2017, pp. 1487–1495.

[GPV19] Badih Ghazi, Rasmus Pagh, and Ameya Velingker. “Scalable and differentially private distributed aggregation in the shuffled model”. In: *arXiv preprint arXiv:1906.08320* (2019).

[GRS12] Arpita Ghosh, Tim Roughgarden, and Mukund Sundararajan. “Universally utility-maximizing privacy mechanisms”. In: *SIAM Journal on Computing* 41.6 (2012), pp. 1673–1693.

[GRS19] Marco Gaboardi, Ryan Rogers, and Or Sheffet. “Locally private mean estimation: Z-test and tight confidence intervals”. In: *Proceedings of Machine Learning Research*. Ed. by Kamalika Chaudhuri and Masashi Sugiyama. Vol. 89. PMLR, 2019, pp. 2545–2554. URL: http://proceedings.mlr.press/v89/gaboardi19a.html.

[GRU12] Anupam Gupta, Aaron Roth, and Jonathan Ullman. “Iterative constructions and private data release”. In: *Theory of Cryptography Conference*. Springer. 2012, pp. 339–356.

[GSC17] Joseph Geumlek, Shuang Song, and Kamalika Chaudhuri. “Rényi differential privacy mechanisms for posterior sampling”. In: *Advances in Neural Information Processing Systems*. 2017, pp. 5289–5298.

[Gup+13] Anupam Gupta, Moritz Hardt, Aaron Roth, and Jonathan Ullman. “Privately releasing conjunctions and the statistical query barrier”. In: *SIAM Journal on Computing* 42.4 (2013), pp. 1494–1520.

[GV14] Quan Geng and Pramod Viswanath. “The optimal mechanism in differential privacy”. In: *2014 IEEE International Symposium on Information Theory*. IEEE. 2014, pp. 2371–2375.

[GV15] Quan Geng and Pramod Viswanath. “The optimal noise-adding mechanism in differential privacy”. In: *IEEE Transactions on Information Theory* 62.2 (2015), pp. 925–951.

[Gym+13] Melissa Gymrek, Amy L McGuire, David Golan, Eran Halperin, and Yaniv Erlich. “Identifying personal genomes by surname inference”. In: *Science* 339.6117 (2013), pp. 321–324.

[Has+09] Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. *The elements of statistical learning: data mining, inference, and prediction*. Vol. 2. Springer, 2009.

[HCB16] Jihun Hamm, Yingjun Cao, and Mikhail Belkin. “Learning privately from multiparty data”. In: *International Conference on Machine Learning*. PMLR. 2016, pp. 555–563.

[HLM12] Moritz Hardt, Katrina Ligett, and Frank McSherry. “A simple and practical algorithm for differentially private data release”. In: *Proceedings of the 25th International Conference on Neural Information Processing Systems*. 2012.

[Hol+19] Naoise Holohan, Stefano Braghin, Pól Mac Aonghusa, and Killian Levacher. “Diffprivlib: The IBM differential privacy library”. In: *ArXiv e-prints* 1907.02444 [cs.CR] (July 2019).

[Hol79] Sture Holm. “A simple sequentially rejective multiple test procedure”. In: *Scandinavian Journal of Statistics* (1979), pp. 65–70.

[HPY00] Jiawei Han, Jian Pei, and Yiwen Yin. “Mining frequent patterns without candidate generation”. In: *ACM Sigmod Record* 29.2 (2000), pp. 1–12.

[HR10] Moritz Hardt and Guy N Rothblum. “A multiplicative weights mechanism for privacy-preserving data analysis”. In: *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*. IEEE. 2010, pp. 61–70.

[HRS12] Moritz Hardt, Guy N Rothblum, and Rocco A Servedio. “Private data release via learning thresholds”. In: *Proceedings of the Twenty-third Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM. 2012, pp. 168–187.

[HW04] Kohei Hatano and Osamu Watanabe. “Learning r-of-k functions by boosting”. In: *International Conference on Algorithmic Learning Theory*. Springer. 2004, pp. 114–126.

[JE13] Zhanglong Ji and Charles Elkan. “Differential privacy based on importance weighting”. In: *Machine learning* 93.1 (2013), pp. 163–183.

[Jen+17] Rodolphe Jenatton, Cedric Archambeau, Javier González, and Matthias Seeger. “Bayesian optimization with tree-structured dependencies”. In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. International Convention Centre, Sydney, Australia: PMLR, 2017, pp. 1655–1664.

[JK02] Kalervo Järvelin and Jaana Kekäläinen. “Cumulated gain-based evaluation of IR techniques”. In: *ACM Transactions on Information Systems (TOIS)* 20.4 (2002), pp. 422–446.

[Jos+19a] Matthew Joseph, Janardhan Kulkarni, Jieming Mao, and Steven Z Wu. “Locally private gaussian estimation”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2019, pp. 2980–2989.

[Jos+19b] Matthew Joseph, Jieming Mao, Seth Neel, and Aaron Roth. “The role of interactivity in local differential privacy”. In: *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2019, pp. 94–105.

[Kam+19a] Gautam Kamath, Jerry Li, Vikrant Singhal, and Jonathan Ullman. “Privately learning high-dimensional distributions”. In: *Proceedings of the Thirty-Second Conference on Learning Theory*. Ed. by Alina Beygelzimer and Daniel Hsu. Vol. 99. PMLR, 2019, pp. 1853–1902. URL: http://proceedings.mlr.press/v99/kamath19a.html.

[Kam+19b] Gautam Kamath, Or Sheffet, Vikrant Singhal, and Jonathan Ullman. “Differentially private algorithms for learning mixtures of separated gaussians”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2019, pp. 168–180.

[Kas+11] Shiva Prasad Kasiviswanathan, Homin K Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. “What can we learn privately?” In: *SIAM Journal on Computing* 40.3 (2011), pp. 793–826.

[KB15] Diederik P Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *Proceedings of International Conference on Learning Representations (ICLR)*. 2015.

[Kea+87] Michael Kearns, Ming Li, Leonard Pitt, and Leslie Valiant. “On the learnability of Boolean formulae”. In: *Proceedings of the Nineteenth Annual ACM symposium on Theory of Computing*. 1987, pp. 285–295.

[KK23] Antti Koskela and Tejas Kulkarni. “Practical differentially private hyperparameter tuning with subsampling”. In: *arXiv preprint arXiv:2301.11989* (2023).

[Kle+17] Aaron Klein, Stefan Falkner, Simon Bartels, Philipp Hennig, and Frank Hutter. “Fast Bayesian optimization of machine learning hyperparameters on large datasets”. In: *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*. Ed. by Aarti Singh and Jerry Zhu. Vol. 54. Proceedings of Machine Learning Research. Fort Lauderdale, FL, USA: PMLR, 2017, pp. 528–536. URL: http://proceedings.mlr.press/v54/klein17a.html.

[Knu+17] Nicolas Knudde, Joachim van der Herten, Tom Dhaene, and Ivo Couckuyt. “GPflowOpt: A Bayesian optimization library using tensorflow”. In: *arXiv preprint arXiv:1711.03845* (2017).

[Koh+96] Ron Kohavi et al. “Scaling up the accuracy of naive-bayes classifiers: A decision-tree hybrid”. In: *KDD*. Vol. 96. 1996, pp. 202–207.

[Kop21] Andreas Kopp. “Microsoft smartnoise differential privacy machine learning case studies”. In: *Microsoft Azure White Papers* (2021).

[Kor+09] Aleksandra Korolova, Krishnaram Kenthapadi, Nina Mishra, and Alexandros Ntoulas. “Releasing search queries and clicks privately”. In: *Proceedings of the International Conference on World Wide Web (WWW)*. ACM. 2009, pp. 171–180.

[Kor12] Aleksandra Korolova. *Protecting privacy when mining and sharing user data*. Stanford University, 2012.

[Kot+17] Ios Kotsogiannis, Ashwin Machanavajjhala, Michael Hay, and Gerome Miklau. “Pythia: Data dependent differentially private algorithm selection”. In: *Proceedings of the 2017 ACM International Conference on Management of Data*. ACM. 2017, pp. 1323–1337.

[KOV14] Peter Kairouz, Sewoong Oh, and Pramod Viswanath. “Extremal mechanisms for local differential privacy”. In: *Advances in neural information processing systems* 27 (2014), pp. 2879–2887.

[KSU20] Gautam Kamath, Vikrant Singhal, and Jonathan Ullman. “Private mean estimation of heavy-tailed distributions”. In: *Conference on Learning Theory*. PMLR. 2020, pp. 2204–2235.

[Kus+15] Matt Kusner, Jacob Gardner, Roman Garnett, and Kilian Weinberger. "Differentially private Bayesian optimization". In: *International Conference on Machine Learning*. 2015, pp. 918–927.

[KV18] Vishesh Karwa and Salil Vadhan. "Finite sample differentially private confidence intervals". In: *9th Innovations in Theoretical Computer Science Conference (ITCS 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik. 2018.

[KW52] Jack Kiefer and Jacob Wolfowitz. "Stochastic estimation of the maximum of a regression function". In: *The Annals of Mathematical Statistics* (1952), pp. 462–466.

[LeC+98] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[Li+12] Ninghui Li, Wahbeh Qardaji, Dong Su, and Jianneng Cao. "PrivBasis: frequent itemset mining with differential privacy". In: *Proceedings of the VLDB Endowment* 5.11 (2012), pp. 1340–1351.

[Li13] Chao Li. *Optimizing linear queries under differential privacy*. University of Massachusetts Amherst, 2013.

[Li+15] Chao Li, Gerome Miklau, Michael Hay, Andrew McGregor, and Vibhor Rastogi. "The matrix mechanism: optimizing linear counting queries under differential privacy". In: *The VLDB journal* 24.6 (2015), pp. 757–781.

[Li+17] Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. "Hyperband: A novel bandit-based approach to hyperparameter optimization". In: *Journal of Machine Learning Research* 18.1 (Jan. 2017), pp. 6765–6816. ISSN: 1532-4435. URL: http://dl.acm.org/citation.cfm?id=3122009.3242042.

[Lig+17] Katrina Ligett, Seth Neel, Aaron Roth, Bo Waggoner, and Steven Z Wu. "Accuracy first: Selecting a differential privacy level for accuracy constrained ERM". In: *Advances in Neural Information Processing Systems*. 2017, pp. 2566–2576.

[Lit88] Nick Littlestone. "Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm". In: *Machine learning* 2.4 (1988), pp. 285–318.

[Liu+21] Terrance Liu, Giuseppe Vietri, Thomas Steinke, Jonathan Ullman, and Steven Wu. "Leveraging public data for practical private query release". In: *International Conference on Machine Learning*. PMLR. 2021, pp. 6968–6977.

[LS20] Zhigang Lu and Hong Shen. "Differentially private k-Means clustering with convergence guarantee". In: *IEEE Transactions on Dependable and Secure Computing* 18.4 (2020), pp. 1541–1552.

[LSL17] Min Lyu, Dong Su, and Ninghui Li. “Understanding the sparse vector technique for differential privacy”. In: *Proceedings of the VLDB Endowment* (2017).

[LT19] Jingcheng Liu and Kunal Talwar. “Private selection from private candidates”. In: *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*. STOC 2019. Phoenix, AZ, USA: ACM, 2019, pp. 298–309. ISBN: 978-1-4503-6705-9. DOI: 10.1145/3313276.3316377.

[Luc12] R Duncan Luce. *Individual choice behavior: A theoretical analysis*. Courier Corporation, 2012.

[Luk72] Eugene Lukacs. “A survey of the theory of characteristic functions”. In: *Advances in Applied Probability* 4.1 (1972), pp. 1–37.

[LVW21] Terrance Liu, Giuseppe Vietri, and Steven Z Wu. “Iterative methods for private synthetic data: Unifying framework and new methods”. In: *Advances in Neural Information Processing Systems* 34 (2021).

[MA16] Andre Martins and Ramon Astudillo. “From softmax to sparsemax: A sparse model of attention and multi-label classification”. In: *International Conference on Machine Learning*. PMLR. 2016, pp. 1614–1623.

[MA18] H. Brendan McMahan and Galen Andrew. “A general approach to adding differential privacy to iterative training procedures”. In: *NeurIPS 2018 Workshop on Privacy Preserving Machine Learning*. 2018.

[Mac+07] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkitasubramaniam. “l-diversity: Privacy beyond k-anonymity”. In: *ACM Transactions on Knowledge Discovery from Data (TKDD)* 1.1 (2007), 3–es.

[Mal57] Colin L Mallows. “Non-null ranking models. I”. In: *Biometrika* 44.1/2 (1957), pp. 114–130.

[MC89] Gary L Miller and Bradley W Carroll. “Modeling vertebrate dispersal distances: alternatives to the geometric distribution”. In: *Ecology* 70.4 (1989), pp. 977–986.

[McK+18] Ryan McKenna, Gerome Miklau, Michael Hay, and Ashwin Machanavajjhala. “Optimizing error of high-dimensional statistical queries under differential privacy”. In: *Proceedings of the VLDB Endowment* 11.10 (2018).

[McK+22] Ryan McKenna, Brett Mullins, Daniel Sheldon, and Gerome Miklau. “AIM: An adaptive and iterative mechanism for differentially private synthetic data”. In: *Proc. VLDB Endow.* 15.11 (2022), 2599–2612. ISSN: 2150-8097. DOI: 10.14778/3551793.3551817.

[McM+18] H. Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. “Learning differentially private recurrent language models”. In: *International Conference on Learning Representations*. 2018.

[McS09] Frank D McSherry. “Privacy integrated queries: an extensible platform for privacy-preserving data analysis”. In: *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*. 2009, pp. 19–30.

[Mir17] Ilya Mironov. “Rényi differential privacy”. In: *Computer Security Foundations Symposium (CSF), 2017 IEEE 30th*. IEEE. 2017, pp. 263–275.

[MN12] Shanmugavelayutham Muthukrishnan and Aleksandar Nikolov. “Optimal private halfspace counting via discrepancy”. In: *Proceedings of the Forty-fourth Annual ACM Symposium on Theory of Computing*. 2012, pp. 1285–1292.

[Moč75] J Močkus. “On Bayesian methods for seeking the extremum”. In: *Optimization Techniques IFIP Technical Conference*. Springer. 1975, pp. 400–404.

[Moh+22] Shubhankar Mohapatra, Sajin Sasy, Xi He, Gautam Kamath, and Om Thakkar. “The role of adaptive optimizers for honest private hyperparameter selection”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 36. 7. 2022, pp. 7806–7813.

[MRT12] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. MIT press, 2012.

[MSM19] Ryan McKenna, Daniel Sheldon, and Gerome Miklau. “Graphical-model based estimation and inference for differential privacy”. In: *International Conference on Machine Learning*. PMLR. 2019, pp. 4435–4444.

[Nik22] Aleksandar Nikolov. “Private query release via the Johnson-Lindenstrauss transform”. In: *arXiv preprint arXiv:2208.07410* (2022).

[Nix+22] Michelle Nixon, Andres Barrientos, Jerome Reiter, and Aleksandra Slavkovic. “A latent class modeling approach for differentially private synthetic data for contingency tables”. In: *Journal of Privacy and Confidentiality* 12.1 (2022).

[Noc+16] Richard Nock, Raphaël Canyasse, Roksana Boreli, and Frank Nielsen. “k-variates++: more pluses in the k-means++”. In: *International Conference on Machine Learning (ICML)*. 2016, pp. 145–154.

[NRS07] Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. “Smooth sensitivity and sampling in private data analysis”. In: *Proceedings of the Thirty-ninth Annual ACM Symposium on Theory of Computing*. 2007, pp. 75–84.

[NS08] Arvind Narayanan and Vitaly Shmatikov. “Robust de-anonymization of large sparse datasets”. In: *2008 IEEE Symposium on Security and Privacy (sp 2008)*. IEEE. 2008, pp. 111–125.

[NS18] Kobbi Nissim and Uri Stemmer. “Clustering algorithms for the centralized and local models”. In: *Proceedings of Algorithmic Learning Theory*. Ed. by Firdaus Janoos, Mehryar Mohri, and Karthik Sridharan. Vol. 83. Proceedings of Machine Learning Research. PMLR, 2018, pp. 619–653. URL: http://proceedings.mlr.press/v83/nissim18a.html.

[NTS14] Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. “In search of the real inductive bias: On the role of implicit regularization in deep learning”. In: *arXiv preprint arXiv:1412.6614* (2014).

[NTZ13] Aleksandar Nikolov, Kunal Talwar, and Li Zhang. “The geometry of differential privacy: The sparse and approximate cases”. In: *Proceedings of the 45th Annual ACM Symposium on Theory of computing*. 2013, pp. 351–360.

[OVL18] Kensuke Okada, Joachim Vandekerckhove, and Michael D Lee. “Modeling when people quit: Bayesian censored geometric models with hierarchical and latent-mixture extensions”. In: *Behavior research methods* 50.1 (2018), pp. 406–415.

[Pap+17] Nicolas Papernot, Martín Abadi, Úlfar Erlingsson, Ian J. Goodfellow, and Kunal Talwar. “Semi-supervised knowledge transfer for deep learning from private training data”. In: *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. 2017. URL: https://openreview.net/forum?id=HkwoSDPgg.

[Pap19] Nicolas Papernot. “Machine learning at scale with differential privacy in {TensorFlow}”. In: *2019 {USENIX} Conference on Privacy Engineering Practice and Respect ({PEPR} 19)*. 2019.

[Pih+22] Vasyl Pihur, Aleksandra Korolova, Frederick Liu, Subhash Sankuratripati, Moti Yung, Dachuan Huang, and Ruogu Zeng. “Differentially-private “draw and discard” machine learning: Training distributed model from enormous crowds”. In: *International Symposium on Cyber Security, Cryptology, and Machine Learning*. Springer. 2022, pp. 468–486.

[Pla75] Robin L Plackett. “The analysis of permutations”. In: *Journal of the Royal Statistical Society Series C: Applied Statistics* 24.2 (1975), pp. 193–202.

[Pop35] Tiberiu Popoviciu. “Sur les équations algébriques ayant toutes leurs racines réelles”. In: *Mathematica* 9 (1935), pp. 129–145.

[PS] Nicolas Papernot and Thomas Steinke. “Hyperparameter tuning with Renyi differential privacy”. In: *International Conference on Learning Representations*.

[PS22] Nicolas Papernot and Thomas Steinke. “Hyperparameter tuning with Renyi differential privacy”. In: *International Conference on Learning Representations*. 2022. URL: https://openreview.net/forum?id=-70L8lpp9DF.

[Qin+16] 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*. 2016, pp. 192–203.

[RM51] Herbert Robbins and Sutton Monro. “A stochastic approximation method”. In: *The Annals of Mathematical Statistics* (1951), pp. 400–407.

[Rod21] Rolando A. Rodríguez. “Disclosure avoidance and the American community survey”. 2021 ACS Data Users Conference. 2021. URL: https://acsdatacommunity.prb.org/m/2021-acs-conference-files/147/download.

[RR10] Aaron Roth and Tim Roughgarden. “Interactive privacy via the median mechanism”. In: *Proceedings of the 42nd ACM Symposium on Theory of Computing*. 2010, pp. 765–774.

[RW05] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian processes for machine learning (adaptive computation and machine learning)*. The MIT Press, 2005. ISBN: 026218253X.

[SGR04] Edward Snelson, Zoubin Ghahramani, and Carl E Rasmussen. “Warped Gaussian processes”. In: *Advances in Neural Information Processing Systems*. 2004, pp. 337–344.

[Shi+17] Elaine Shi, T-H Hubert Chan, Eleanor Rieffel, and Dawn Song. “Distributed private data analysis: Lower bounds and practical constructions”. In: *ACM Transactions on Algorithms (TALG)* 13.4 (2017), pp. 1–38.

[Sil10] Fabrizio Silvestri. “Mining query logs: Turning search usage data into knowledge”. In: *Foundations and Trends in Information Retrieval* 4.1–2 (2010), pp. 1–174.

[SLA12] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. “Practical Bayesian optimization of machine learning algorithms”. In: *Advances in neural information processing systems*. 2012, pp. 2951–2959.

[Smi+18] Michael Smith, Mauricio Álvarez, Max Zwiessele, and Neil Lawrence. “Differentially private regression with Gaussian processes”. In: *International Conference on Artificial Intelligence and Statistics*. 2018, pp. 1195–1203.

[SON95] Ashok Savasere, Edward Robert Omiecinski, and Shamkant B Navathe. *An efficient algorithm for mining association rules in large databases*. Tech. rep. Georgia Institute of Technology, 1995.

[SS18] Joshua Snoke and Aleksandra Slavković. “pMSE mechanism: Differentially private synthetic data with maximal distributional similarity”. In: *International Conference on Privacy in Statistical Databases*. Springer. 2018.

[SS98] Pierangela Samarati and Latanya Sweeney. “Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression”. In: (1998).

[SSBD14] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge University Press, 2014.

[Ste20] Uri Stemmer. “Locally private k-means clustering”. In: *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM. 2020, pp. 548–559.

[STU17] Adam Smith, Abhradeep Thakurta, and Jalaj Upadhyay. “Is interaction necessary for distributed private learning?” In: *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2017, pp. 58–77.

[Su+16] Dong Su, Jianneng Cao, Ninghui Li, Elisa Bertino, and Hongxia Jin. “Differentially private k-means clustering”. In: *Proceedings of the Sixth ACM Conference on Data and Application Security and Privacy*. 2016, pp. 26–37.

[SU17] Thomas Steinke and Jonathan Ullman. “Tight lower bounds for differentially private selection”. In: *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2017, pp. 552–563.

[SVK21] Pranav Subramani, Nicholas Vadivelu, and Gautam Kamath. “Enabling fast differentially private sgd via just-in-time compilation and vectorization”. In: *Advances in Neural Information Processing Systems* 34 (2021).

[Swe97] Latanya Sweeney. “Guaranteeing anonymity when sharing medical data, the Datafly system”. In: *Proceedings of the AMIA Annual Fall Symposium*. American Medical Informatics Association. 1997, p. 51.

[SZY19] Lin Sun, Jun Zhao, and Xiaojun Ye. “Distributed clustering in the anonymized space with local differential privacy”. In: *arXiv preprint arXiv:1906.11441* (2019).

[Tan+17] Jun Tang, Aleksandra Korolova, Xiaolong Bai, Xueqiang Wang, and XiaoFeng Wang. “Privacy Loss in Apple’s implementation of differential privacy on MacOS 10.12”. In: *arXiv preprint arXiv:1709.02753* (2017). https://arxiv.org/abs/1709.02753.

[Tao+21] Yuchao Tao, Ryan McKenna, Michael Hay, Ashwin Machanavajjhala, and Gerome Miklau. “Benchmarking differentially private synthetic data generation algorithms”. In: *arXiv preprint arXiv:2112.09238* (2021).

[TBM21] Yuchao Tao, Johes Bater, and Ashwin Machanavajjhala. “Prior-aware distribution estimation for differential privacy”. In: *arXiv preprint arXiv:2106.05131* (2021).

[Tea17] Apple Differential Privacy Team. “Learning with privacy at scale”. In: vol. 1. https://machinelearning.apple.com/research/learning-with-privacy-at-scale. Apple Machine Learning Journal, 2017.

[Toi+96] Hannu Toivonen et al. “Sampling large databases for association rules”. In: *VLDB*. Vol. 96. 1996, pp. 134–145.

[TUV12] Justin Thaler, Jonathan Ullman, and Salil Vadhan. “Faster algorithms for privately releasing marginals”. In: *International Colloquium on Automata, Languages, and Programming*. Springer. 2012, pp. 810–821.

[Ull13] Jonathan Robert Ullman. “Privacy and the complexity of simple queries”. PhD thesis. Harvard University, 2013.

[Ull16] Jonathan Ullman. “Answering nˆ2+o(1) counting queries with differential privacy is hard”. In: *SIAM Journal on Computing* 45.2 (2016), pp. 473–496.

[UV11] Jonathan Ullman and Salil Vadhan. “PCPs and the hardness of generating private synthetic data”. In: *Theory of Cryptography Conference*. Springer. 2011, pp. 400–416.

[Val+09] Hamed Valizadegan, Rong Jin, Ruofei Zhang, and Jianchang Mao. “Learning to rank by optimizing ndcg measure”. In: *Advances in Neural Information Processing Systems* 22 (2009).

[Vap98] Vladimir Vapnik. “The support vector method of function estimation”. In: *Nonlinear Modeling*. Springer, 1998, pp. 55–85.

[Vap99] Vladimir N Vapnik. “An overview of statistical learning theory”. In: *IEEE Transactions on Neural Networks* 10.5 (1999), pp. 988–999.

[Vee18] Koen Lennart van der Veen. “A practical approach to differential private learning”. MA thesis. University of Amsterdam, The Netherlands, 2018.

[Vie+20] Giuseppe Vietri, Grace Tian, Mark Bun, Thomas Steinke, and Steven Wu. “New oracle-efficient algorithms for private synthetic data release”. In: *International Conference on Machine Learning*. PMLR. 2020, pp. 9765–9774.

[Vie+22] Giuseppe Vietri, Cedric Archambeau, Sergul Aydore, William Brown, Michael Kearns, Aaron Roth, Ankit Siva, Shuai Tang, and Zhiwei Steven Wu. “Private synthetic data for multitask learning and marginal queries”. In: *arXiv preprint arXiv:2209.07400* (2022).

[Wan+21] Di Wang, Huangyu Zhang, Marco Gaboardi, and Jinhui Xu. “Estimating smooth glm in non-interactive local differential privacy model with public unlabeled data”. In: *Algorithmic Learning Theory*. PMLR. 2021, pp. 1207–1213.

[War65] Stanley L Warner. “Randomized response: A survey technique for eliminating evasive answer bias”. In: *Journal of the American Statistical Association* 60.309 (1965), pp. 63–69.

[WBK19] Yu-Xiang Wang, Borja Balle, and Shiva Kasiviswanathan. “Subsampled Rényi differential privacy and analytical moments accountant”. In: *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS)*. 2019.

[WCP13] Weiran Wang and Miguel A Carreira-Perpinán. “Projection onto the probability simplex: An efficient algorithm with a simple proof, and an application”. In: *arXiv preprint arXiv:1309.1541* (2013).

[Wil+20] Royce J Wilson, Celia Yuxin Zhang, William Lam, Damien Desfontaines, Daniel Simmons-Marengo, and Bryant Gipson. “Differentially private SQL with bounded user contribution”. In: *Proceedings on Privacy Enhancing Technologies* 2020.2 (2020), pp. 230–250.

[Woo+18] Alexandra Wood, Micah Altman, Aaron Bembenek, Mark Bun, Marco Gaboardi, James Honaker, Kobbi Nissim, David R O’Brien, Thomas Steinke, and Salil Vadhan. “Differential privacy: A primer for a non-technical audience”. In: *Vanderbilt Journal of Entertainment and Technology Law* 21 (2018), p. 209.

[Wu+17] Xi Wu, Fengan Li, Arun Kumar, Kamalika Chaudhuri, Somesh Jha, and Jeffrey Naughton. “Bolt-on differential privacy for scalable stochastic gradient descent-based analytics”. In: *Proceedings of the 2017 ACM International Conference on Management of Data*. ACM. 2017, pp. 1307–1322.

[Xia+20] Chang Xia, Jingyu Hua, Wei Tong, and Sheng Zhong. “Distributed k-means clustering guaranteeing local differential privacy”. In: *Computers & Security* 90 (2020), p. 101699.

[XSM16] Sijie Xiong, Anand D Sarwate, and Narayan B Mandayam. “Randomized requantization with local differential privacy”. In: *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2016, pp. 2189–2193.

[You+21] Ashkan Yousefpour, Igor Shilov, Alexandre Sablayrolles, Davide Testuggine, Karthik Prasad, Mani Malek, John Nguyen, Sayan Ghosh, Akash Bharadwaj, Jessica Zhao, Graham Cormode, and Ilya Mironov. “Opacus: User-Friendly Differential Privacy Library in PyTorch”. In: *arXiv preprint arXiv:2109.12298* (2021).

[Yu+04] Jeffrey Xu Yu, Zhihong Chong, Hongjun Lu, and Aoying Zhou. “False positive or false negative: Mining frequent itemsets from high speed transactional data streams”. In: *VLDB*. Vol. 4. 2004, pp. 204–215.

[Zen+12] Jianping Zeng, Jiangjiao Duan, Wenjun Cao, and Chengrong Wu. “Topics modeling based on selective Zipf distribution”. In: *Expert Systems with Applications* 39.7 (2012), pp. 6541–6546.

[Zha+21] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. “Understanding deep learning (still) requires rethinking generalization”. In: *Communications of the ACM* 64.3 (2021), pp. 107–115.

[ZKP16] Marcela Zuluaga, Andreas Krause, and Markus Püschel. “ε-pal: An active learning approach to the multi-objective optimization problem”. In: *The Journal of Machine Learning Research* 17.1 (2016), pp. 3619–3650.