BAYESIAN METHODS FOR TACKLING COMPLEX INFERENTIAL PROBLEMS IN DATA SCIENCE

by
SONER AYDIN

Submitted to the Graduate School of Social Sciences
in partial fulfillment of
the requirements for the degree of Doctor of Philosophy

Sabancı University
December 2024

ABSTRACT

BAYESIAN METHODS FOR TACKLING COMPLEX INFERENTIAL PROBLEMS IN DATA SCIENCE

SONER AYDIN

INDUSTRIAL ENGINEERING Ph.D DISSERTATION, DECEMBER 2024

Dissertation Supervisor: DR SİNAN YILDIRIM

Keywords: hyperparameter tuning, posterior sampling, differential privacy, local differential privacy, adaptive online frequency estimation, robust regression

Bayesian methods encompass a principled way of modeling, solving and analyzing various estimation and inference problems in data science. In this dissertation, we utilize a variety of Bayesian methods, such as posterior sampling, EM algorithm for mixture models, subsampling for prior probability estimation, to tackle a wide range of inferential problems. These problems include hyperparameter tuning in regularized linear models in supervised learning, robust regression, frequency estimation for dynamic/online datasets under global and local differential privacy frameworks. For each of these problems, we propose new algorithms that can compete with the existing approaches in terms of estimation accuracy, while performing these tasks in a computationally more efficient way via utilizing sampling and subsampling. Along with each algorithm, we also provide both theoretical analyses and numerical experiments that demonstrate their estimation performance.

ÖZET

VERİ BİLİMİNDEKİ KARMAŞIK ÇIKARIMSAL SORUNLARI ÇÖZMEK İÇİN BAYES YÖNTEMLERİ

SONER AYDIN

ENDÜSTRİ MÜHENDİSLİĞİ DOKTORA TEZİ, ARALIK 2024

Tez Danışmanı: Dr. SİNAN YILDIRIM

Anahtar Kelimeler: hiperparametre ayarı, arka örnekleme, diferansiyel mahremiyet, yerel diferansiyel mahremiyet, uyarlanabilir çevrimiçi frekans tahmini, gürbüz regresyon

Bayes yöntemleri, veri bilimindeki çeşitli tahmin ve çıkarım problemlerini modelleme, çözme ve analiz etmede ilkeli bir yolu kapsar. Bu tezde, çok çeşitli çıkarımsal problemleri ele almak için, arka örnekleme, karışım modelleri için EM algoritması, ön olasılık tahmini için alt örnekleme gibi çeşitli Bayes yöntemlerini kullanıyoruz. Bu problemler arasında, denetlenen öğrenmede düzenlenmiş doğrusal modellerde hiperparametre ayarlama, gürbüz regresyon, küresel ve yerel diferansiyel mahremiyet çerçeveleri altında dinamik/çevrimiçi veri kümeleri için frekans tahmini yer alır. Bu problemlerin her biri için, örnekleme ve alt örneklemeyi kullanarak bu görevleri hesaplama açısından daha verimli bir şekilde gerçekleştirirken, tahmin doğruluğu açısından mevcut yaklaşımlarla rekabet edebilen yeni algoritmalar öneriyoruz. Her algoritmayla birlikte, tahmin performanslarını gösteren hem teorik analizler hem de sayısal deneyler sağlıyoruz.

ACKNOWLEDGEMENTS

First of all, I would like to thank my thesis advisor Sinan Yıldırım for his endless support throughout this research journey. I have learned a lot of new technical and practical skills thanks to him.

I thank all the jury members who spared their time and energy for reading, watching and evaluating my thesis.

I thank my parents who raised me and supported me throughout my whole life with their love and kindness. I wouldn't be here without them.

Special thanks to my grandfather, Aptilazim AYDIN. He had been a great role model for me since my childhood years. May he rest in peace...

To Baruch Spinoza...
...an unsung hero of the Age of Enlightenment

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

1.1 Overview

Bayesian statistical paradigm offers a unified framework for modeling and solving a vast number of different problems in data science, by encouraging researchers and practitioners alike to treat a given problem in terms of probabilistic models. This general approach allows developing computationally efficient algorithms for a given task, without sacrificing the accuracy and interpretability of the results. It also allows quantifying the uncertainty of parameters and hyperparameters in these models in a principled way. In this dissertation, we take advantage of this general framework to develop such models for a wide variety of tasks. Additionally, Bayesian models naturally lend themselves to implementation of sampling and subsampling-based methods which we use heavily throughout this dissertation. This is the main aspect that makes these methods computationally efficient in application areas that involve large-scale datasets.

In this dissertation, we propose various Bayesian methods for tackling estimation problems and inferential tasks in a diverse range of data science problems, including hyperparameter tuning in regularized supervised learning methods, robust estimation of regression parameters, online frequency estimation in global and local differential privacy settings.

For each of these tasks, having their own dedicated chapters, we provide theoretical justifications/proofs, numerical experiments to demonstrate their accuracy, compare them with the existing approaches, discuss their potential applications in real-life scenarios, state our contributions, and make suggestions for future directions of research.

1.2 Technical Literature Review

First of all, let us introduce the basic terminology and some of the existing methods that are related to the research topics that we cover in this dissertation. More detailed treatment of these elements are provided in each chapter.

Hyperparameter tuning: In the general sense, hyperparameter tuning refers to fine-tuning of hyperparameters for a given machine learning model. However, in the scope of this research, we focus on the tuning of regularization hyperparameters. Regularization was first introduced in 1970 as a solution to the multicollinearity problem in regression models (Hoerl & Kennard, 1970). In its most basic form, it is based on adding a penalty term that penalizes the norm of model parameters, as in the following ridge regression problem

$$\min_{\beta \in \mathbb{R}^{d\times 1}} (Y - X\beta)^2 + \lambda ||\beta||_2^2,$$

where λ is the hyperparameter to be tuned. Later on, it has grown into a general methodology for controlling model complexity in other machine learning models by introducing sparsity into model parameters (Hastie, Tibshirani & Wainwright, 2015). In real-life applications, the most popular way of tuning this hyperparameter is based on grid-search along with cross-validation (CV), *i. e.*, trying different λ values from a specified range and computing their CV error, and choosing the one that yields the smallest CV error. However, CV is not the only method that is used in this field.

Methods that are based on CV (k-fold CV, leave-one-out CV, nested CV) can be efficient only when the goal is to find a single or a few number of hyperparameters. However, in real applications, there can be multiple hyperparameters which penalize more than one penalty terms, or they can be in vector form in which each element of the vector penalizes a different component of β. In these cases, CV-based methods become intractable. Another way of casting this problem is to think of these hyperparameters as *precision* terms (reciprocal of variance) for the prior distribution of β. When the hyperparameter tuning problem is cast as a probabilistic model like this, Bayesian methods are a natural choice to tackle this problem. For example, approaches like Markov chain Monte Carlo (MCMC) sampling (Xiang, 2020), (Chaari, Batatia, Dobigeon & Tourneret, 2014), Bayesian Optimization, (Kochenderfer & Wheeler, 2019), Expectation-Maximization Lin & Lee (2006), simulated annealing (Kuhn & Silge, 2022) have been proposed to address this problem

in Bayesian framework. Even though these approaches have provided significant improvements for simultaneous estimation of multiple hyperparameters, they are still based on using the whole training dataset for a given model, and this still poses computational challenges while using large datasets. We address this problem by proposing a subsampling-based algorithm to tune multivariate hyperparameters for linear models in supervised learning. Our algorithm is based on obtaining a quick estimate of prior mean and covariance hyperparameters of β by utilizing subsamples from a given training data. The details of this approach are given in Chapter 2.

Robust regression: Standard linear regression model assumes that the residuals are distributed according to normal distribution and their variance is constant for all data points for a given dataset. Howeveri in many real-life applications, practitioners often come across datasets that have unusual noise patterns that are ridden with outliers and high skewness. When one fits ordinary least-squares (OLS) to such a dataset, it causes a drastic ddterioration in the predictive performance of linear regression. A common way of remedying this issue is to replace the quadratic loss function (MSE) of OLS with another one which is more resilient to outliers and skewness. In the literature, there is a large family of estimators, called *m-estimators*, which contain mean-squared error, least absolute deviation, Huber loss and many other symmetric, positive and continuous loss functions as its special cases (De Menezes, Prata, Secchi & Pinto, 2021). Even though MSE is also an m-estimator, we will focus on its robust members.

The term *robust regression* denotes a family of regression methods that are not too sensitive to outliers and skewness in residuals. It is also deeply connected to the regularized regression problem that we address in Chapter 2. When one applies a regularization method on regression parameters, one implicitly makes them more robust against outliers by reducing the variance of these parameters. Regularization is achieved by penalizing the irrelevant features (columns) of the predictor matrix, whereas robustness is achieved by penalizing the observations (rows) which have less predictive power. As we will see in Chapter 3, fitting an m-estimator is achieved by finding optimal weights to penalize these "peculiar" observations.

For a single m-estimator, finding the optimal weights for observations is often handled by using *iteratively reweighted least squares* (IRLS) procedure (Susanti, Pratiwi, Sulistijowati, Liana & others, 2014), but in the case of a mixture of m-estimators it requires more complicated methods. Additionally, using a single m-estimator may not always be sufficient to capture the characteristics of the stochastic process that generates a given dataset. In the literature, there are some other uses of mixture models (mixture of m-estimators) to fit robust regression Bai, Yao &

Boyer (2012), Tak, Ellis & Ghosh (2019). However, the methods that we have observed are either not flexible enough or not efficient enough to incorporate more than two m-estimators to the mixture model. For these reasons, we propose an efficient Expectation-Maximization algorithm to fit a mixture of m-estimators for a given dataset.

Differentially Private Sketches: *Differential privacy* (DP) is a general framework that allows answering queries about a given dataset while preserving the privacy of the individuals whose personal information is in that dataset Dwork (2006a). Via randomly perturbing the answer of a query, or via perturbing each individuals' data, it provides "plausible deniability" for the individuals when a malevolent analyst is trying to recover their personal information from the answers to specific queries about the population statistics. We say that $\mathcal{A}$ is (ϵ, δ)-differentially private if, for any pair of neighboring data sets $X, X' \in \mathsf{X}$ from an input set and any subset of output values $S \subseteq \mathcal{S}$, it satisfies

$$\mathbb{P}\left[\mathcal{A}(X) \in S\right] \leq e^{\epsilon}\mathbb{P}\left[\mathcal{A}(X') \in S\right] + \delta.$$

According to the above inequality, a randomized algorithm is differentially private if the probability distributions for the output obtained from two neighboring databases are '*similar*'. The parameters ϵ and δ determine the *privacy budget*, or *privacy loss*. Those parameters are desired to be as small as possible as far as privacy is concerned. In the DP framework, the trade-off between accuracy in responses and privacy guarantees is often balanced by finding an optimal randomization mechanism that is crafted according to the sensitivity of the query responses. In the applications that involve data streams, *sketches* are often used as inherently randomized data structures that can provide approximate population statistics within mathematically proven error bounds, while using much less memory than traditional data-keeping approaches. Their inherently randomized structure makes them well-suited for implementations that require DP guarantees. For this reason, utilizing DP for data sketches have been a popular approach in the literature in the last decades Cormode, Procopiuc, Srivastava & Tran (2012); Dwork, Naor, Pitassi, Rothblum & Yekhanin (2010); Melis, Danezis & Cristofaro (2016); Mir, Muthukrishnan, Nikolov & Wright (2011); Mishra & Sandler (2006); Sparka, Tschorsch & Scheuermann (2018); von Voigt & Tschorsch (2019). Our study is confined to only the Count and Count-Min Sketches Charikar, Chen & Farach-Colton (2002); Cormode & Muthukrishnan (2005). The Count Sketch is proposed in Charikar et al. (2002) as a useful tool for answering frequency queries, by producing unbiased estimators. Similarly, the Count-Min Sketch Cormode & Muthukrishnan (2005) is proposed for the same task.

In Chapter 4, we tackle the problem of responding to intermittent frequency queries about the information contained in a data stream while providing DP guarantees and keeping the utility of the responses at a reasonable level, *i.e.*, reducing the potential deterioration of accuracy that is caused by accumulation of random noise. The thing that is significant about the setting of "intermittent queries" is the possibility that even between two consecutive *identical* queries, the true answer may have differed due to the addition of new individuals' data. The challenge related to data privacy is that the answers to the queries should continually protect the privacy of individuals' data that are included in the data stream at any time of the streaming process. Hence, the setting being investigated in this work can be considered as a generalization of the setting which focuses on one-time queries.

Adaptive Online Bayesian Frequency Estimation Under Local Differential Privacy: Now, we will introduce a slightly different notion of differential privacy than the previous one, namely *local differential privacy* (LDP).

Definition 1 (Local differential privacy)**.** *A randomized mechanism $\mathcal{M} : \mathcal{X} \mapsto \mathcal{Y}$ satisfies ϵ-LDP if the following inequality holds for any pairs of inputs $x, x' \in \mathcal{X}$, and for any output (response) $y \in \mathcal{Y}$:*

$$e^{-\epsilon} \leq \frac{\mathbb{P}(\mathcal{M}(x) = y)}{\mathbb{P}(\mathcal{M}(x') = y)} \leq e^{\epsilon}.$$

This definition of ϵ-LDP is almost the same as that of global ϵ-DP. The main difference is that, in the global DP, inputs x, x' are two datasets that differ in only one individual's record, whereas in LDP, x, x' are two different data points from $\mathcal{X}$. In Definition 4, $\epsilon \geq 0$ is the privacy parameter. A smaller ϵ value provides stronger privacy.

Suppose a type of sensitive information is represented as a random variable X with a categorical distribution denoted by $\text{Cat}(\theta)$, where θ is a K-dimensional probability vector. Our goal is to estimate this parameter as accurately as possible, while satisfying the LDP constraint. For this purpose, in Chapter 5, we propose an adaptive and online algorithm to estimate θ in a *Local Differential Privacy* (LDP) framework where X is unobserved and instead, we have access to a randomized response Y derived from X. In the LDP framework, a central aggregator receives each user's randomized (privatized) data to be used for inferential tasks. In that sense, LDP differs from global DP (Dwork, 2006b) where the aggregator privatizes operations on the sensitive dataset after it collects the sensitive data without noise. Hence LDP can be said to provide a stricter form of privacy and is used in cases where the aggregator may not be trustable (Kasiviswanathan, Lee, Nissim, Raskhodnikova &

Smith, 2011).

The main challenge in most differential privacy settings is to decide how to select a randomized mechanism. In the case of LDP, this is cast as how an individual data point X should be randomized. On top of that, in many cases, individuals' data points are collected sequentially. A basic example is opinion polling, where data is collected typically in time intervals of lengths in the order of hours or days. While sequential collection of individual data may make the estimation task under the LDP constraint harder, it may also offer an opportunity to adapt the randomized mechanism in time to improve the estimation quality. Motivated by that, in Chapter 5, we address the problem of online Bayesian estimation of a categorical distribution (θ) under ϵ-LDP, while at the same time choosing the randomization mechanism adaptively so that the utility is improved continually in time.

Our proposed algorithm, AdOBEst-LDP employs a new randomization mechanism, *randomly restricted randomized response* (RRRR). RRRR is a modified version of the *Standard Randomized Response* mechanism (SRR). This is a well-studied mechanism in the DP literature, and the statistical properties of its basic version (such as its estimation variance) can be found in the works by (Wang, Blocki, Li & Jha, 2017) and (Wang, Lopuhaä-Zwakenberg, Li, Skoric & Li, 2020). When the number of categories K in a dataset is large, the utility of SRR can be too low. RRRR in AdOBEst-LDP is designed to circumvent this problem by constraining its output to a subset of categories. Unlike SRR, the perturbation probability of responses in our algorithm changes adaptively, depending on the cardinality of the selected subset of categories for the privatization of X, and the cardinality of its complementary set. At each step of the algorithm, we select the subset of these categories with respect to a given utility function that measures the informativeness of the subset, given the posterior sample from the current estimate of the categorical density parameter $\hat{\theta}$. In other words, we choose the subset that maximizes the given utility function whose input is the posterior sample of the population parameter estimate. After adapting the randomized mechanism, we use it to privatize the newly arrived data. Next, this new data is used for updating the posterior sampling procedure of $\hat{\theta}$. To put it simply, both randomization and parameter estimation steps of the algorithm guide each other adaptively and continually. To quantify the utility of a given subset, we explore various well-known information metrics, including the Fisher information matrix, total variation distance, and information entropy. For the Bayesian estimation of parameter $\hat{\theta}$, we utilize *posterior sampling* through stochastic gradient Langevin dynamics (SGLD) which is a computationally efficient, approximate Markov chain Monte Carlo (MCMC) method.

1.3 Outline

The main material of this dissertation is organized in four chapters and a **Conclusion** chapter, while some of the technical details (additional proofs) of these chapters are deferred to **Appendices**. Here is a brief outline of the four main chapters:

Chapter 2: Hyperparameter Tuning in Linear Models
In this chapter, we propose a subsampling-based algorithm to handle hyperparameter tuning for regularized linear models in supervised learning. We compare the proposed algorithm with grid-search-based tuning of hyperparameters with cross-validation and unregularized solution of each linear model.

Chapter 3: A Bayesian Approach for Solving Robust Regression Problems
In this chapter, we tackle a subclass of regression problems where the response variable violates the basic assumptions of ordinary least-squares; namely, these response variables have outliers and non-normally distributed noise. For this purpose, we model the given problem as a mixture of m-estimators and propose an Expectation-Maximization algorithm to solve them.

Chapter 4: Differentially Private Frequency Sketches for Intermittent Queries on Large Data Streams
This chapter is a replication of a conference paper that we wrote in 2020, with minor editing. The author of this dissertation is also one of the co-authors of that paper[1]. In this chapter, we did not use a Bayesian method, but it is still an important part of this dissertation, in that, it contains our earlier excursions into the field of online frequency estimation under differential privacy (DP), and is partially related to the topic of the next chapter.

Chapter 5: Bayesian Frequency Estimation Under LDP With an Adaptive Randomized Response Mechanism
This chapter is a replication of an article that we wrote in 2024, with minor additions and editing. Again, the author of this dissertation is also one of the co-authors of that paper[2]. Here, differently from the previous chapter, we tackle the frequency estimation problem under *local* differential privacy, as opposed to global DP, and we directly make use of Bayesian methods, such as posterior sampling.

---

[1]The conference paper can be accessed via this link: https://ieeexplore.ieee.org/document/9377786

[2]The article can be accessed via this link: https://dl.acm.org/doi/10.1145/3706584

2. HYPERPARAMETER TUNING IN LINEAR MODELS

We propose a new subsampling-based algorithm for tuning hyperparameters in ℓ_2-norm regularized learning models. Our algorithm draws random subsets from the training dataset and fits the unregularized version of a given model on each one of these subsets. The solutions obtained from those subsets are used to construct a regularization term which, in a Bayesian context, corresponds to a multivariate Gaussian prior for the model parameter vector. This regularization term is then applied to the whole dataset. We applied our algorithm to several well-known supervised learning models and tested it on real datasets. Our experiments show that the test set accuracy of our algorithm is on par with that of the famous k-fold cross-validation while its computation time is significantly shorter.

Most of the research in this chapter was conducted by the author of this dissertation. Supplementary derivations in Appendix A.1 were made by Sinan Yıldırım. Additionally, İlker Birbil provided useful suggestions during our meetings on this topic.

2.1 Introduction

In this chapter, our focus is on the computationally efficient tuning of regularization hyperparameters in linear models [1] for supervised learning. First, we will explain and demonstrate our method for ridge regularization in linear regression. Later, we will show that our method for tuning these hyperparameters can be extended to other generalized linear models used for regression and classification.

Regularization hyperparameters are widely used for controlling the complexity of

[1] The method that we propose is also applicable to nonlinear models, but for convenience, we will focus on its application to linear models here.

regression models (and other supervised learning models), in order to prevent the overfitting problem and to maximize the generalization performance of these methods. These parameters are often tuned by brute-force search, using cross-validation (CV) in practice; in other words, many of the possible hyperparameter values in a predetermined range are exhaustively tested with CV (Hastie, Tibshirani & Wainwright, 2019). However, CV can be computationally expensive when there are multiple hyperparameters and even more so when these hyperparameters are vectors rather than scalars. Our motivation is to develop a method that can tune these multidimensional hyperparameters efficiently, approximating their optimal values in a short time.

Our method is primarily designed to regularize the ℓ_2-norm of the parameters of *generalized linear models* (GLM), such as linear regression, logistic regression, and Poisson regression. Here is the general form of the ℓ_2-norm regularization problem. Given the collections of response values $Y = (y_1, \ldots, y_N)$, and the predictor matrix $X = (x_1, \ldots, x_N)$ of sizes N, where for each observation i we have $y_i \in \mathcal{Y} \subseteq \mathbb{R}$ and row vector $x_i \in \mathcal{X} \subseteq \mathbb{R}^d$, the optimization task in a generalized linear model with ℓ_2-norm regularization problem can be stated as

$$\min_{\beta \in \mathbb{R}^{d \times 1}} \sum_{i=1}^{N} \mathcal{L}(y_i, g(x_i\beta)) + \lambda ||\beta||_2^2, \tag{2.1}$$

where $\mathcal{L} : \mathcal{Y}^2 \mapsto \mathbb{R}$ is a loss function such that $\mathcal{L}(y, \hat{y})$ measures the discrepancy between the response variable y and its 'fitted' value $\hat{y}$. The second term in (2.1) is the regularization term that penalizes the irrelevant components of the model parameters. In the case of GLMs, for a pair (x, y), the response variable y is approximated by a *link function* $g : \mathbb{R} \mapsto \mathcal{Y}$ that takes the linear combination $x_i\beta$ as input, and applies some transformation on this input, depending on the probabilistic model that relates y and x (Dobson & Barnett, 2018). The simplest example of a regularized GLM is the ridge regression problem where the optimization task is

$$\min_{\beta} \sum_{i=1}^{N} (y_i - x_i\beta)^2 + \lambda ||\beta||_2^2.$$

Here, the hyperparameter λ is to be tuned to approximate the optimal generalization performance (*i.e.*, the fitted parameter will yield high prediction accuracy on any unseen test datasets). CV is the most popular way of tuning this hyperparameter. Other alternatives are based on randomized search methods (such as Bayesian Optimization) that walk through the search space of λ's in a more principled way than grid search (Bischl, Binder, Lang, Pielok, Richter, Coors, Thomas, Ullmann, Becker, Boulesteix & others, 2023), and approximate the optimal solution faster

than CV. In any case, optimization of the exact generalization performance function is computationally intractable because it is based on averaging the performance functions of all possible combinations of train-test splits of the available dataset. We will cover the details of this problem in Section 2.3.

Since the regularization problem that we have just introduced is equivalent to the estimation of *maximum a posteriori (MAP)*, we can restate the regularization problem as finding a reasonable prior that can reduce the variance of β without introducing too much bias. We tackle this problem by drawing small subsamples from the training set, fitting an unregularized linear model to each of these subsamples, and obtaining separate β estimates from each of them. After that, we use the mean vector and covariance matrix of these subsampled βs as the prior mean and covariance of the given regularized linear model. We will elaborate on the details of how we used these priors as hyperparameters in Section 2.3. Due to our use of small subsamples to estimate the hyperparameters, our approach yields a fast approximation of the optimal hyperparameters, even when the number of hyperparameters is in the order of the square of the number of features.

2.2 Related Work

Hyperparameter tuning has been an active research topic since the 1970s after (Hoerl & Kennard, 1970) proposed a procedure to approximate the optimal penalty hyperparameter of the ridge regression problem. From then on, numerous approaches to tackle this problem have been developed for both Generalized Linear Models and more complicated machine learning (ML) methods. Due to the use of large-scale datasets in recent applications of ML methods and due to the computational intractability of finding their exact generalization performance, all of the proposed methods seek to find nearly optimal hyperparameters in a short time. We can classify these methods in mainly two categories: i) optimization-based methods and ii) sampling- and resampling-based methods.

Some of the existing optimization-based heuristics for tuning a single hyperparameter are surveyed in the work of (Qian, 2017) which are often based on striking a balance between the bias and variance components of the mean squared error. Methods based on CV (k-fold CV, leave-one-out CV, nested CV) are among the most popular methods in the category of resampling-based approaches, but these

methods can be efficient only when the goal is to find a few number of hyperparameters. When the hyperparameter tuning problem is cast as a hierarchical probability model, it naturally lends itself to various Bayesian approaches. For example, (Xiang, 2020) developed an MCMC method (which also falls under the category of sampling-based approaches) to estimate the hyperparameter of *Bridge* regression problem (which corresponds to penalizing the ℓ_p-norm of the regression parameter, where $1 < p < 2$) by sampling from the posterior distribution of both regression parameters and hyperparameters in a cyclic fashion. Similarly, (Chaari et al., 2014) developed a Gibbs sampler to tune the hyperparameters that penalize ℓ_0, ℓ_1-, and ℓ_2- norms of the regression parameter. Other well-known Bayesian methods used for solving this problem fall under the category of optimization-based approaches; such as *Bayesian Optimization* which is also known as *surrogate optimization* (Kochenderfer & Wheeler, 2019), and expectation-maximization algorithm as in the work of Lin & Lee (2006). The first of these is based on approximating the surface of the unknown objective function by a surrogate function and updating this surrogate function at each iteration by taking new samples from the search space. The second one is composed of two steps at each iteration: computing the expected values of the hyperparameters for a fixed regression parameter β (E-step), and maximizing the β for fixed hyperparameters (M-step). Another popular approach to optimize hyperparameters is to use *simulated annealing* algorithm (Kuhn & Silge, 2022) which falls under the categories of both optimization-based and sampling-based approaches. This is a generic randomized optimization method that is used for approximating the global optimal solution in a wide variety of nonconvex problems; but at the same time, it can also be classified as a subset of MCMC methods, since its implementation is very similar to Metropolis-Hastings algorithm (Robert, Casella & Casella, 2010).

In our work, we introduce a subsampling-based algorithm to tune multivariate hyperparameters, that is applicable not only for ridge regression but also for other linear models in supervised learning as well. In our computational experiments, this algorithm yields comparable results to the popular approaches (such as 10-fold CV) in a shorter time. In all of the existing approaches that we have encountered in the literature so far, the search space of hyperparameters grows exponentially with the dimension of the hyperparameter vector λ, thus demanding more computational time. Given this fact, our subsampling-based method provides a quick shortcut to estimate all of the hyperparameters simultaneously, even when Λ (multivariate analogue of λ) is an $d \times d$ matrix where d is the number of columns in X.

2.3 Methodology

As we briefly mentioned in Section 2.1, finding the exact generalization performance of a given regularized model is computationally intractable. Cross-validation is a widely used technique to estimate the generalization error for a given hyperparameter setting from the given data. The technique is based on splitting the available data into training and validation parts. Given some training data (X, y), and a hyperparameter λ, the ideal approximation of the theoretical generalization error by cross-validation based on this data can be given as

$$\binom{N}{N_s}^{-1} \sum_{\substack{(X_r,Y_r)\\(X_s,Y_s)}} \sum_{(x,y)\in(X_s,Y_s)} \mathcal{L}(y, g(x\hat{\beta}(X_r, y_r, \lambda))). \tag{2.2}$$

That is, one would like to consider all possible combinations of training-test partitions (X_r, y_r)-(X_s, y_s) of the available data (X, y), where for each partition the model is trained with (X_r, y_r) and tested against X_s, y_s. In (2.2), N, N_r, and N_s are the sizes of the whole dataset, training set, and test set, respectively, hence the term $\binom{N}{N_s}$ for the number of all possible partitions. The typically large number of terms in this summation in (2.2) makes its exact computation intractable. Minimization of this function is also challenging because it does not attain an exact analytical solution. For this reason, all of the existing hyperparameter tuning methods (including k-fold cross-validation) are essentially based on approximating this function (or its minimum point) in one way or another. Our proposed method to find approximately optimal hyperparameters is based on drawing random subsamples from the training set. The proposed method is displayed in Algorithm 1.

In Algorithm 1, we sample (without replacement) m subsamples of size n from the training set (X, y) of size N, where $n \ll N$, [2] and fit a given unregularized linear model to each of these subsamples [3] to estimate β_0 and $\hat{S}$ which are the prior mean and covariance, respectively. Finally, we use these prior hyperparameters by appending them as an ellipsoidal penalty term to the loss function which uses the whole training set as input, and solve it to obtain the MAP estimator $\hat{\beta}$.

[2] This assumption is important especially when N is large, so that using relatively small subsamples is more favorable in terms of computation time.

[3] We implemented this part of the method in a sequential manner, but the same part can be parallelized as well.

Algorithm 1 Training via subsampling for hyperparameter tuning

Input: Dataset: X, y, m: number of subsamples, n: subsample size
Output: Posterior mean parameter $\hat{\beta}$
for $i = 1 \ldots, m$ **do**
    `Sample` n `rows from the dataset` (X, y)`, to form` X_{sub} `and` y_{sub}`.`
    $\beta^{(i)} = \arg\min_{\beta} \sum_{(x,y)\in(X_{sub},y_{sub})} \mathcal{L}(y, g(x\beta))$

`Set` $\beta_0 \leftarrow \frac{1}{m}\sum_{i=1}^{m} \beta^{(i)}$
`Set` $\hat{S} \leftarrow \frac{1}{m-1}\sum_{i=1}^{m}(\beta^{(i)} - \beta_0)(\beta^{(i)} - \beta_0)^T$
`Find the final estimate`

$$\hat{\beta} \leftarrow \arg\min_{\beta} \sum_{i=1}^{N} \mathcal{L}(y_i, g(x_i\beta)) + (\beta - \beta_0)^T \hat{S}^{-1}(\beta - \beta_0)$$

return $\hat{\beta}$

2.3.1 Applications to Different Linear Models

In order to better explain how our algorithm works in practice, we will first discuss its application to linear regression. Later on, we will talk about other linear models that do not have analytical solutions (such as SVM, logistic regression, and Poisson regression) that we used in our paper. After that, we will provide a transformation method that facilitates the implementation of our method on models that are fitted by using the existing libraries which do not take multivariate hyperparameters as input.

2.3.1.1 Implemented Models

Linear Regression. In the standard form of ridge regression, it is assumed that the prior mean of β is a zero vector and the prior variance is unknown. When we use the sum of squared residuals as a special case of the loss function $\mathcal{L}(y, X\beta)$, as in linear regression, we have the following form:

$$\min_{\beta} f(\beta) = ||y - X\beta||_2^2 + (\beta - \beta_0)^T \hat{S}^{-1}(\beta - \beta_0),$$

where β_0 is the sample mean of the β's obtained from subsamples and $\hat{S}$ is a positive definite matrix that is constructed from the sample covariance of β's obtained from

the subsamples. The solution to the minimization problem above is given by

$$\hat{\beta} = (X^T X + \hat{S}^{-1})^{-1}(X^T y + \hat{S}^{-1}\beta_0),$$

which is the MAP solution when the prior for β is $\mathcal{N}(\beta_0, S)$. This formula can be interpreted as a weighted combination of the full-sample ordinary least squares (OLS) estimator and the prior β_0, where the contribution of β_0 to the overall estimate is proportional to the inverse of $\hat{S}$. As a special case, when $\hat{S}^{-1} = \lambda I$ and $\beta_0 = 0$, this expression boils down to the solution of standard ridge regression. But in the general case, $\hat{S}$ can be any positive definite matrix, and β_0 can also have nonzero entries. For $\hat{S}$, we used the full covariance matrix of the β's obtained from subsamples, but for simplicity and interpretability, one can also use the diagonal entries (variances) of that covariance matrix. This way, one can easily interpret which components of β are shrunken towards β_0 by which amount.

Support Vector Machines. The second model that we used is support vector machines (SVM) for classification. The primal optimization model of the soft-margin SVM is given as:

$$\begin{array}{lll} \underset{\beta,b,\xi}{\text{minimize}} & \frac{1}{2}||\beta||_2^2 + C\sum_{i=1}^N \xi_i & \\ \text{subject to} & y_i(\beta^T x_i + b) \geq 1 - \xi_i, & i = 1,\ldots,N, \\ & \xi_i \geq 0, & i = 1,\ldots,N. \end{array}$$

Instead, we solve this:

$$\begin{array}{lll} \underset{\beta,b,\xi}{\text{minimize}} & \frac{1}{2}||L^T(\beta - \beta_0)||_2^2 + \sum_{i=1}^N \xi_i & \\ \text{subject to} & y_i(\beta \cdot x_i + b) \geq 1 - \xi_i, & i = 1,\ldots,N, \\ & \xi_i \geq 0, & i = 1,\ldots,N. \end{array}$$

Let $\beta' = L^T(\beta - \beta_0)$. Hence, $\beta = L^{-T}\beta' + \beta_0$

$$\begin{array}{lll} \underset{\beta,b,\xi}{\text{minimize}} & \frac{1}{2}||\beta'||_2^2 + \sum_{i=1}^N \xi_i & \\ \text{subject to} & y_i((L^{-T}\beta' + \beta_0) \cdot x_i + b) \geq 1 - \xi_i, & i = 1,\ldots,N, \\ & \xi_i \geq 0, & i = 1,\ldots,N. \end{array}$$

$$\begin{array}{lll} \underset{\beta,b,\xi}{\text{minimize}} & \frac{1}{2}||\beta'||_2^2 + \sum_{i=1}^N \xi_i & \\ \text{subject to} & y_i(\beta' \cdot (L^{-1}x_i) + \beta_0 \cdot x_i + b) \geq 1 - \xi_i, & i = 1,\ldots,N, \\ & \xi_i \geq 0, & i = 1,\ldots,N. \end{array}$$

$$y_i(\begin{bmatrix}\beta' & 1\end{bmatrix} \cdot \begin{bmatrix}L^{-1}x_i & \beta_0 x_i\end{bmatrix} + b) \geq 1 - \xi_i$$

Regularization in SVM problems is a bit different from regularization in other supervised learning methods. In the soft-margin SVM problem, the objective function is composed of both $\frac{1}{2}||\beta||_2^2$ term which is related to the reciprocal of the margin between classes (in binary classification setting) and the penalty term $C\sum_{i=1}^{N}\xi_i$ which penalizes the slack variable ξ_i if a given observation i violates the margin of its class. So, the penalty hyperparameter C can be thought of as $1/\lambda$ with respect to the other regularization problems that we mentioned before. For large datasets, using CV to tune C can be computationally very time-consuming, as we will see in Section 2.4. However, our approach does not explicitly tune this hyperparameter; it just obtains prior estimates of the mean (β_0) and covariance matrix ($\hat{S}$) of β by using subsamples from the training dataset, and incorporates these estimates into the full training dataset indirectly, by applying a simple affine transformation on predictor matrix of the training set X_r (scaling it by a matrix L^{-T} where $LL^T = \hat{S}^{-1}$, and adding a new vector, $X_r\beta_0$). We will provide a detailed explanation of this affine transformation in Section 2.3.1.2, as it is applicable to other linear models. This transformation is not an essential component of our algorithm; we just suggest this transformation to make it easier to implement our algorithm by using the existing ML libraries, instead of directly casting it as a new optimization model. Alternatively, users can also implement our algorithm by explicitly re-writing the optimization model (adding the penalty term to the loss function) and solve it by one of the existing optimization solvers.

Logistic Regression. Logistic regression is one of the members of the GLM's and it is used for classification tasks. In a nutshell, for binary classification, (binomial) logistic regression takes $X\beta$ as input and returns the class membership probabilities ($\hat{Y}$) as output by using *sigmoid* function. For multiclass classification, similarly, (multinomial) logistic regression takes a linear combination of feature matrix and returns the class membership probabilities for each class by using *softmax* function. This linearity between X and β allows us to use the same subsampling-based method for logistic regression as well. In order to apply our subsampling-based regularization method, we use the same *affine transformation* trick that we used for SVM, to incorporate prior mean and prior covariance information into the model indirectly.

Poisson Regression. Poisson regression is often used to predict count data, such as the number of events that occured in a certain time interval. We also applied the subsampling-based regularization approach on Poisson regression, since it is another

member of the GLM's. For the implementation of both subsampling and CV, we used *glmnet* library. This library does not take multivariate λ vector (λ_j per each feature j) as input, but we circumvented that problem via applying the same *affine transformation trick* again, by using the prior mean and covariance estimates for β to scale and shift X matrix to implicitly regularize each of the parameters.

2.3.1.2 Implementation Details

When a user wants to implement our algorithm, we offer two alternative approaches. The first approach is to append the penalty term $(\beta-\beta_0)^T\hat{S}^{-1}(\beta-\beta_0)$ directly into the loss function manually, and solve it as a convex optimization problem. The only time this approach can be easy is when the loss function is in the form of linear (ridge) regression, since linear regression attains an analytical solution when the hyperparameters are fixed. However, the other models that we implemented (SVM, logistic regression, Poisson regression) do not have analytical solution; and in practice, these models are often fitted by using existing libraries which do not take multivariate hyperparameters as input. In that case, the second approach that we offer is to apply an affine transformation on the feature matrix X.

Now we will explain the details of the affine transformation on SVM problem, but this approach can be applied to other linear models by following the same steps. We would like to write $(\beta-\beta_0)^T\hat{S}^{-1}(\beta-\beta_0)$ in the form of $\hat{\beta}^T\hat{\beta}$. Define $\hat{\beta}=L^T(\beta-\beta_0)$, where L is such that $LL^T=\hat{S}^{-1}$, as mentioned earlier. Rearranging the terms, we have $\beta=L^{-T}\hat{\beta}+\beta_0$. Thus, $X\beta=XL^{-T}\hat{\beta}+X\beta_0$. From this, we can obtain a new feature matrix $X'=[XL^{-T},X\beta_0]$, which is the transformed version of X. After this transformation, we can fit the SVM with the new X', y, $C=1$. After solving this problem, there are two alternative ways to use the estimate $\hat{\beta}$ for prediction. One can either use $\hat{\beta}$ to recover β by reversing the above transformation, or one can apply the same affine transformation on the test feature data. The second approach is easier to implement with the R packages that do not take a user-defined β as input at the prediction stage, so we used that one.

2.3.2 Theoretical Arguments

If the m different parameter estimators $\beta^{(1)}, \beta^{(2)}, \ldots, \beta^{(m)}$ obtained from m different subsamples were independent and identically distributed, we could argue that the distribution of their sample mean β_0 is approximately $N(\beta, \frac{1}{m}\Sigma)$ (where Σ is the unknown covariance matrix of the parameter), by the central limit theorem (CLT). But those estimators are not independent from each other, because they are computed by sampling observations from the same training set. Still, we can use the CLT for weakly dependent random variables, by carefully selecting the subsample size n and the number of subsamples m. For each observation, x_i, the total number of times that it is selected in all subsamples is distributed according to a binomial distribution with m trials and the success probability of $\frac{n}{N}$. If we set $m \approx \frac{N}{n}$, then the expected number of times the observation x_i is sampled across all m subsamples will be $m\frac{n}{N} \approx m\frac{1}{m} = 1$. By using this setting, on average, $\beta^{(i)}$'s obtained from different subsamples will be weakly dependent on each other, thus allowing the CLT to hold [4]. In practical applications, we recommend using this particular setting, whenever the dimensions of a given dataset and the specific convergence conditions of a given model allow it.

At this point, the *asymptotic normality* property of maximum likelihood estimator (MLE) is also worth mentioning (Murphy, 2023). According to this property, if $\hat{\theta}_n$ is the ML estimate of the true parameter θ of a probabilistic model that generates a certain sample of size n, then the sampling distribution of $\sqrt{n}\hat{\theta}_n$ converges to $N(\theta, I(\theta)^{-1})$ when n goes to infinity, where $I(\theta)$ is the *Fisher information matrix* for the given model, and for multivariate Gaussian distribution, $I(\theta)$ is also inverse of the unknown true covariance matrix. From this point of view, the prior covariance matrix $\hat{S}$ that we obtain via subsampling can be seen as a computationally cheap approximation of $I(\beta)^{-1}$ for a given linear model whose unknown parameter is β, when the regularity conditions of MLE hold for that model.

Additionally, we also derived a closed-form solution for the ridge regression problem with a single hyperparameter, λ, for comparison with subsampling. Given an estimator, the expectation of the test error (MSE) for this can be written as

$$\sigma^2 + (\hat{\beta}_\lambda - \beta)^T S (\hat{\beta}_\lambda - \beta)$$

where $S = E(xx^T)$, σ^2 is the unknown variance of residuals, and β is the unknown true parameter for regression.

One can show that in ridge regression, the expectation of this error, with respect to the distribution of the dataset (X, y), is minimized when the hyperparameter λ is

[4] See, for example, Zou, Li, Liang & Wang (2021, Theorem 2(i)) for a more precise statement on estimators like β_0.

chosen as follows

$$\hat{\lambda} \approx \frac{\mathrm{tr}S^{-1}}{\frac{1}{n}\mathrm{tr}S^{-2} + \frac{1}{\sigma^2}\beta^T S^{-1}\beta}.$$

The derivation of this result is included in Appendix A.1. This solution is computationally more efficient than using grid search along with CV, but it is applicable to only ridge regression with a single (scalar) hyperparameter. In the asymptotic case, this formula is somewhat similar to the Hoerl-Kannard-Baldwin formula (Hoerl, Kannard & Baldwin, 1975), but in the finite-sample case, it is different. In our experiments, we used this formula by plugging in OLS estimates of $\hat{\beta}$ and $\hat{\sigma}^2$, since their real values are unknown.

2.4 Numerical Experiments

In our experiments, we tested our methods against k-fold cross-validation (CV) (where $k = 10$) in terms of average test set accuracy (for different training set - testing set partitions) and computation time.

Our results are averaged over 50 training-testing partitions of the data set, where the training set is 75% of the whole dataset. For comparison, we also implemented the diagonal version of the subsampling algorithm (denoted as **subs.diag** in our tables) which uses only the diagonal elements of the prior covariance matrix; and also implemented the subsample-size-adjusted version of the algorithm (denoted as **subs.adj** in our tables) where we scaled the prior covariance matrix by n, the number of observations in a single subsample. We added these two variants of our subsampling-based method to check whether the standard version of our algorithm yields significantly different results from these two apparently refined versions.

We defer some of the additional experimental results, such as the ones that are related to closed-form approximate solution of ridge regression from Section 2.3.2, to Appendix A.2.

2.4.1 Experiments with Linear Regression

In Table 2.1, we compare the test set performance of OLS, CV-based grid-search tuning, closed-form approximate solution and our subsampling-based method for ridge regression, on 6 real datasets.

Dataset		MSE						Time (s)		
		OLS	**CV**	**Closed**	**subs**	**subs.diag**	**subs.adj**	**CV**	**Closed**	**subs**
housing	Avg.	0.2647	0.2647	0.2643	0.273	0.2692	0.2679	0.2066	0.0034	0.0266
(506 × 13)	Std.	0.0538	0.0538	0.0546	0.0622	0.0613	0.0581	0.0608	0.02	0.0177
ccpp	Avg.	0.2647	0.2647	0.2643	0.273	0.2692	0.2679	0.2066	0.0034	0.0266
(9568 × 4)	Std.	0.0538	0.0538	0.0546	0.0622	0.0613	0.0581	0.0608	0.02	0.0177
cadata	Avg.	0.3633	0.3633	0.3633	0.3633	0.3633	0.3633	0.4488	0.0076	0.0514
(20640 × 8)	Std.	0.0114	0.0114	0.0114	0.0114	0.0114	0.0114	0.1072	0.0104	0.0199
superconduct	Avg.	0.2659	0.2659	0.2659	0.266	0.2659	0.2659	1.2174	0.3388	1.3152
(21263 × 81)	Std.	0.0051	0.0051	0.0051	0.0051	0.0051	0.0051	0.291	0.1032	0.2736
abalone	Avg.	0.4771	0.4771	0.4771	0.4771	0.4772	0.4771	0.159	0.0018	0.0214
(4177 × 7)	Std.	0.0341	0.0341	0.0341	0.0341	0.0341	0.0341	0.0498	0.0063	0.012
airfoil	Avg.	0.4879	0.4879	0.488	0.4879	0.4879	0.4879	0.0894	0.0018	0.0144
(1503 × 5)	Std.	0.0363	0.0363	0.0362	0.0363	0.0363	0.0363	0.0339	0.006	0.0095

Table 2.1 Linear Regression Results

2.4.2 Experiments with SVM

In this subsection, we compare our approach with unregularized SVM, subagging (subsample aggregation), and CV-based grid-search tuning, in terms of accuracy and computation time, on 6 real datasets. For convenience, the CV-based method was parallelized by using seven cores on a laptop with with 16 GB RAM and Intel Core i7-10510U CPU with clock rate 1.80 GHz [5]. Comparisons in terms of accuracy (percentage misclassification error, i.e. the percentage of the observations that are classified incorrectly) and timing (in seconds) are given in Table 2.2.

2.4.3 Experiments with Logistic Regression

We tested our method on six real datasets, and compared it with both CV-based grid-search tuning and the version without tuning. These datasets are the same ones that we used for SVM, since their y variables are binary class labels. For convenience, again, CV-based method was parallelized by using seven cores. The results are summarized in Table 2.3.

[5] Without this parallelization, running the CV on these datasets with 50 replications on the same laptop took more than 1 day.

Dataset		Misclassification Error (%)					Time (s)		
		Not Tuned	**CV**	**subs**	**subagg**	**subs.adj**	**CV**	**subs**	**subagg**
spambase	Avg.	12.3113	7.1583	7.3617	9.8504	7.2557	225.955	0.4702	0.2314
(4601 × 57)	Std.	2.4037	0.6377	0.7018	0.9792	0.6999	25.8729	0.1358	0.0464
abalone	Avg.	27.682	17.431	17.41	17.6916	17.8889	94.3186	2.187	1.8304
(4177 × 8)	Std.	9.7732	1.2351	1.2506	1.0229	1.1902	10.5735	0.6558	0.6084
banknote	Avg.	1.7726	0.8863	1.0146	4.8455	0.9796	11.102	0.1786	0.156
(1372 × 4)	Std.	1.4522	0.4997	0.5595	3.0583	0.4776	0.9162	0.0387	0.0346
liver	Avg.	40.6047	30.7209	32.2791	40.5581	31.1628	9.3368	0.1096	0.1038
(345 × 6)	Std.	6.4157	4.2161	4.8348	6.3673	4.0074	1.0348	0.0237	0.0237
wdbc	Avg.	4.8028	2.3099	2.3662	14.2394	2.3099	7.137	0.105	0.0994
(569 × 31)	Std.	1.7114	1.3723	1.3021	4.0699	1.3274	0.9705	0.0152	0.0181
fico	Avg.	40.4613	27.2023	27.2088	27.6895	30.6137	683.1602	6.6982	4.8478
(9781 × 23)	Std.	6.2338	0.8118	0.8117	0.8559	3.591	38.7412	0.7629	0.7349

Table 2.2 SVM Results

Dataset		Misclassification Error (%)					Time (s)		
		Not Tuned	**CV**	**subs**	**subagg**	**subs.adj**	**CV**	**subs**	**subagg**
spambase	Avg.	7.5322	7.553	7.9391	10.0696	7.9722	14.6958	0.2414	0.1492
(4601 × 57)	Std.	0.7047	0.7101	0.7102	1.0094	0.7508	1.3492	0.0482	0.0274
abalone	Avg.	17.6743	17.7146	17.5977	17.3659	17.636	1.2878	0.1106	0.101
(4177 × 8)	Std.	0.867	0.8728	0.8417	0.8917	0.839	0.2109	0.024	0.0192
banknote	Avg.	1.0496	1.4519	1.8717	6.3848	2.7172	0.3912	0.08	0.0806
(1372 × 4)	Std.	0.4963	0.6981	0.6922	3.6851	0.7912	0.1055	0.027	0.0222
liver	Avg.	31.186	31.5814	31.3721	42.186	31.3023	0.1526	0.0628	0.0632
(345 × 6)	Std.	3.9984	4.1118	4.1461	6.0945	3.9856	0.0336	0.0235	0.0185
wdbc	Avg.	3.1268	2.0423	2.2676	9.4085	2.2676	0.3754	0.089	0.0928
(569 × 31)	Std.	1.166	1.0085	1.0189	3.0705	1.0579	0.0733	0.0211	0.0269
fico	Avg.	27.0312	26.9704	27.0361	27.2906	27.0361	12.261	0.1796	0.1526
(9781 × 23)	Std.	0.7608	0.8056	0.7716	0.8892	0.7819	0.8207	0.0478	0.0323

Table 2.3 Logistic Regression Results

2.4.4 Experiments with Poisson Regression

This time we used 20 subsamples for each dataset and used 5% of the training data as subsample size (instead of 1%) because, if we had used less number of data points, it would have severely affected the convergence of the optimization algorithm that *glmnet* uses to fit Poisson regression. We used *deviance* as the error measure to fit and test the algorithm, rather than MSE, because it is much less sensitive to extremely large values in y for the Poisson regression setting than MSE. The results are summarized in Table 2.4.

2.4.5 Overall Comparison via Ranking

Dataset		Deviance					Time (s)	
		Unregularized	**CV**	**subs**	**subs.diag**	**subs.adj**	**CV**	**subs**
phdpublications	Avg.	1.8233	1.8236	1.822	1.822	1.822	0.1506	0.0609
(915 × 7)	Std.	0.1423	0.1443	0.1461	0.1472	0.1461	0.0136	0.0109
NMES1988	Avg	5.2242	5.2135	5.2394	5.2076	5.2409	1.6186	0.3166
(4406 × 30)	Std.	0.3476	0.3261	0.3897	0.3435	0.3919	0.4689	0.2221
Medicaid1986	Avg.	3.1233	3.1132	3.105	3.1061	3.1047	0.215	0.083
(996 × 18)	Std.	0.4104	0.4256	0.4193	0.4339	0.419	0.0441	0.0296
bikeshare	Avg.	8.5747	10.0315	8.637	8.5918	8.6432	5.467	1.0084
(8645 × 51)	Std.	0.2664	0.2356	0.2664	0.2597	0.267	1.52	0.3081
recreationdemand	Avg.	6.6554	3.897	4.1585	3.8284	4.1585	0.2666	0.3462
(659 × 9)	Std.	5.3766	1.136	1.5761	1.0999	1.576	0.0588	0.1198

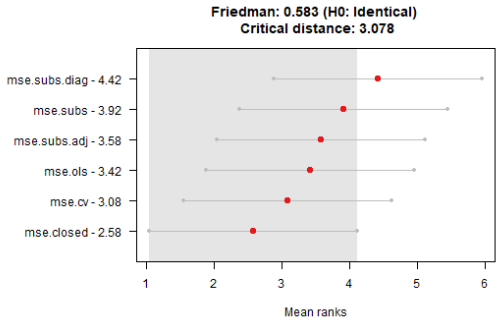
Table 2.4 Poisson Regression Results

In order to make sure about the performance of our algorithm, we also applied Friedman and Nemenyi tests on the rankings of the methods (in terms of mean test-set accuracy results in the aforementioned tables) that were obtained from our experiments. The results of these tests are summarized in the Figure 2.1. These test results further reinforce our conclusion that the accuracy of the subsampling-based method is comparable to that of CV-based grid-search tuning. At first look, the difference between rankings in SVM and logistic regression results may seem a little bit odd, but this difference is caused by the role of regularization term in these methods. More specifically, regularization term in SVM is a crucial component of the loss function (this loss function can not be written without that term in the primal SVM), whereas the classical loss function of logistic regression does not need this term. For this reason, the effect of hyperparameter tuning turned out to be more visible in the ranking of SVM results than that of logistic regression.

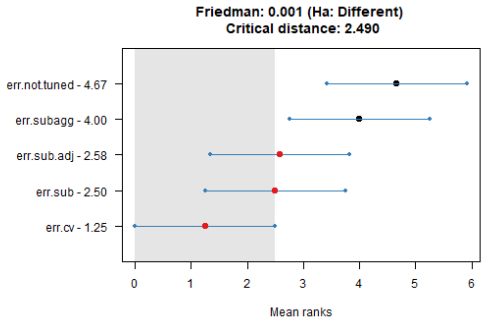
2.5 Conclusion and Discussion

In this chapter of the dissertation, we proposed a computationally efficient algorithm to tune regularization hyperparameters for linear models. We implemented our algorithm on various types of linear models that are used in supervised learning, and tested each of them on real datasets. Our experimental results show that our algorithm yields accurate results that are comparable to CV-based grid-search tuning, but in a much shorter time.

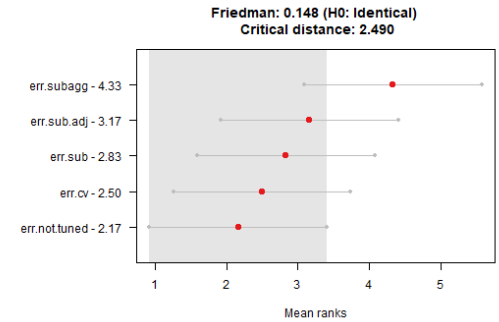
One possible future direction of research on our approach can be the “optimal" selection of subsample size and number of subsamples that guarantee better performance



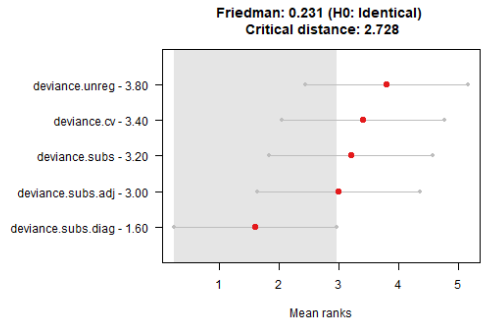
(a) Linear regression



(b) SVM



(c) Logistic regression



(d) Poisson regression

Figure 2.1 Friedman and Nemenyi rank test results for the test set accuracies of the models

than the one that we have suggested as a rule-of-thumb.

Another possible direction of future research would be to facilitate the application of subsampling-based algorithm to nonlinear models (such as neural networks). In that case, the affine transformation trick that we had suggested would not be applicable, but one can still apply our method to a given nonlinear model by directly adding the multivariate penalty term (obtained from subsampling) to the objective function. This would require writing the optimization model manually (instead of using an existing ML library), and optimizing it by using a solver. However, a potential research along these lines might be able to find a more user-friendly way of incorporating this multivariate penalty term to a given nonlinear model.

Using a subsample of the dataset inherently introduces a noise in the estimation of a given parameter β. One can take advantage of this inherent noise in a differentially private (DP) parameter estimation setting for linear models, *i.e.*, by using the relation between the subsample size and the variance of the parameter estimate, one can adjust the subsample size to make the estimation of β satisfy DP constraints.

Discussion: Connections Between Subsampling and Delete-d Jackknife Estimator

Jackknife estimator is one of the classical and popular methods for computing the variance of a sample statistic $\hat{\theta} = s(X)$ obtained from a given statistical procedure $s(\cdot)$. For a given sample $X_1, \ldots, X_N$, it is essentially based on computing the given statistical procedure $s(\cdot)$ N times while leaving out one data point at each computation and obtaining the estimates $\hat{\theta}_1, \ldots, \hat{\theta}_N$. Then, one can compute the variance of the $\hat{\theta}$ by using the standard (leave-one-out) jackknife formula

$$Var(\hat{\theta})_{jack} = \frac{N-1}{N} \sum_{i=1}^{N} (\hat{\theta}_i - \hat{\theta}_{(\cdot)})^2$$

where

$$\hat{\theta}_{(\cdot)} = \frac{1}{N} \sum_{i=1}^{N} \hat{\theta}_i.$$

For nonsmooth statistics (like median), standard jackknife method can yield highly biased estimates. To overcome this limitation, a more general version of jackknife (namely, *delete-d jackknife*) was proposed (Shao & Wu, 1989), which is based on leaving out d data points ($2 < d < N$) at each iteration, instead of 1. Its formula is

$$Var(\hat{\theta})_{dd-jack} = \frac{N-d}{d\binom{N}{d}} \sum_{i=1}^{\binom{N}{d}} (\hat{\theta}_i - \hat{\theta}_{(\cdot)})^2$$

where

$$\hat{\theta}_{(\cdot)} = \frac{1}{\binom{N}{d}} \sum_{i=1}^{\binom{N}{d}} \hat{\theta}_i.$$

When $d = 1$, this formula boils down to the standard jackknife formula. In practice, d is chosen as $\sqrt{N} < d < N$. For a large sample size N and $d \geq 2$, computation of the exact delete-d jackknife estimator is often intractable, due to the requirement of using $\binom{N}{d}$ different combinations of subsets from the given data. For a large dataset and a complex learning algorithm, one can avoid this computational challenge by using a much smaller number of subsamples $t \ll \binom{N}{d}$, each having the sample size $m = N - d$. In that case, the delete-d jackknife formula would be simply modified as

$$Var(\hat{\theta})_{sub-jack} = \frac{m}{(N-m)t} \sum_{i=1}^{t} (\hat{\theta}_i - \hat{\theta}_{(\cdot)})^2$$

where

$$\hat{\theta}_{(\cdot)} = \frac{1}{t} \sum_{i=1}^{t} \hat{\theta}_i.$$

From this perspective, our subsampling-based algorithm can be viewed as a Monte Carlo approximation of the exact delete-d jackknife estimator of variance, but it requires to be multiplied by the correction term $\frac{m(t-1)}{(N-m)t}$ for a better approximation of the exact delete-d jackknife.

For another potential direction of future work, this relation can be further investigated and implemented as a modified version of our algorithm.

New Experiments

As an extension of our work, we also compared our subsampling-based approach with two other Bayesian hyperparameter tuning methods, namely (i) Bayesian Optimization, and (ii) Automatic Relevance Determination.

Bayesian Optimization, as we mentioned in 5.2, is based on approximating the surface of the unknown objective function (generalization error function) by a surrogate function obtained from a Gaussian process and updating this surrogate function at each iteration by taking new samples from the search space. In our experiments, we used *rBayesianOptimization* library in R.

Automatic Relevance Determination is another Bayesian method for finding optimal hyperparameters for regularization. It was originally developed by (MacKay, 1995) and (Neal, 1995) for regularized fitting of neural networks. Later on, modified versions of this method were also applied to regularize linear regression and SVM's under different names, such as *Sparse Bayesian Learning* and *Relevance Vector Ma-*

chines, respectively (Tipping, 2001). This method is based on using a Gaussian prior for β, as in $p(\beta) = N(\beta|0, \Lambda^{-1})$, where $\Lambda = diag([\lambda_1, \ldots, \lambda_m])$, and directly maximizing the marginal likelihood function. In this approach, since each feature is associated with a different precision hyperparameter (λ_j), redundant features are sparsified by having larger λ_j values. So, it yields similar solutions to LASSO regularization method. To our knowledge, R programming language does not have a dedicated library that implements ARD for GLMs, so we manually implemented an iterative algorithm that updates β and Λ in a cyclic fashion, which was proposed by (Wipf & Nagarajan, 2010).

Each of these new methods were compared against unregularized models, CV-based regularization, and our subsampling-based method. Each of these experiments were run 50 times for each dataset. The averages and standard deviations of the test-set accuracy results are given in Tables 2.5-2.7. In these new experiments, we used the correction term that relates our method to delete-d jackknife estimator. It seems that this correction term has provided some improvements in the performance of our method.

Dataset		**Accuracy (MSE)**					**Time (s)**			
		OLS	**CV**	**Subs**	**ARD**	**BayesOpt**	**CV**	**Subs**	**ARD**	**BayesOpt**
housing	Avg.	23.5195	23.4643	23.5132	25.9136	23.5539	0.0606	0.0058	11.4430	17.0842
(506×13)	Std.	4.4905	4.8876	4.5086	6.1010	4.6254	0.0299	0.0085	3.0044	1.2934
airfoil	Avg.	23.2368	23.3602	23.2381	23.9707	23.2481	0.0694	0.0038	57.1512	21.2120
(1503×5)	Std.	1.9117	1.8666	1.9122	1.8407	1.9190	0.0222	0.0069	8.7735	4.2071
generated	Avg.	4.3028	12.0065	4.3297	4.3418	4.2304	0.0888	0.0074	42.5000	25.0592
(1000×10)	Std.	0.4102	1.0950	0.4770	0.6044	0.3724	0.0321	0.0087	8.9193	6.0379

Table 2.5 Linear Regression Results

Dataset		**Misclassification (%)**					**Time (s)**			
		Unregularized	**CV**	**Subs**	**ARD**	**BayesOpt**	**CV**	**Subs**	**ARD**	**BayesOpt**
banknote	Avg.	0.9854	0.9795	1.0903	0.9795	27.2186	0.3982	0.0172	108.7200	19.7154
(1372×4)	Std.	0.4323	0.4628	0.4705	0.4628	12.9516	0.0792	0.0121	13.0367	2.7522
liver	Avg.	31.7674	31.7906	31.7906	31.7906	40.3488	0.1588	0.0158	23.4136	17.0388
(345×6)	Std.	4.3281	4.2843	4.6784	4.2843	4.8841	0.0279	0.0097	3.1638	1.9817
wdbc	Avg.	7.8732	7.9014	7.1408	7.9014	9.7042	0.5398	0.0272	44.6262	19.2664
(569×31)	Std.	2.0255	2.0827	2.4312	2.0827	5.9607	0.1733	0.0092	7.6393	4.7868

Table 2.6 Logistic Regression Results

Dataset		**Deviance**					**Time (s)**			
		Unregularized	**CV**	**Subs**	**ARD**	**BayesOpt**	**CV**	**Subs**	**ARD**	**BayesOpt**
Medicaid1986	Avg.	3.0292	3.0245	3.0149	3.0292	3.0853	0.2632	0.1106	29.1596	20.5720
(996×18)	Std.	0.4690	0.4944	0.5004	0.4690	0.5526	0.0658	0.0329	4.5420	2.6470
phdpublications	Avg.	1.7998	1.8027	1.8018	1.7998	1.8411	0.1884	0.0932	12.4072	19.1148
(915×7)	Std.	0.1492	0.1531	0.1550	0.1492	0.1827	0.0651	0.0411	2.9877	3.5762
recreationdemand	Avg.	6.6141	4.0019	3.9055	6.6258	4.0723	0.1464	0.2782	10.6616	23.2950
(659×9)	Std.	6.1798	1.5058	1.4482	6.1859	1.5019	0.0245	0.1159	2.6995	2.4716

Table 2.7 Poisson Regression Results

3. A BAYESIAN APPROACH FOR SOLVING ROBUST REGRESSION PROBLEMS

In this chapter, we propose an Expectation-Maximization (EM) algorithm for fitting a mixture of m-estimators, specifically crafted for robust regression problems. In contrast to standard linear regression, robust regression is often utilized when the residuals in a regression problem have non-normal distribution; for example, it may have heavy tails, outliers, non-constant variance and high skewness. In these cases, standard linear regression becomes too sensitive to deviations from the normality assumptions of the residuals. For example, some outliers may drastically change the slope of the hyperplane that is fitted by ordinary least-square (OLS); also some data points with a high *leverage score* can cause a significant shift in the fitted hyperplane (James, 2013). We address these issues by using a mixture of m-estimators whose loss functions are associated with heavy-tailed distributions (such as Laplace and Cauchy distributions), thus they are less sensitive to peculiarities in residuals. Our EM-based algorithm handles this fitting procedure by learning the location and scale parameters of these distributions, and mixture weights that associate each data point with each m-estimator. In our experiments, we demonstrate the estimation of this mixture model over each m-estimator and standard OLS.

The work in this chapter was conducted mostly by the author of this dissertation. Sinan Yıldırım and İlker Birbil also guided the author by their useful suggestions throughout our meetings on this work.

3.1 Introduction

Robust regression refers to a variety of regression methods that are not too sensitive to outliers and skewness in residuals. It is also implicitly connected to the regularized regression problem that we tackled in the previous chapter of this dissertation. When

we apply regularization on regression parameters, we implicitly make them more robust against outliers, because regularization causes shrinkage in the variance of these parameters. Regularization is achieved by penalizing the irrelevant features (columns) of the predictor matrix, whereas robustness is achieved by penalizing the observations (rows) which have less predictive power. Therefore, regularization and robustness problems are in some sense "dual" to each other; one can formulate these two problems together as a multi-objective optimization problem, in which the goal would be to find the optimal weights for penalizing observations and features, simultaneously.

3.2 Related Literature

In real-life applications of regression, one often comes across datasets that have peculiar noise patterns that have outliers and high skewness. Standard linear regression assumes that the residuals have normal distribution, but when we fit ordinary least-squares (OLS) to a dataset which violates this assumption, it reduces the predictive performance of linear regression. One way of remedying this problem is to replace the quadratic loss function (MSE) of OLS with another one which is less sensitive to outliers and skewness. In the literature, there is a large family of estimators, called *m-estimators*, which contain mean-squared error, least absolute deviation, Huber loss and many other symmetric, positive and continuous loss functions as its special cases (De Menezes et al., 2021). Even though MSE is also an m-estimator, we will focus on its robust members.

Most of the methods that we have seen in the literature focus on fitting only a single m-estimator to data [Arslan & Billor (2000), Noor-Ul-Amin, Asghar, Sanaullah & Shehzad (2018), Huang, Wang & Zheng (2014), De Menezes et al. (2021)]. For a single m-estimator, finding the optimal weights for observations is often handled by using *iteratively reweighted least squares* (IRLS) procedure (Susanti et al., 2014), but in the case of a mixture of m-estimators it requires more complicated methods. Additionally, using a single m-estimator may not always be sufficient to capture the characteristics of the stochastic process that generates a given dataset. In the literature, there are some other uses of mixture models (mixture of m-estimators) to fit robust regression. For example, (Tak et al., 2019) proposes a Gibbs sampler to fit a mixture of Gaussian and Student's t-distribution to distinguish outliers from "normal" observations. (Bai et al., 2012) and (Doğru & Arslan, 2021) use an EM

algorithm to fit a mixture of linear models, but for each linear model they use the same m-estimator. These methods are either not flexible enough or not efficient enough to incorporate more than two m-estimators to the mixture model. For these reasons, we propose an efficient Expectation-Maximization algorithm to fit a mixture of m-estimators for a given dataset. In our experiments, we use a mixture of three m-estimators, but our method is flexible enough to incorporate as many of these estimators as one may want.

3.3 Methodology

In regression setting, for a given loss function $\rho(y_i, \hat{y_i})$ that measures the discrepancy between a continuous observation y_i and its estimate $\hat{y_i}$, let $r_i = y_i - f(\theta; x_i)$, where r_i's be the residuals for each observation. The objective function can be parametrized in terms of residuals as follows:

$$\min_{\theta} \sum_{i=1}^{n} \rho(r_i(\theta)) = \min_{\theta} -\log \prod_{i=1}^{n} p(r_i(\theta)) = \max_{\theta} \prod_{i=1}^{n} p(r_i(\theta))$$

where $p(r_i) = \exp(-\rho(r_i))$.

In the presence of different ρ_j's (loss functions), consider the mixture of error distributions

$$p_m(r_i) = \sum_{j=1}^{J} a_j . \underbrace{\exp(-\rho_j(r_i))}_{p_j(r_i)},$$

where $\sum_{j=1}^{J} a_j = 1$. Let $\rho_m(r_i) = -\log p_m(r_i)$. Then the M-estimator for the mixture is formulated as

$$\min_{\theta} \sum_{i=1}^{n} \rho_m(r_i) = \min_{\theta} \sum_{i=1}^{n} -\log \left[\sum_{j=1}^{J} a_j . \exp(-\rho_j(r_i(\theta))) \right].$$

This objective function does not attain an analytical solution. In order to optimize it, one needs to use iterative algorithms to update the parameters of regression (θ), and the mixture weights (a_j) separately. For each iteration, when the mixture weights are fixed, regression parameters can be computed via *iteratively reweighted least squares (IRLS)* method. Here, the actual complicated part is how to determine the mixture weights.

In order to derive updating rules for these parameters, let us first work on the partial derivatives of the objective function. During this derivation, we will also use the notions of *influence function* and *weight function* that are used for measuring the sensitivity of regression parameters to each observation (Zhang, 1997).

First, we can write the gradient with respect to θ (while keeping a_j's constant) and set it equal to zero, as follows:

$$\nabla_\theta NLL = -\sum_{i=1}^{n} \left[\frac{-\sum_{j=1}^{J} a_j . \frac{\partial \rho_j(r_i(\theta))}{\partial r_i(\theta)} . \frac{\partial r_i(\theta)}{\partial \theta} . \exp(-\rho_j(r_i(\theta)))}{\sum_{j=1}^{J} a_j . \exp(-\rho_j(r_i(\theta)))} \right] = 0.$$

The influence function $\psi_j(r_i(\theta))$ of an m-estimator j is defined as

$$\psi_j(r_i(\theta)) = \frac{\partial \rho_j(r_i(\theta))}{\partial r_i(\theta)}.$$

Using this definition, we can rewrite the gradient of NLL as follows:

$$\nabla_\theta NLL = \sum_{i=1}^{n} \frac{\sum_{j=1}^{J} a_j . \psi_j(r_i(\theta)) . \frac{\partial r_i(\theta)}{\partial \theta} . \exp(-\rho_j(r_i(\theta)))}{\sum_{j=1}^{J} a_j . \exp(-\rho_j(r_i(\theta)))}$$

Similarly, the weight function $w_j(r_i(\theta))$ of an m-estimator j is defined as

$$w_j(r_i(\theta)) = \frac{\psi_j(r_i(\theta))}{r_i(\theta)} \rightarrow \psi_j(r_i(\theta)) = w_j(r_i(\theta)) . r_i(\theta).$$

Using this relation, we can again rewrite the gradient of NLL as follows:

$$\nabla_\theta NLL = \sum_{i=1}^{n} \frac{\sum_{j=1}^{J} a_j . w_j(r_i(\theta)) . r_i(\theta) . \frac{\partial r_i(\theta)}{\partial \theta} . \exp(-\rho_j(r_i(\theta)))}{\sum_{j=1}^{J} a_j . \exp(-\rho_j(r_i(\theta)))}.$$

At this point, we can rewrite some of the terms above as an auxiliary variable z_{ij}, which denotes the posterior probability that observation i is related to jth m-estimator, via the following relation:

$$z_{ij} = \frac{a_j . \exp(-\rho_j(r_i(\theta)))}{\sum_{j=1}^{J} a_j . \exp(-\rho_j(r_i(\theta)))}.$$

Here, z_{ij}'s are analogous to "cluster membership" posterior probabilities (for each observation i and cluster j) that are used in some soft-clustering algorithms and some Bayesian methods for mixture density estimation (Deisenroth, Faisal & Ong,

2020).

If we substitute z_{ij} instead of the corresponding term inside the gradient of NLL function, we have the following succinct form:

$$\nabla_\theta NLL = \sum_{i=1}^{n}\sum_{j=1}^{J} z_{ij}.w_j(r_i(\theta)).r_i(\theta).\frac{\partial r_i(\theta)}{\partial \theta} = 0.$$

This last form of the gradient is equivalent to first order optimality condition of the following weighted least-squares problem:

$$\min_\theta \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{J} z_{ij}^{(t-1)}.w_j(r_i(\theta^{(t-1)})).r_i(\theta^{(t)})^2.$$

Here, for a particular iteration t, the terms $z_{ij}^{(t-1)}.w_j(r_i(\theta^{(t-1)}))$ are fixed weights that were obtained by using $r_i(\theta^{(t-1)})$'s from the previous iteration $(t-1)$. For these fixed terms, the overall problem can be solved as a weighted least squares problem, as we intuited before. After solving it for θ, we again update these weights by using new $r_i(\theta)$'s and continue in cyclic fashion between weights and θ until their values do not change significantly in consecutive iterations (in other words, until they converge to a local minimum).

Now we can take the derivative of the NLL with respect to a_j to see how we can update these mixture weights while keeping θ constant. Before that, we need to augment the constraint $\sum_{j=1}^{J} a_j = 1$ into the NLL objective function, by using a Lagrange multiplier (λ) as follows:

$$\min_a \sum_{i=1}^{n} -\log\left[\sum_{j=1}^{J} a_j.\exp(-\rho_j(r_i(\theta)))\right] + \lambda(\sum_{j=1}^{J} a_j - 1).$$

Now we can proceed with the derivative with respect to a_j.

$$\frac{\partial NLL}{\partial a_j} = -\sum_{i=1}^{n} \frac{\exp(-\rho_j(r_i(\theta)))}{\sum_{j=1}^{J} a_j.\exp(-\rho_j(r_i(\theta)))} + \lambda$$

$$= -\sum_{i=1}^{n} \frac{1}{a_j}.\frac{a_j.\exp(-\rho_j(r_i(\theta)))}{\sum_{j=1}^{J} a_j.\exp(-\rho_j(r_i(\theta)))} + \lambda = 0$$

Some terms above seem familiar, from the part that we defined z_{ij}'s. We can now

substitute z_{ij} to simplify the expression above, as follows:

$$\frac{\partial NLL}{\partial a_j} = -\sum_{i=1}^{n} \frac{1}{a_j}.z_{ij} + \lambda = 0 \rightarrow a_j = \frac{\sum_{i=1}^{n} z_{ij}}{\lambda} = \frac{n_j}{\lambda},$$

where n_j is the sum of mixture weights over all n observations (analogous to number of members assigned to cluster j, in clustering methods) for the jth m-estimator.

For further simplification, we also need to determine λ by taking the derivative of augmented NLL with respect to λ:

$$\frac{\partial NLL}{\partial \lambda} = \sum_{j=1}^{J} a_j - 1 = 0 \rightarrow \sum_{j=1}^{J} a_j = 1$$

So,

$$1 = \sum_{j=1}^{J} a_j = \sum_{j=1}^{J} \frac{n_j}{\lambda} = \frac{n}{\lambda} \rightarrow n = \lambda.$$

Finally,

$$a_j = \frac{n_j}{\lambda} = \frac{n_j}{n}.$$

Algorithm 2 EM algorithm for fitting mixture of m-estimators

Input: Dataset: X, Y, initial weight matrices: $W_j = I_{n \times n}$ for each m-estimator j, termination condition number: ϵ, initial prior probability vector: $a = \frac{1}{j} * 1_{1 \times j}$, initial posterior probability matrix: $Z = \frac{1}{j} * 1_{N \times j}$

Output: Regression parameter: β, final weight matrices: W_j, final posterior probabilities: Z, final prior probabilities: a

while $|\beta^t - \beta^{t-1}| > \epsilon$ **do**

- `Compute` β^t `using` X, Y, W^{t-1}
- `Compute residuals` R^t `using` X, Y, β^t
- `Update posterior probability matrix` Z^t `using` a^{t-1}, R^t
- `Update prior probability vector` a^t `using` Z^t
- `Compute` W_j^t `using` Z^t, W_j^{t-1} `for each` j
- `Set` $W^t = \sum_{j=1}^{J} W_j^{t-1}$

return β, W_j's, Z, and a

3.4 Experimental Results

The derivatives that we obtained in previous part directly give us updating rules that can be used in *Expectation-Maximization* (EM) algorithm (or other coordinate-based algorithms). We implemented an EM algorithm by using these update rules, and its steps are summarized as pseudocode in Algorithm 2. In our implementation, we used a mixture of Laplace m-estimator (least absolute deviation), Gaussian m-estimator (which is not robust on its own) and Cauchy m-estimator (whose distribution has heavy tails).

The weight function for Laplace estimator is in the form of $\frac{1}{|r_i|}$. For small residual (r_i) values, this form can cause potential problems, like over-inflated weights, and division by zero. In order to tame the over-inflated values in the weight matrices, we used normalization (dividing the entries of each weight matrix by the sum of all weights in that matrix). Also, in order to avoid division by zero in the weight function of Laplace m-estimator, we used $\frac{1}{\max\{\delta,|r|)\}}$ where δ is a small constant like 0.0001. Here are the scenarios for our experiments: (i) No outlier, (ii) Randomly selected 10 percent of the Y_{tr} were added noise from intervals [-4sd, -2sd] and [2sd, 4sd], (iii) Randomly selected 10 percent of the Y_{tr} were added noise from intervals [-6sd, -3sd] and [3sd, 6sd], (iv) Randomly selected 10 percent of the Y_{tr} were added noise from intervals [-10sd, -5sd] and [5sd, 10sd], where sd denotes the standard deviation of Y_{tr}. We ran the algorithm and its competitors (Gaussian, Laplace, Cauchy m-estimators) for 100 times for different training set - testing set partitions (where the training set is the 75% of the whole dataset) to compare its test MSE with that of OLS. For these experiments, we used three real datasets, namely *boston (housing)*, *airfoil self-noise* and *abalone* datasets. The first one has 506 observations and 13 predictors, the second one has 1503 observations and 5 predictors, and the third one has 4177 observations and 7 predictors. Each of them were standardized before running the algorithm. Our results (averages and standard deviations of MSE values) are summarized in tables 3.1, 3.2 and 3.3.

Setting	MSE_{ols}	$MSE_{mixture}$	$MSE_{laplace}$	MSE_{cauchy}
No outliers	0.2759	0.2787	0.2927	0.2860
	0.0571	0.0673	0.0781	0.0768
Outliers from [-4sd, -2sd] and [2sd, 4sd]	0.4268	0.2963	0.3030	0.2966
	0.1026	0.0789	0.0823	0.0804
Outliers from [-6sd, -3sd] and [3sd, 6sd]	0.6029	0.2963	0.3048	0.2978
	0.1959	0.0683	0.0732	0.0693
Outliers from [-10sd, -5sd] and [5sd, 10sd]	1.1686	0.2975	0.3076	0.3006
	0.4016	0.0855	0.0914	0.0879

Table 3.1 Comparison of mixture model with 3 single m-estimators in *boston (housing)* dataset

Setting	MSE_{ols}	$MSE_{mixture}$	$MSE_{laplace}$	MSE_{cauchy}
No outliers	0.4934	0.4935	0.5100	0.5041
	0.0358	0.0358	0.0415	0.0401
Outliers from [-4sd, -2sd] and [2sd, 4sd]	0.4935	0.4915	0.5066	0.5004
	0.0346	0.0353	0.0413	0.0395
Outliers from [-6sd, -3sd] and [3sd, 6sd]	0.5003	0.4934	0.5075	0.5016
	0.0408	0.0402	0.0425	0.0413
Outliers from [-10sd, -5sd] and [5sd, 10sd]	0.5207	0.4872	0.5009	0.4950
	0.0451	0.0364	0.0400	0.0385

Table 3.2 Comparison of mixture model with 3 single m-estimators in *airfoil* dataset

Setting	MSE_{ols}	$MSE_{mixture}$	$MSE_{laplace}$	MSE_{cauchy}
No outliers	0.4839	0.4811	0.4990	0.4913
	0.0356	0.0356	0.0372	0.0363
Outliers from [-4sd, -2sd] and [2sd, 4sd]	0.4808	0.4834	0.4939	0.4864
	0.0308	0.0336	0.0342	0.0331
Outliers from [-6sd, -3sd] and [3sd, 6sd]	0.4939	0.4955	0.5078	0.4997
	0.0366	0.0366	0.0375	0.0373
Outliers from [-10sd, -5sd] and [5sd, 10sd]	0.5042	0.4901	0.5071	0.4986
	0.0437	0.0347	0.0373	0.0354

Table 3.3 Comparison of mixture model with 3 single m-estimators in *abalone* dataset

According to these results, the mixture model performed better than its competitors most of the time. The differences between the results can be seen more clearly in *boston* dataset which has the lowest ratio of observations-to-predictors ($\frac{n}{m}$) among them. This ratio is higher for *airfoil* dataset, and *abalone* dataset has the highest ratio. When this ratio gets higher, the significance of the difference between robust and non-robust methods diminishes; because, when a dataset has a higher ratio of observations for each predictor, variance of the regression parameters gets smaller, thus the fitted model becomes less sensitive to outliers - even when the fitted model is non-robust on its own.

In order to further refine our model, we also derived an updating rule for the scale parameters of m-estimators (such as the variance parameter of Gaussian m-estimator), by taking the derivative of the NLL with respect to each of the j scale parameters. Let us now re-write our NLL:

$$NLL := \sum_{i=1}^{n} -\log\left[\sum_{j=1}^{J} a_j.\exp(-\rho_j(r_i(\theta, s_j)))\right]$$

If we denote the scale parameter of m-estimator j by s_j, here is the derivative of NLL with respect to s_j:

$$\frac{\partial NLL}{\partial s_j} = -\sum_{i=1}^{n}\left[\frac{-a_j.\frac{\partial \rho_j(r_i(\theta,s_j))}{\partial s_j}.\exp(-\rho_j(r_i(\theta,s_j)))}{\sum_{j=1}^{J} a_j.\exp(-\rho_j(r_i(\theta,s_j)))}\right] = 0$$

Similar to what we did before, we can use the auxiliary variable z_{ij} again, which denotes the posterior probability that observation i is related to jth m-estimator, via the following relation:

$$z_{ij} = \frac{a_j.\exp(-\rho_j(r_i(\theta, s_j)))}{\sum_{j=1}^{J} a_j.\exp(-\rho_j(r_i(\theta, s_j)))}$$

Again, if we substitute z_{ij} instead of the corresponding term inside the gradient of NLL function, we have the following succinct form:

$$\frac{\partial NLL}{\partial s_j} = \sum_{i=1}^{n} z_{ij}.\frac{\partial \rho_j(r_i(\theta, s_j))}{\partial s_j} = 0$$

So, this derivation gives us an updating rule for optimizing the scale parameter of each m-estimator at the M-step of our EM algorithm. We can now provide special cases of the term above, for each of the m-estimators (Gaussian, Laplace, Cauchy) that we use in our mixture model.

For the Gaussian m-estimator, we have the following log-likelihood (as part of the overall mixture model):

$$LL^{Gaussian} = \sum_{i=1}^{n} z_{ij}.\left[-\frac{\log(2\pi)}{2} - \frac{\log(\sigma^2)}{2} - \frac{r_i(\theta)^2}{2\sigma^2}\right]$$

Taking its derivative w.r.t. σ^2, we have:

$$\frac{\partial LL^{Gaussian}}{\partial \sigma^2} = \frac{1}{2}\sum_{i=1}^{n} z_{ij}.\left[-\frac{1}{\sigma^2} + \frac{r_i(\theta)^2}{\sigma^4}\right] = 0$$

After rearranging these terms, we have:

$$\sigma^2 = \frac{\sum_{i=1}^{n} z_{ij}.r_i(\theta)^2}{\sum_{i=1}^{n} z_{ij}}$$

Now, we can do the same operations for Laplace m-estimator:

$$LL^{Laplace} = \sum_{i=1}^{n} z_{ij}.\left[-\log(2b) - \frac{|r_i(\theta)|}{b}\right]$$

Its derivative w.r.t. b is as follows:

$$\frac{\partial LL^{Laplace}}{\partial b} = \sum_{i=1}^{n} z_{ij}.\left[-\frac{1}{b} + \frac{|r_i(\theta)|}{b^2}\right] = 0$$

Rearranging the terms, we have:

$$b = \frac{\sum_{i=1}^{n} z_{ij}.|r_i(\theta)|}{\sum_{i=1}^{n} z_{ij}}$$

We can now try to do the same operations for Cauchy m-estimator.

$$LL^{Cauchy} = \sum_{i=1}^{n} z_{ij}.\left[-\log(\gamma\pi) - \log\left(1 + \left(\frac{r_i(\theta)}{\gamma}\right)^2\right)\right]$$

Taking its derivative w.r.t. γ, we have:

$$\frac{\partial LL^{Cauchy}}{\partial \gamma} = \sum_{i=1}^{n} z_{ij}.\left[-\frac{1}{\gamma} + \frac{2.r_i(\theta)^2}{\gamma(\gamma^2 + r_i(\theta)^2)}\right] = 0$$

Or,

$$\frac{\partial LL^{Cauchy}}{\partial \gamma} = \sum_{i=1}^{n} z_{ij}.\left[\frac{2.r_i(\theta)^2}{\gamma^2 + r_i(\theta)^2} - 1\right] = 0$$

The last expression above does not attain an analytical solution, but one can handle this step with a basic univariate optimization solver. For this purpose, we used R programming language's built-in *optimize()* function which efficiently handles the optimization of γ parameter. After this improvement we ran the algorithm on *boston*, *airfoil* and *abalone* datasets again, with the same experimental settings as before. Additionally, for comparison, we also developed a naive method which clusters the

observations as "outlier" or "non-outlier", and fits OLS by using the observations that were clustered as non-outliers. This method cycles between updating the cluster labels and updating the regression parameters until the mean absolute value of the residuals converge to a constant value. Here, the clustering rule is to label the observations whose absolute residuals are at least 3 times higher than the mean absolute error as "outliers". Our results (averages and standard deviations of MSE values) are summarized in tables 3.4, 3.5 and 3.6.

Setting	MSE_{ols}	$MSE_{mixture}$	$MSE_{laplace}$	MSE_{cauchy}	MSE_{naive}
No outliers	0.2833	0.3228	0.3111	0.3064	0.3302
	0.0658	0.0885	0.0849	0.0830	0.0907
Outliers from [-4sd, -2sd] and [2sd, 4sd]	0.4304	0.2926	0.3022	0.2945	0.2897
	0.0977	0.0682	0.0707	0.0716	0.0581
Outliers from [-6sd, -3sd] and [3sd, 6sd]	0.6381	0.3194	0.3289	0.3215	0.3077
	0.1800	0.0827	0.0904	0.0895	0.0769
Outliers from [-10sd, -5sd] and [5sd, 10sd]	1.1907	0.2992	0.3052	0.2967	0.2824
	0.5181	0.0707	0.0741	0.0731	0.0576

Table 3.4 Comparison of mixture model with its competitors in *boston (housing)* dataset

Setting	MSE_{ols}	$MSE_{mixture}$	$MSE_{laplace}$	MSE_{cauchy}	MSE_{naive}
No outliers	0.4842	0.4854	0.4990	0.4936	0.4929
	0.0332	0.0345	0.0407	0.0390	0.0400
Outliers from [-4sd, -2sd] and [2sd, 4sd]	0.4929	0.5160	0.5058	0.5002	0.4905
	0.0339	0.0426	0.0400	0.0383	0.0338
Outliers from [-6sd, -3sd] and [3sd, 6sd]	0.5054	0.5009	0.5101	0.5046	0.4953
	0.0394	0.0399	0.0436	0.0418	0.0364
Outliers from [-10sd, -5sd] and [5sd, 10sd]	0.5150	0.4882	0.5029	0.4964	0.4867
	0.0342	0.0316	0.0373	0.0357	0.0306

Table 3.5 Comparison of mixture model with its competitors in *airfoil* dataset

Setting	MSE_{ols}	$MSE_{mixture}$	$MSE_{laplace}$	MSE_{cauchy}	MSE_{naive}
No outliers	0.4813	0.5212	0.4990	0.4910	0.5021
	0.0275	0.0323	0.0296	0.0287	0.0305
Outliers from [-4sd, -2sd] and [2sd, 4sd]	0.4804	0.5183	0.4950	0.4872	0.4777
	0.0324	0.0386	0.0365	0.0347	0.0328
Outliers from [-6sd, -3sd] and [3sd, 6sd]	0.4955	0.4956	0.5050	0.4968	0.4884
	0.0410	0.0408	0.0417	0.0410	0.0394
Outliers from [-10sd, -5sd] and [5sd, 10sd]	0.5023	0.4929	0.5050	0.4961	0.4831
	0.0392	0.0318	0.0326	0.0320	0.0312

Table 3.6 Comparison of mixture model with its competitors in *abalone* dataset

Surprisingly, the naive method yielded the best results in the settings which contain artificial outliers. This may imply that the estimation of scale parameters in the modified mixture model might be unnecessary and it may lead to over-parametrization. However, our modified approach still performs well in terms of accuracy and it is more useful in terms of making inference about the distribution of outliers.

For completeness, we also include the mixture weights of the m-estimators (found by our method) in Tables 3.7, 3.8, 3.9 for our latest experiments. One can check these weights to make inference about the underlying stochastic process that produced the real data, and about the effect of the added artificial noise on top of that.

Setting	$a_{Laplace}$	$a_{Gaussian}$	a_{Cauchy}
No outliers	0.4197	0.5466	0.0336
Outliers from [-4sd, -2sd] and [2sd, 4sd]	0.2790	0.5511	0.1697
Outliers from [-6sd, -3sd] and [3sd, 6sd]	0.4444	0.4207	0.1348
Outliers from [-10sd, -5sd] and [5sd, 10sd]	0.4447	0.4345	0.1207

Table 3.7 Mixture weights of the mixture model in fitting *boston (housing)* dataset under different settings

Setting	$a_{Laplace}$	$a_{Gaussian}$	a_{Cauchy}
No outliers	0.2958	0.6926	0.0115
Outliers from [-4sd, -2sd] and [2sd, 4sd]	0.4349	0.5092	0.0558
Outliers from [-6sd, -3sd] and [3sd, 6sd]	0.1507	0.5299	0.3193
Outliers from [-10sd, -5sd] and [5sd, 10sd]	0.1845	0.6515	0.1638

Table 3.8 Mixture weights of the mixture model in fitting *airfoil* dataset under different settings

Setting	$a_{Laplace}$	$a_{Gaussian}$	a_{Cauchy}
No outliers	0.4623	0.5124	0.0252
Outliers from [-4sd, -2sd] and [2sd, 4sd]	0.4015	0.5343	0.0641
Outliers from [-6sd, -3sd] and [3sd, 6sd]	0.0994	0.5598	0.3406
Outliers from [-10sd, -5sd] and [5sd, 10sd]	0.2669	0.5030	0.2299

Table 3.9 Mixture weights of the mixture model in fitting *abalone* dataset under different settings

3.5 Conclusion and Discussion

In this chapter, we derived a flexible method for fitting a mixture of m-estimators for robust regression task, as a unique version of the general EM algorithm. We demonstrated its estimation accuracy by comparing it with OLS, and some other popular m-estimators, under different scenarios of added outliers on real datasets. We also pointed out the potential use of these approach for making about the underlying nature of a given data as well.

We also proposed a modified mixture model which updates the scale parameters of the m-estimators along with the other parameters that are covered by its simpler version. This model is also solved by almost the same EM-based algorithm; the only difference is that, on top of the existing update rules for the parameters in the simpler model, we added update rules for scale parameters as well. According to our experiments, we observed that, even though this model is more flexible than the other ones, its performance is slightly worse than the "naive approach" which is based on detecting the outliers and removing them [1]. Therefore, we concluded that the fitting of scale parameters may lead to an overparameterization problem in cases where simpler models would be sufficient to estimate the underlying distribution of residuals. As a potential direction of research, one can investigate the ways of penalizing the effects of unnecessary parameters that are fitted by this modified model. For now, in real-life practices, we suggest tackling robust regression problems as follows: First, one can fit a standard OLS to the data and observe if there are any peculiarities (outliers, skewness, heteroscedasticity etc.) in the residuals. If so, one should incrementally add some robust m-estimators to the existing model by using our proposed method and measure its test-set performance (or CV error) at each incremental increase in its complexity. Then, one can stop adding new m-estimators when the test-set performance of the mixture model does not improve any further.

As we briefly mentioned before, robust regression and regularized regression problems are deeply connected to each other. As a possible line of research, one can consider developing an algorithm for handling these two problems simultaneously.

[1] This approach may seem appealing at first, in terms of improving predictive accuracy, but the observations with outliers may still contain useful information about the underlying structure of the process that generated the given data.

More precisely, these problems can be modeled together as a multi-objective optimization problem as in

$$\min_{\beta,W,\Lambda}(Y-X\beta)^T W(Y-X\beta)+\beta^T\Lambda\beta$$

where W and Λ are diagonal matrices that contain penalty weights for observations and features, respectively. To facilitate the unified solution of this problem, we can define a new feature matrix $\hat{X}=\begin{pmatrix} X \\ I_{m\times m}\end{pmatrix}$, a new response vector $\hat{Y}=\begin{pmatrix} Y \\ 0_{m\times 1}\end{pmatrix}$, and a new weight matrix $\hat{W}=diag([w_1,\ldots,w_n,\lambda_1,\ldots,\lambda_m])$. So, the re-modeled regression problem would be

$$\min_{\beta,\hat{W}}(\hat{Y}-\hat{X}\beta)^T\hat{W}(\hat{Y}-\hat{X}\beta)$$

.

After this re-modeling, one can consider solving this optimization problem for β and $\hat{W}$ by combining the methods that we proposed in Chapter 2 and the current chapter, as future work. By considering this combination, one can also develop a differentially private (DP) extension of this method. As we mentioned before in Chapter 2, the use of subsamples for these estimation tasks brings some inherent noise which can be controlled in a way to make these tasks satisfy DP constraints.

4. DIFFERENTIALLY PRIVATE FREQUENCY SKETCHES FOR INTERMITTENT QUERIES ON LARGE DATA STREAMS

We propose novel and differentially private versions of Count Sketch, particularly suited for dynamic, intermittent queries for observed frequencies of elements in a universal set. Our algorithms are designed for scenarios where the queries are made intermittently, that is, at different times during the course of the data stream. We explore several approaches, all based on the Laplace mechanism, and ultimately propose an algorithm that is robust and efficiently handles multiple queries at multiple times while keeping its utility at reasonable levels. We demonstrate the performance of the proposed algorithm in various scenarios with a numerical example.

As we mentioned before, this chapter is based on a conference paper in which the author of this dissertation is a co-author. The author of this dissertation (partially) contributed to Sections 4.1 , 4.2, 4.5, and to a lesser extent, Section 4.6. The rest of the research was conducted by the other co-authors (Sinan Yıldırım, Kamer Kaya, Hakan Buğra Erentuğ). For completeness, we present the material of the whole paper here.

4.1 Introduction

Sketches are probabilistic data structures that can provide approximate results within mathematically proven error bounds while using orders of magnitude less memory than traditional approaches. They are tailored for streaming data analysis on architectures even with limited memory such as single-board computers that are widely exploited for IoT and edge computing. With the emergence of massive-scale data streams in various fields, the concern of extracting useful information from these data streams while preserving the privacy of individual data becomes an important problem. Differential privacy (DP) provides a framework as a solu-

tion in which information about a data set is revealed while, at the same time, the privacy of the individuals that have contributed to the dataset is preserved Dwork (2006a). This is why leveraging DP for data sketches have been popular in the literature in the last decade Cormode et al. (2012); Dwork et al. (2010); Melis et al. (2016); Mir et al. (2011); Mishra & Sandler (2006); Sparka et al. (2018); von Voigt & Tschorsch (2019). In this chapter, we focus on developing differentially private sketches for count queries. We confine our study to the Count and Count-Min Sketches Charikar et al. (2002); Cormode & Muthukrishnan (2005). The Count Sketch is proposed in Charikar et al. (2002) as a useful tool for answering frequency queries, by producing unbiased estimators. Similarly, the Count-Min Sketch Cormode & Muthukrishnan (2005) is proposed for the same task. In the literature, there have been many studies on incorporating either Count Sketch, Count-Min Sketch, or both in a privacy-preserving setting.

In this chapter, we tackle the problem of answering intermittent frequency queries regarding the information contained in a data stream while providing DP and keeping the utility of the responses at a reasonable level. What is significant about the setting of "intermittent queries" is the possibility that even between two consecutive *identical* queries, the true answer may have differed due to addition of new individuals' data. The challenge related to data privacy is that the answers to the queries should continually protect the privacy of individuals' data that are included in the data stream at any time of the streaming process. Hence, the setting being investigated in this work can be considered as a generalization of the setting which focuses on one-time queries. Overall, the contributions can be summarized as follows:

- We discuss various methods for differentially private counting sketches for intermittent queries that can be considered as reasonable approaches under certain circumstances. However, as our ultimate choice, we propose a single method that we show to be most suitable.

- The algorithms we discuss and propose are based on two different competing approaches. The first approach considers randomly perturbing the cells in the sketch table, while the second group consider perturbing the answer returned from the sketch. We discuss the pros and cons of those approaches in terms of accuracy. We show that while both approaches can compete in a setting where a batch of queries are to be answered at the same time, the first approach is more suited to intermittent queries.

- Despite the algorithms in this chapter are presented with the Count Sketch, they can easily be adapted to the Count-Min Sketch, following almost identical steps.

The organization of this chapter is as follows: In Section 4.2, we review the definition and basic properties of DP and present the Count and Count-Min Sketches as event-oriented dynamic processes. The main tools and approaches for developing a differentially private Count Sketch are presented in Section 4.3. In Section 4.4, we present our differentially private sketch tailored for intermittent queries. Section 4.5 discusses the related work on sketches providing DP guarantees. In Section 4.6, we present an experimental study and subsequently comment on its results. In Section 5.8, we give conclusive remarks and mention some possible extensions of the methodology.

4.2 Background and Notation

In this section, we provide some background, along with some basic notation, for differential privacy and the count and count-min sketches.

4.2.1 Differential privacy

In recent years, *differential privacy (DP)* has become a popular framework for achieving privacy-preserving estimates of basic statistics in data sets. We call two data sets $X, X' \in \mathsf{X}$ neighbors if X' is obtained by the addition or deletion of a single entry to or from X. We call $\mathcal{A}$ a randomized algorithm whose output upon taking an input X is a random variable $\mathcal{A}(X)$ taking values from some $\mathcal{S}$.

Definition 2. *We say that $\mathcal{A}$ is (ϵ, δ)-differentially private if, for any pair of neighboring data sets $X, X' \in \mathsf{X}$ from an input set and any subset of output values $S \subseteq \mathcal{S}$, it satisfies Dwork (2006a)*

$$\mathbb{P}\left[\mathcal{A}(X) \in S\right] \leq e^{\epsilon}\mathbb{P}\left[\mathcal{A}(X') \in S\right] + \delta.$$

According to the above inequality, a randomized algorithm is differentially private if the probability distributions for the output obtained from two neighboring databases are '*similar*'. The parameters ϵ and δ determine the *privacy budget*, or *privacy loss*. Those parameters are desired to be as small as possible as far as privacy is concerned.

Assume that a privacy preserving algorithm is required to return the value of a function $\varphi : \mathsf{X} \mapsto \mathbb{R}$ evaluated at the sensitive data set X in a private fashion. One basic way of achieving this is via the *Laplace mechanism* (Dwork, 2008, Theorem 1), which relies on the *sensitivity* of this function.

Definition 3. *The sensitivity of* $\varphi : \mathsf{X} : \mathbb{R}$ *is given by*

$$\nabla_\varphi = \sup_{X, X' \in \mathsf{X}} |\varphi(X) - \varphi(X')|.$$

Theorem 1 (Laplace mechanism)**.** *Let* $\mathcal{A}$ *be an algorithm that returns* $\hat{\varphi} = \varphi(X) + v$ *on an input* $X \in \mathsf{X}$*, where* $v \sim \text{Laplace}(\nabla_\varphi / \epsilon)$*. Then* $\mathcal{A}$ *is* ϵ*-DP.*

One property of differential privacy is the composition property, which quantifies the privacy loss of multiple uses of a differentially private mechanism.

Theorem 2 (Composition)**.** *Let* $\mathcal{S}$ *be the output space and* X *be the input space. Let* $\mathcal{A}_1, \ldots, \mathcal{A}_m$ *are* ϵ*-DP algorithms that take* $X \in \mathsf{X}$ *as input and return random outputs in* $\mathcal{S}$ *as outputs. Then, let the composition of those algorithms,* $\mathcal{A} = (\mathcal{A}_1, \ldots, \mathcal{A}_m)$ *return outputs* $(S_1, \ldots, S_m)$ *is* $m\epsilon$*-DP. Moreover, this result does not change when* $(\mathcal{A}_1, \ldots, \mathcal{A}_m)$ *is applied sequentially and each algorithm's output is generated conditional on the outputs of the previous algorithms.*

Another property of differential privacy relevant to our work is that no further privacy loss is suffered by transforming the output through a deterministic algorithm.

Theorem 3 (Post-processing)**.** *Let* $\mathcal{A}_1$ *be an* (ϵ, δ)*-DP algorithm with inputs from* X *and outputs from* $\mathcal{S}_1$*, and let* $\mathcal{A}_2 : \mathcal{S}_1 \mapsto \mathcal{S}$ *be deterministic algorithm that does not depend on* X*. Then, the algorithm* $\mathcal{A} = (A_2 o \mathcal{A}_1)$ *is* (ϵ, δ)*-DP.*

4.2.2 Count and Count-Min Sketches

Suppose we have a data stream from a universal set $\mathcal{X}$ and its i'th element is x_i. Count and Count-Min Sketches are developed in order to answer queries regarding the count of an element $x \in \mathcal{X}$, i.e., its frequency, in such a data set. That is those sketches answer questions in the form of "*How many times has a certain element occurred so far in the data stream?*". The sketching algorithms summarize a large amount of data into a small table by the help of hash functions. Both Count and Count-Min Sketch use similar type of sketch tables.

Count-Min Sketch This sketch uses a table/matrix C with w columns and d rows where row i of the sketch table is governed by a hash function $h_i : \mathcal{X} \to \{1, \ldots, w\}$

which is chosen independently for each row. Let $C(i,j)$ be the value in the ith row and jth column. The table is initialized with all zeroes, i.e., $C(i,j)=0$ for all $1 \leq i \leq d$ and $1 \leq j \leq w$. Every time an element x from the stream is received, the sketch is updated according to the procedure given in Cormode & Muthukrishnan (2005):

$$C(i,h_i(x)) \leftarrow C(i,h_i(x))+1, \quad i=1,\ldots,d.$$

When an element x is queried, the sketch returns the estimation of x's frequency, i.e., f_x, as

$$\hat{f}_x = \min\{C(1,h_1(x)),\ldots,C(d,h_d(x))\}.$$

Count Sketch The difference of Count Sketch from Count-Min Sketch is that the contribution of each element to C is given as either $+1$ or -1 with equal probability, and the sign of increment is determined by using another hash function, shown as $s_i : \mathcal{X} \mapsto \{-1,1\}$. Accordingly, the estimate of f_x for an item x is given as the median of the values corresponding to x in C, instead of their minimum Charikar, Chen & Farach-Colton (2004).

The initialization of the table for Count Sketch is the same, that is, $C(i,j)=0$ for each $1 \leq i \leq d$, $1 \leq j \leq w$. Upon receipt of an element x, the sketch updates the table as

$$C(i,h_i(x)) \leftarrow C(i,h_i(x))+s_i(x), \quad i=1,\ldots,d.$$

When queried, the frequency f_x of any $x \in \mathcal{X}$ is estimated by

$$\hat{f}_x = \text{median}\{s_1(x)C(1,h_1(x)),\ldots,s_d(x)C(d,h_d(x))\}.$$

Although our differentially private sketching algorithms can be implemented with both sketches, we will only focus on the Count Sketch to avoid repetitions.

Dynamic system view with intermittent queries In this chapter, we are interested in a dynamic system, where queries can be made while the sketch is still being updated by the stream. Moreover, we assume that queries can be made in arbitrary times and for arbitrary subsets of $\mathcal{X}$ of arbitrary size. That is why we prefer to present a sketching algorithm as an event-oriented *dynamic system* that has two types of events: (i) arrival of an element from the stream, and (ii) a query set $Q \subseteq \mathcal{X}$ whose elements are queried for their frequencies. Algorithm 3 models such a dynamic system and employs a Count Sketch to answer intermittent query sets.

In this chapter, we investigate the ways to modify Algorithm 3 so that it would

Algorithm 3 Count Sketch in a dynamic system

```
{Initialization of the table}
for i = 1, ..., d do
    for j = 1, ..., w do
        Set C(i, j) ← 0
{Events start}
repeat
    if stream element x arrives then
        for i = 1, ..., d do
            C(i, h_i(x)) ← C(i, h_i(x)) + s_i(x)
    else if a set Q of queries are made then
        for x ∈ Q do
            Set S ← ∅
            for i = 1, ..., d do
                Append S with s_i(x)C(i, h_i(x))
            return f̂_x = median S
until end of events;
```

provide DP while keeping the accuracy at a reasonable rate. In the following section, we present a detailed discussion and results on several approaches.

4.3 Privacy-preserving count sketch

A privacy preserving algorithm protects privacy generally by adding noise to its calculations. When multiple queries are made about sensitive data, as the case in our setting, usually more noise is required not to exceed a privacy budget, see Theorem 2. As a result, the algorithm's utility inevitably deteriorates. Our main concern is to use noise adding mechanisms in an intelligent way so that the utility of the privacy preserving algorithm deteriorates as slowly as possible.

4.3.1 Setting

Here, we provide a setting that places the privacy issue in a certain context. For that, it is firstly necessary to define the neighborhood relations between streams. We assume that the stream elements which are processed by the Count Sketch correspond to distinct individuals, and the sketch is constructed to return the frequency values of individuals who are placed in a certain category with respect to a certain feature. For example, when the data stream is composed of individuals with their residence addresses (so $\mathcal{X}$ is a set of certain type of addresses, such as postal codes), the queries are in form of "*How many people reside in address x?*". We assume in this work that the hash functions used by the sketch are publicly known.

In this setting, two ordered data sets are neighbors if they have all the same elements in the same order, except for the presence or absence of a single element. $X = \{x_1, \ldots, x_n\}$ and $X' = \{x_1, \ldots, x_{k-1}, x_{k+1}, \ldots, x_n\}$ can be given as an example of such sets. The regular Count Sketch does not provide differential privacy for two neighboring data sets, because when a new data point arrives, the cells in C which correspond to this data point are increased or decreased by exactly 1. An '*unfortunate*' query will make this difference reflect on the median calculation, hence violate the privacy.

4.3.1.1 Single query and Laplace mechanism

Notice that the sensitivity of the sketch estimate $\hat{f}_x$ for any element $x \in \mathcal{X}$ is 1, since absence or presence of a single individual in the stream can change the estimate by at most 1. Therefore, when a single query for the frequency of an element $x \in \mathcal{X}$ is made, a simple distortion of the output by using a random value from Laplace$(1/\epsilon)$, yielding the noisy output

$$\hat{f}_x + v, \quad v \sim \text{Laplace}(1/\epsilon), \tag{4.1}$$

is sufficient for providing ϵ-differential privacy.

4.3.2 Multiple queries

Assume a set of queries $Q \subseteq \mathcal{X}$ is received and the frequency estimations for all the items in Q are requested. Such *multiple queries* can be quite relevant to practical

implementations. For example, a single user may want to know frequencies of a set of elements, which constitute a range or whom the user thinks are *heavy-hitters*, i.e., items that frequently appear in data streams. Alternatively, Q may be the union of different query sets made by different users, i.e.,

$$Q = \bigcup_i Q^{(i)}$$

where $Q^{(i)}$ is the set of elements whose frequencies are queried by user i. If the actual timestamps of those query sets are sufficiently close to each other, they can be considered as simultaneous. For example, the dynamic system may perform batching to answer queries and consider successive periods of a buffer time, Δ, where all the queries made within the same period are considered simultaneous.

4.3.2.1 Median perturbation

Let n_Q denote the size of the set Q. A straightforward application of the composition theorem for DP, Theorem 2, yields a query response for each element in Q that satisfies ϵ/n_Q-DP. This can be achieved by

$$\hat{f}_x = f_x + v_x, \quad v_x \sim \text{Laplace}(n_Q/\epsilon), \quad x \in Q,$$

where v_x is sampled independently for each $x \in Q$. While this basic approach is reasonable when n_Q is small, the estimates get unreliable quickly as n_Q increases. However, Laplace(n_Q/ϵ) may be an (unnecessarily) conservative choice for the amount of noise to be added. We explain why so in the following.

Let X, X' be neighbour data sets with the different element denoted by x^*. We will call $\{(1, h_1(x^*)), \ldots, (d, h_d(x^*))\}$ the set of *sensitive cells* for this pair X, X'. One can argue that it is possible that not all queries in Q require a *sensitive cell*. Given Q, we should instead add noise based on *the maximum possible number of sensitive queries*, i.e., queries for which at least one sensitive cell is to be used for the median calculation. Let this number be m_Q. Then, each query in Q can be responded by

$$\hat{f}_x = f_x + v_x, \quad v_x \sim \text{Laplace}(m_Q/\epsilon), \quad x \in Q. \tag{4.2}$$

providing ϵ-DP by the composition theorem, Theorem 2.

We provide a proposition below that characterizes m_Q. Let $j^x_{1:d} = (j^x_1, \ldots, j^x_d)$ be the

vector of column indices of the table C's cells for x, i.e., the hash functions on x yield reads from cells $(1, j_1^x), \ldots, (d, j_d^x)$ in C. The number m_Q can be written as[1]

$$m_Q = \max_{j_{1:d} \in \{1,\ldots,w\}^d} \sum_{x \in Q} [(j_1 = j_1^x) \vee \ldots \vee (j_d = j_d^x)] \tag{4.3}$$

The problem of finding m_Q in (4.3) can be shown to be a generalized version of maximum satisfiability (MAXSAT) problem, where the variables are non-binary but integers, and therefore, it is NP-hard. For moderate values of n_Q, one can compute m_Q with a brute-force grid search over a maximum of $(n_Q)^d$ combinations for $j_{1:d}$ in (4.3). However, for large n_Q, we should resort to an approximation to find m_Q, which has to be an upper bound in order to satisfy ϵ-DP. An upper bound can be computed as in Algorithm 4.

Algorithm 4 An upper bound estimate for m_Q

Input: Query set Q
Output: Estimate of m_Q, $\hat{m}_Q$
{Initialize an auxiliary $d \times w$ table N}
$N(i,j) = 0$ for all $1 \le i \le d$, $1 \le j \le w$.
for $i = 1, \ldots, d$ **do**
  Compute the maximum frequency for the i'th row: **for** $x \in Q$ **do**
    Increment $N(i, h_i(x)) = N(i, h_i(x)) + 1$.
  Calculate $m_i = \max_{j=1,\ldots,w} N(i,j)$.
return $\hat{m}_Q = \sum_{i=1}^d m_i$.

Proposition 1. *We have $m_Q \le \hat{m}_Q$, where $\hat{m}_Q$ is computed in Algorithm 4. Furthermore, if $n_Q \le d$ we have $m_Q = n_Q$.*

Proof. Let $j_i^* = \arg\max_{j=1,\ldots,w} N(i,j)$, where $N(i,j)$ is calculated in Algorithm 4. The worst case scenario, that is the scenario where a maximum number of the elements in Q touch a sensitive cell, is realized when $(1, j_1^*), \ldots, (d, j_d^*)$ are the sensitive cells, there are m_i queries in Q that touch the cell (i, j_i^*), for $i = 1, \ldots, d$, and those queries are all distinct (that is, no query touches more than one sensitive cell). This leads to $m_Q = \hat{m}_Q$. In the case of $n_Q \le d$, it is trivial by definition that we have $m_Q \le n_Q$. For the equality itself, letting $x_1, \ldots, x_{n_Q}$ be the elements in Q, note that one can always select $j_1, \ldots, j_d$ such that j_i coincides with $h_i(x_i)$ for $i = 1, \ldots, n_Q \le d$, making $m_Q = n_Q$. □

The expected value of $\hat{m}_Q$ in Algorithm 4 can be computed exactly by using the method in Freeman (1979), upon observing that m_i in Algorithm 4 is the highest

[1]Strictly speaking, (4.3) is an upper bound, with equality holding if for all $j_{1:d}$ satisfying the condition in (4.3), there is an $x \in \mathcal{X}$ such that $j_{1:d}^x = j_{1:d}$.

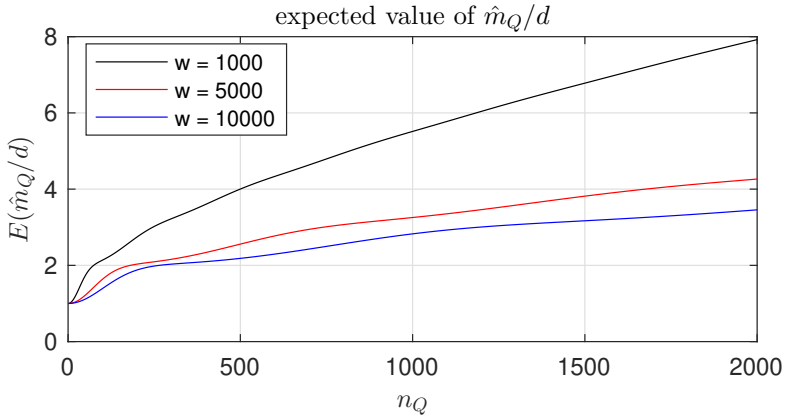


Figure 4.1 $E(m_i) = E(\hat{m}_Q/d)$ vs query size n_Q, where m_i is given in Algorithm 4. Observe the sub-linear increase.

frequency for a multinomial sample with size n_Q and w bins. From the properties of m_i, m_Q can be claimed to be much less than n_Q when n_Q is large. This is demonstrated in Figure 4.1 which shows $\mathbb{E}(\hat{m}_Q)$ computed by using the recursion in Ramakrishna (1988).

4.3.2.2 Cell perturbation

While m_Q can be significantly smaller than n_Q, by Proposition 1, when $n_Q \geq d$, the parameter of the Laplace noise is d/ϵ at best.

An effective alternative to the median perturbation approach is observed in Melis et al. (2016). This approach is based on adding noise to the C's cells once and for all. Observe that there are only d cells in C that are affected by a presence/absence of an item in the stream. Namely, construct the noisy sketch

$$\tilde{C}(i,j) = C(i,j) + v_{i,j}, \quad v_{i,j} \overset{\text{i.i.d.}}{\sim} \text{Laplace}(d/\epsilon). \tag{4.4}$$

and answer all the queries by using the noisy sketch $\tilde{C}$. That is, for all the elements $x \in Q$, we return

$$\hat{f}_x = \text{median}\{\tilde{C}(1,h_1(x))s_1(x), \ldots, \tilde{C}(d,h_d(x))s_d(x)\} \tag{4.5}$$

In $\tilde{C}$, each of those sensitive cells are corrupted by a Laplace(d/ϵ) noise, which preserves ϵ/d privacy. By the composition theorem, the privacy level of constructing $\tilde{C}$ as such is ϵ. Moreover, by the post-processing theorem for DP, returning any number of query responses calculated from $\tilde{C}$ as in (4.5), which is a deterministic

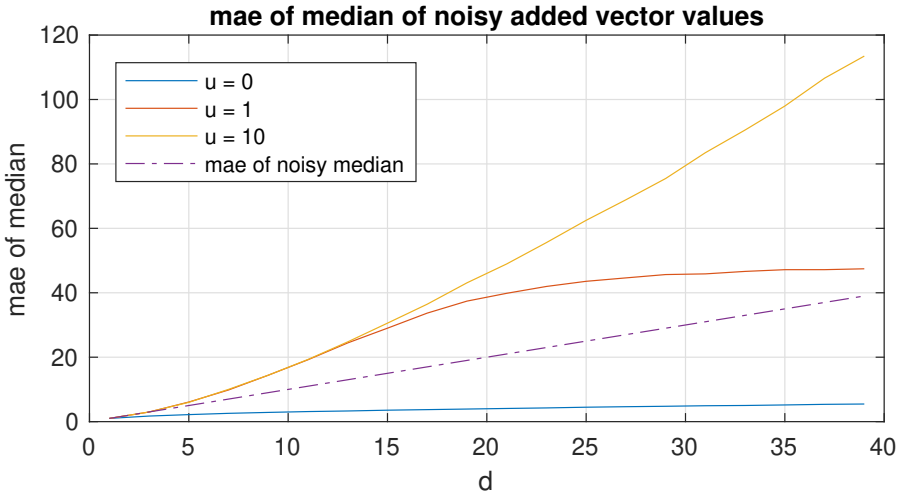


Figure 4.2 A comparison between the mean absolute values of noise added median of the cell values (median perturbation) and the median of noisy cell values (cell perturbation).

operation given $\tilde{C}$, is also ϵ-DP. In fact, it suffices to add noise to the cells that are relevant to the queries in Q. $C(i,j)$ is added noise if there is an $x \in Q$ such that $h_i(x) = j$. While this is a straightforward observation, it plays an important role in case of intermittent queries.

4.3.2.3 A comparison

A natural question is which method is better over the other one; the median perturbation in (4.2), or cell perturbation in (4.4) followed by (4.5)? Some probabilistic error bounds are available for both approaches; see Theorem 10 regarding cell perturbation and the theorems in Appendix B.1.2 for median perturbation. Since those bounds only enable a qualitative elaboration, we instead resort to a numerical comparison and show that one method is not uniformly better than the other, and the outcome of the comparison depends on d, m_Q, and the variability among the true frequency values in the d cells to be used for the median calculation.

Figure 4.2 compares both methods for a hypothetical query set Q of size $n_Q \geq d$ (so that it is not too small) and assumes the best scenario $m_Q = d$ for the median perturbation method. We made this assumption so that the same amount of noise, drawn from Laplace(d/ϵ), is added to the median in the output perturbation method and to each of the cells in the cell perturbation method. Under this setting, we compare the performances of $\hat{f}_x$ in (4.2) and $\hat{f}_x$ in (4.5), when both provide ϵ-DP. The non-noisy values that are subject to the median calculation are designed as

$(0,\ldots,0,u,\ldots,u)$ where there are $(d+1)/2$ zeroes and $(d-1)/2$ u's, so that the median is always 0 and u resembles the amount of variation among the cell values.

The plots suggest that one method is not uniformly better than the other; and the first method can be more advantageous when the variation is higher. However, note that the assumption $m_Q = d$ may be a big favor for the first method and may not be even close to the truth when n_Q is large; see Figure 4.2.

While the approach of adding noise to C once and for all, or shortly the cell perturbation approach, is quite efficient when multiple queries are made at the same time, it is not tailored for a dynamic system where different query sets have to be handled at different times. This latter challenge, by which this work is motivated, is tackled in the next section.

4.4 Queries at different times

Suppose we have a stream of data and query sets are made at different times, while the sketch is still being updated by the stream. Suppose query sets $Q_1, Q_2, \ldots$ are made at times $t_1 < t_2 < \ldots$. Crucially, new elements from the stream may arrive between successive query times. That is why the sketch and/or the noisy outputs of the previous time are not up-to-date. Here, we will analyze two classes of methods; *use-and-forget* and *use-and-keep*.

4.4.1 Use-and-forget methods

We first describe two methods, in both of which the sketch is carried on exactly and fresh noise is added to the medians or the cells themselves at each time t_k.

4.4.1.1 Median perturbation

The first method is based on median perturbation. As the total number of times a query is made is unknown beforehand, we can preserve ϵ-DP by preserving ϵ_k-DP for query time t_k, where ϵ_k is the k'th member of a geometric series,

$$\epsilon_k = \epsilon(1-\eta)\eta^{k-1}, \quad k = 1, 2, \ldots. \tag{4.6}$$

for some $\eta < 1$. Therefore, one way to preserve ϵ_k-DP privacy for the k'th set of queries, Q_k is made at time t_k. However, this scheme is hardly useful since the geometric increase in the amount of noise will soon destroy the accuracy in the responses.

4.4.1.2 Cell perturbation

An alternative method can be built up on an extension of the method for adding noise to the cells themselves, which allows a noisy cell value to be used more than one queries made at the same time. Since there are d sensitive cells, each has to be protected with ϵ/d-DP. For this, we need to keep track on the number of times each cell is used in median calculation. Let that information be kept in a $d \times w$ matrix U. Specifically, assume that x is queried and

$$(1, j_1^x), (2, j_2^x), \ldots, (d, j_d^x)$$

are the indices of the cells of the sketch table x is hashed. The numbers of times the those cells have been used is given by

$$U(1, j_1^x), U(2, j_2^x), \ldots, U(d, j_d^x).$$

Then, the algorithm uses noisy cell values $\tilde{C}(i, j_i^x) = C(i, j_i^x) + V_i$, where $V_i \overset{\text{i.i.d.}}{\sim}$ Laplace(d/ϵ_i') is adjusted to preserve ϵ_i'-DP, which be chosen as

$$\epsilon_i' = \frac{\epsilon}{d}(1-\eta)\eta^{U(i, j_i^x)}, \quad i = 1, \ldots, d.$$

The noisy counts are then used to produce the final answer as

$$\text{median}\{\tilde{C}(1, j_1^x)s_1(x), \ldots, \tilde{C}(d, j_d^x)s_d(x)\}.$$

Again, the reason for division by d is due to the existence of d sensitive cells. This algorithm guarantees that at the k'th use of the cell, we provide ϵ_k-DP. In total,

each sensitive cell's information is protected with $\sum_k \epsilon_k = \epsilon/d$-DP, yielding ϵ-DP in total.

This approach can be implemented as in Algorithm 5. Note that any noise added to the cells of C is thrown away after being used in the median calculation. When a next set of queries is received at a future time, a fresh noise has to be added to the relevant cells. Because of that, we call this approach the use-and-forget approach.

Algorithm 5 Use-and-forget DP Count Sketch

{Initialization of the table}
for $i = 1, \ldots, d$ **do**
  for $j = 1, \ldots, w$ **do**
    Set $C(i,j) \leftarrow 0$ and $U(i,j) \leftarrow 0$.
{Events start}
repeat
  if *stream element* x *arrives* **then**
    for $i = 1, \ldots, d$ **do**
      $C(i, h_i(x)) \leftarrow C(i, h_i(x)) + s_i(x)$.
  else if *a set* Q *of queries are made* **then**
    Set $E = \emptyset$.
    for $x \in Q$ **do**
      Set $S \leftarrow \emptyset$.
      for $i = 1, \ldots, w$ **do**
        Set $j = h_i(x)$.
        if $(i,j) \notin E$ **then**

$$\epsilon' = \frac{\epsilon}{d}(1-\eta)\eta^{U(i,j)}$$
$$v \sim \text{Laplace}(1/\epsilon')$$
$$\tilde{C}(i,j) = C(i,j) + v,$$
$$U(i,j) \leftarrow U(i,j) + 1$$
$$E \leftarrow E \cup \{(i,j)\}$$

        Append S with $\{\tilde{C}(i,j)s_i(x)\}$.
      return $\hat{f}_x = \mathit{median}(S)$.
until *end of events*;

4.4.2 Use-and-keep methods

The use-and-forget approach is designed to accommodate the requirement that the exact sketch be kept without corruption; observe that the non-noisy sketch table C

is carried on in Algorithm 5. Since each noisy value leaks information about the true value, the algorithm has to apply a noise schedule with a geometrically increasing variance, from which it will eventually suffer.

When the sketch table need not be kept as exact, an alternative scheme is possible with superior statistical properties. This alternative scheme is based on the observation that the different item between X and X' can appear only in one of the intervals $(0, t_1], (t_1, t_2], \ldots$, where we recall that t_i is the time of the query set Q_i is made. Before explaining our new scheme, we will present a lemma on the DP of a very simple counting algorithm; a proof is given in Appendix B.1.

Lemma 1. *Suppose we have a simple counter, c, that counts the number of occurrences of a certain value, x, in a stream. Let the times of queries about c be $t_1, t_2, \ldots$. Also, assume a noisy counter, $\tilde{c}$, which is incremented by 1 on every occurrence of x, just like c, however with the difference that at the time t_i of each query it is updated as*

$$\tilde{c} \leftarrow \tilde{c} + v_i, \quad v_i \overset{i.i.d.}{\sim} \text{Laplace}(1/\epsilon).$$

A mechanism that returns $\tilde{c}$ at each query time $t_1, t_2, \ldots$ immediately after the noise adding step is ϵ-DP.

The theorem can easily be applied to the cells of the sketch table. Namely, one can keep the noise in the cells and carry on sketching the stream with the noisy sketch table. When a cell that has been used before (and hence it is noisy) is to be used for the next time, an independent Laplace(d/ϵ) noise is added to the current value of the cell. The new noisy value goes in the calculations needed to estimate the count, and it is kept as the current value of the cell.

With the new scheme, the total noise on each cell is the sum of k independent Laplace(d/ϵ) noises, where k is the number of times the value of that cell is used. Therefore, the accumulation of noise on a cell value is much less severe than the noise in Algorithm 5 whose variance geometrically increases in k. With this motivation, we present Algorithm 6. Furthermore, Algorithm 6 does not need to know how many times a cell is used before, therefore does not need to keep a matrix such as U in Algorithm 5. At each query x, each cell which x is hashed is used after its value is updated with the addition of an independent Laplace(d/ϵ) noise.

Differential privacy property of Algorithm 6 is indicated in the following theorem.

Theorem 4. *Algorithm 6 provides ϵ-differential privacy.*

Proof. First, note that, for queries made at the same time, a cell is added noise at most once. Secondly, for queries made at multiple times, we update the noisy values by merely adding an independent noise to the noisy count, therefore imitating

Algorithm 6 Use-and-keep DP count sketch

{Initialization of the table}
for $i = 1, \ldots, d$ **do**
  for $j = 1, \ldots, w$ **do**
    Set $C(i,j) \leftarrow 0$.
{Events start}
repeat
  if *stream element* x *arrives* **then**
    for $i = 1, \ldots, d$ **do**
      $C(i, h_i(x)) \leftarrow C(i, h_i(x)) + s_i(x)$.
  else if *a set* Q *of queries are made* **then**
    Initialize $E = \emptyset$
    for $x \in Q$ **do**
      Set $S = \emptyset$.
      for $i = 1, \ldots, d$ **do**
        Set $j = h_i(x)$.
        if $(i,j) \notin E$ **then**
          $v \sim \text{Laplace}(d/\epsilon)$
          $C(i,j) \leftarrow C(i,j) + v$
          $E \leftarrow E \cup \{(i,j)\}$
        Append S with $\{C(i,j)s_i(x)\}$.
      return $\hat{f}_x = median(S)$.
until *end of events*;

the counting process in Lemma 1. This ensures that the value of each used cell in the sketch table $C(i,j)$ is protected with ϵ/d privacy throughout the whole process. Next, observe that there are only d sensitive cells: Letting X and X' be two neighboring data sets, and the differing element be x, the indices of C where X and X' differ after using the same hash functions are

$$I_x = \{(1, h_1(x)), \ldots, (d, h_d(x))\}.$$

Therefore, the total privacy is $d\epsilon/d = \epsilon$, by the composition theorem, Theorem 2. □

4.4.2.1 Error analysis for the use-and-keep algorithm

We can quantify the error in the responses of Algorithm 6 with the following theorem. A proof is given in the Appendix.

Theorem 5. *Assume that Algorithm 6 is run with d rows and w columns and the required privacy level is ϵ. Suppose that a query is made for an element x, and for $1 \leq i \leq d$, the cell that it is hashed to in the i'th row has been used $u_i - 1$ times prior to the query. Then, for any $\kappa^2 > 2\max_{i=1,\ldots,d}\left(\frac{||f||_2^2}{w} + \frac{d^2}{\epsilon^2}u_i\right)$, we have*

$$\mathbb{P}(|\hat{f}_x - f_x| > \kappa) \leq e^{-\frac{d}{2}(1-2\lambda)}[2(1-\lambda)]^{d/2}$$

where $\lambda = \frac{||f||_2^2}{\kappa^2 w} + \frac{d}{\kappa^2\epsilon^2}\sum_{i=1}^{d} u_i$, and $||f||_2^2 = \sum_x f_x^2$.

Theorem 5 has two important implications.

- Fix x, the element to be queried. Fix a small probability ρ and let λ_ρ be such that $e^{-\frac{d}{2}(1-2\lambda^*)}[2(1-\lambda^*)]^{d/2} = \rho$. Then, the absolute error which is exceeded with probability ρ is $\left(\frac{||f||_2^2}{\lambda_\rho w} + \frac{d}{\lambda_\rho\epsilon^2}\sum_{i=1}^{d} u_i\right)^{1/2}$. This grows sub-linearly with $\sum_{i=1}^{d} u_i$, the sum of the total number of uses of the cells that x is hashed to. This suggests that, if x is queried as frequently as its occurrence in the data stream (and similarly for the other elements that are also hashed in the same cells as x), we expect the relative error $\hat{f}_x$ to decrease in time.
- The error bound in the theorem indicates the two sided effect, in terms of performance, of the number of rows d in the sketch table. Observe that the error probability may not be monotonic in d; instead, increasing d up to a certain value may improve the error bound until worsening it beyond that value. This can be explained as follows: while a larger d helps the accuracy of the median, it also corrupts the cell values with more noise. Therefore, the optimum value d in terms of performance is in general somewhere in the middle.

4.5 Related Work

The related research in the literature can be grouped into three categories: (i) those that modify the standard count-based sketches to make them privacy-preserving, (ii) those that combine count based sketches and privacy-preserving mechanisms as separate steps of a master algorithm, (iii) those that argue that the inherent noise of the count-based sketches provides privacy under certain special conditions. These works are generally based on using the Laplace mechanism or its variations. As an

example of (i), Count Sketch and Count-Min Sketch are used for the succinct representation of user data in several different settings (such as recommendation systems, user location prediction, etc.) and Laplace noise is used to enhance the privacy of estimates obtained from these sketches Melis et al. (2016). There exist examples for (ii) which utilize Count (or Count-Min) Sketches and privacy-preserving mechanisms as distinct sub-procedures of a master algorithm, without modifying these sketches Monreale, Wang, Pratesi, Rinzivillo, Pedreschi, Andrienko & Andrienko (2013). Similarly, Cormode et al. (2012) use the Geometric mechanism (a discrete analog of the Laplace mechanism) to perturb the inputs that are later passed into Count Sketch (or alternatively, other summarisation methods which are based on sampling and filtering) . Hence, they do not incorporate this mechanism directly into the sketch; they use it in the preliminary steps of their main algorithms. Aside from these, Balu & Furon (2016) and Li, Liu, Sekar & Smith (2019), which are in category (iii), the Count Sketch is argued to be inherently providing DP in a special setting, where the inputs of the sketch are the gradient updates of an optimization problem. Under some strict assumptions given in those studies, such as the inputs of sketches being Gaussian distribution and their norms being bounded with high probability; Count Sketch satisfies DP by itself. But the authors still use output perturbation (by either adding Laplace noise to gradient updates or clipping the log-likelihood function of the optimization problem) when the inherent noise of sketch is not sufficient to provide ϵ-DP. Also, the assumptions that are used in these studies are not applicable when the input is frequency data which is discrete, nonnegative, and in the case of data streams, where the distribution is generally observed to be a power-law distribution.

All of these existing approaches mentioned above can be thought of as addressing the problem of answering one or multiple queries *at a single time* in a privacy preserving way. Our primary focus is protecting privacy when the data in question are in the form of a data stream that is formed by sensitive information from individuals, and queries are made dynamically and intermittently. The concept of providing privacy under continual observation was originally laid out in Dwork, Naor, Pitassi & Rothblum (2010) where several algorithms were proposed for that objective as well. Although counters, or, more generally, statistics monotonic in time, are considered in Dwork et al. (2010), their methods can be adopted for the cells of the Count Sketch. However, those methods require additional memory.

4.6 Experiments

In this section, we focus on the performance of Algorithm 6. We generated a data stream of size 2^{27} where each item is a random draw from the Zipfian distribution with domain $\{1,\dots,2^{29}\}$ and parameter α, so that $p(X=k) \propto k^{-\alpha}$ for $k=1,\dots,2^{29}$. We tested Algorithm 6 under several scenarios constructed by the combination of several parameters. These parameters are as follows:

- privacy level $\epsilon \in \{0.01, 0.05, 1\}$;
- $\alpha = 1$ of the Zipfian;
- the number of rows $d \in \{3, 9, 15\}$;
- the number of columns $w = 5000$;
- query set size $n_Q \in \{1, 10^3, 10^4, 10^5, 10^6\}$;
- user type parameter $u \in \{50, 5000, 500000\}$ which denotes that the query items are coming from the top u most frequent items.

For each combination of $(\epsilon, \alpha, d, w, n_Q, u)$ with the components taking values in their ranges stated above, we simulated the dynamic system in Algorithm 6. In each simulation, the data stream is generated gradually with the arrival of an element with regular time intervals. Along with the arrivals, queries are scheduled periodically at evenly spaced times. The periods for queries are arranged in such a way that the total number of queried elements is the same and equal to 10^6 for all the scenarios. That is, the period for queries when $n_Q = 10$ is 10 times the period for queries when $n_Q = 1$. This is ensured for fair comparison of scenarios in terms of accuracy. For each scenario, the queries start after 5×10^5 items arrive. Finally, all the scenarios are simulated 20 times with the same data stream but independent random seeds to generate the hash functions, query sets, and the Laplace noises. The average errors over those independent simulations are reported. Figure 4.3 shows the plots for the cumulative mean relative error vs time for combinations of ϵ, n_Q. For each combination, error plots for different values of d are superimposed. All the other parameters are averaged out.

Comparing the plots in Figure 4.3 along each parameter is informative: First, as expected, the error increases with decreasing ϵ. Second, we have smaller errors as n_Q increases. This is because answering a set queries at the same time is more beneficial than answering parts of them at separate times, since the former has the advantage of using the same cell noise for more queries.

A further deduction from Figure 4.3 is that the best value of d, among the ones compared, decreases as the scenario becomes more challenging. Observe, e.g., the

last column, where $\epsilon = 0.01$. As n_Q decreases, which necessitates more frequent noise adding, the value of d that gives the minimum error also decreases: it changes from $d = 15$ to $d = 9$ (at $n_Q = 10000$) and then to $d = 3$ as n_Q further decreases. This may be explained as follows: Smaller n_Q implies more frequent use of a single cell. Recall that we add a Laplace(d/ϵ) noise to a cell value each time the cell is required, making the cells noisier. There is a certain value of d beyond which the effect of taking the median over d rows is overwhelmed by the error due to the amount of noise added to the cells. This value of d is smaller when the cells are noisier (n_Q is smaller). This transition of the best d towards small values occurs earlier for smaller ϵ, since smaller ϵ means a more noisy sketch table.

Figure 4.4 shows plots for cumulative mean relative errors vs time for combinations of ϵ, u and with plots for different values of d superimposed. All the other parameters are averaged out. Observe that the relative error is larger as the queries are made from a larger pool of top elements. This behavior is also typical of the regular Count Sketch. Besides that, similar conclusions can be drawn about the behavior of the error curves as d changes. Once again, smaller values d is preferred as the cells get more noisy.

In both Figure 4.3 and Figure 4.4, we observe the cumulative relative error (left) grows sub-linearly. This supports our claim that in Section 4.4.2.1 stemming from Theorem 5 that the absolute error is expected to grow if queries are made as frequently as the arrival of new elements to the stream.

4.7 Conclusion

In this chapter, we proposed differentially private versions of the count sketch for frequency estimation. We both discussed median perturbation and cell perturbation methods. While in the static case, where all queries are made at the same time, it is not certain which method will prevail, for the dynamic case we propose using the cell perturbation technique as it is able to produce less noisy estimates. The ultimate algorithm proposed for the dynamic case was Algorithm 6, where the noise is used and kept in the cell value so that the DP noise variance increases only linearly, as opposed to geometrically which would happen with a naive implementation of the composition theorem for differential privacy.

Possible extensions to the proposed approach are as follows:

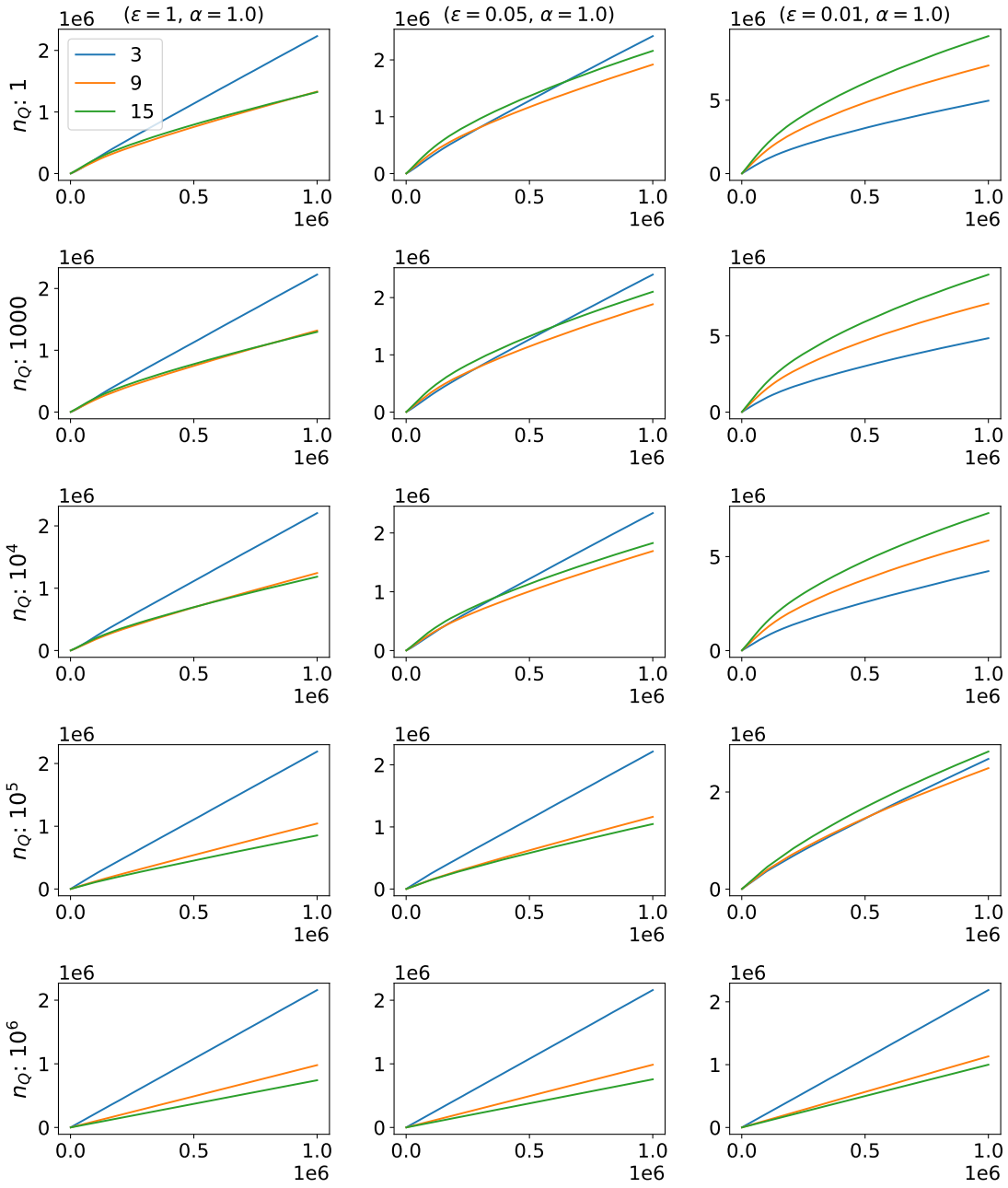


Figure 4.3 Cumulative mean relative error of Algorithm 6 for different combinations of ϵ, n_Q and d.

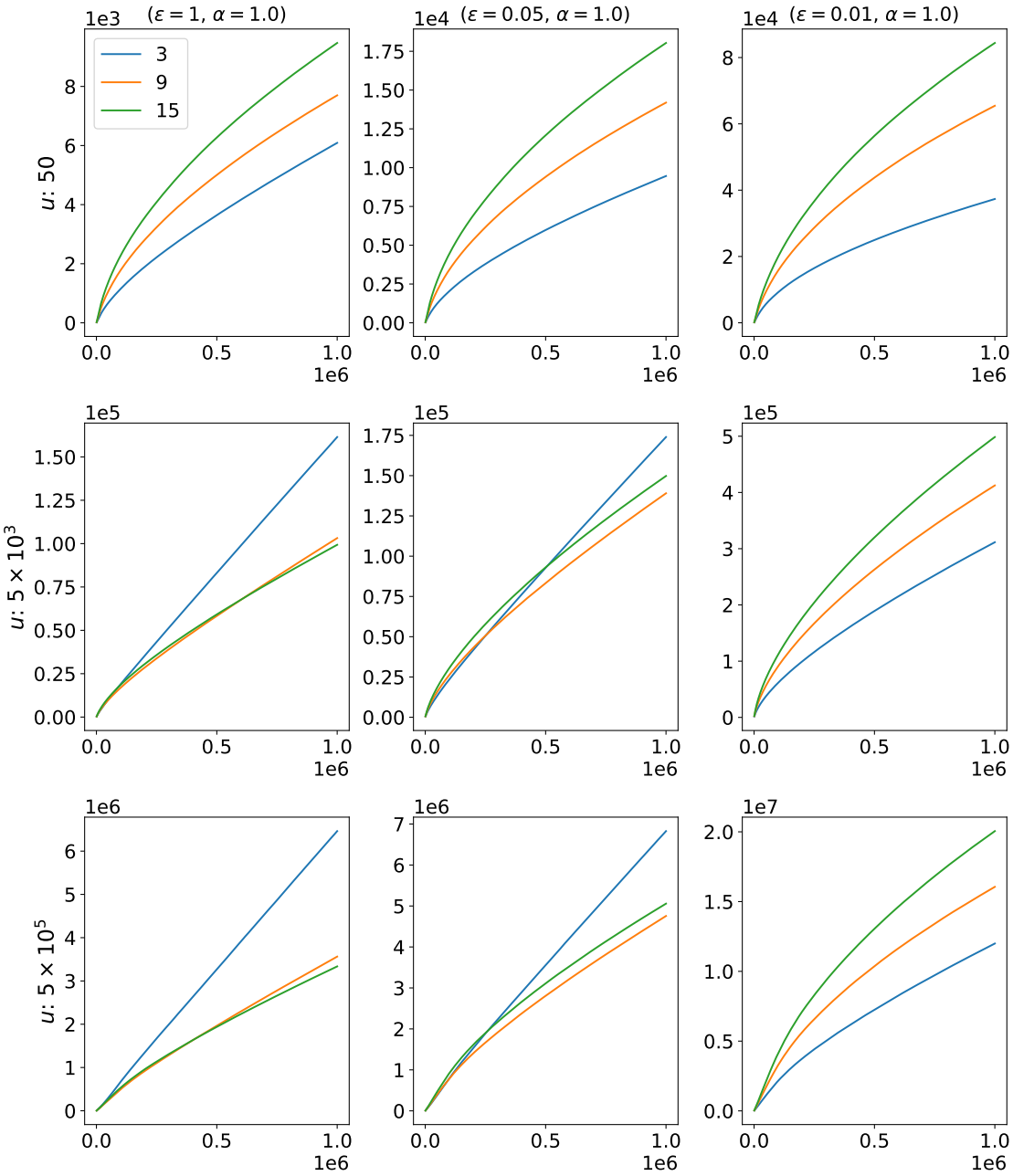


Figure 4.4 Cumulative mean relative error of Algorithm 6 for different combinations of ϵ, u and d.

Subsampling It is possible to increase the privacy level of the algorithms by deciding to process each element in the stream with a certain probability $0 < \rho < 1$ Balle, Barthe & Gaboardi (2018). This increases the privacy level by reducing ϵ to roughly an order of $\rho\epsilon$ for small ρ, see Balle et al. (2018) for the exact expression. The final estimates are then to be rescaled by $1/\rho$ to preserve unbiasedness. This comes, of course, with the expense of reducing the accuracy by means of multiplying the variance of the estimates by $1/\rho^2$.

Algorithms for Count-Min Sketch Just like Count Sketch, Count-Min Sketch can be modified in a similar fashion to provide privacy. Note that like the Count Sketch, the sensitivity of each cell and the minimum of those cells is 1. In our implementations, we used only Count Sketch, since it provides an unbiased estimator of the true frequency values, unlike Count-Min Sketch which overestimates the true counts.

Pan-privacy One direction for future research can be to modify the use-and-keep approach so as to satisfy pan-privacy Dwork et al. (2010). It is not difficult to satisfy pan-privacy against a single intrusion (into the state of the algorithm, which is the sketch C) followed by a set of queries made at a single time. This is achieved by simply starting C with $C(i,j) \overset{\text{i.i.d.}}{\sim} \text{Laplace}(d/\epsilon)$. However, providing pan-privacy against multiple intrusions seems difficult and needs more investigation.

Alternatives to the Laplace mechanism More advanced techniques than the Laplace mechanism for preserving the same amount of privacy were offered in Cormode, Kulkarni & Srivastava (2017). We remark that the choice for the probability distribution of the DP noise is not the main focus of our algorithm. Any noise adding technique is equally applicable and can substitute the Laplace mechanism as long as providing the same level of privacy. Moreover, light-tailed noise mechanisms such as the Gaussian mechanism can be considered instead of Laplace mechanism if one is willing to weaken the privacy requirements (and welcome $\delta > 0$).

Bayesian extension Differentially private estimation and randomized answering of frequency queries can also be handled by using nonparametric Bayesian approaches as well. For this purpose, a *Dirichlet process* is a natural candidate; because, in our case, the number of categories is not known beforehand and it often has a highly skewed distribution in real applications. One can use the Count Sketch's hash table for estimating the "pseudocount" hyperparameter (α) of a Dirichlet process, and then draw a sample of categorical probability vector (θ) from this process to use it as a parameter for multinomial distribution for answering frequency queries.

Since this approach involves a sampling step, the variance of the sample can be preadjusted in order to satisfy differential privacy constraint.

5. BAYESIAN FREQUENCY ESTIMATION UNDER LDP WITH AN ADAPTIVE RANDOMIZED RESPONSE MECHANISM

Frequency estimation plays a critical role in many applications involving personal and private categorical data. Such data are often collected sequentially over time, making it valuable to estimate their distribution online while preserving privacy. We propose AdOBEst-LDP, a new algorithm for adaptive, online Bayesian estimation of categorical distributions under local differential privacy (LDP). The key idea behind AdOBEst-LDP is to enhance the utility of future privatized categorical data by leveraging inference from previously collected privatized data. To achieve this, AdOBEst-LDP uses a new adaptive LDP mechanism to collect privatized data. This LDP mechanism constrains its output to a *subset* of categories that 'predicts' the next user's data. By adapting the subset selection process to the past privatized data via Bayesian estimation, the algorithm improves the utility of future privatized data. To quantify utility, we explore various well-known information metrics, including (but not limited to) the Fisher information matrix, total variation distance, and information entropy. For Bayesian estimation, we utilize *posterior sampling* through stochastic gradient Langevin dynamics, a computationally efficient approximate Markov chain Monte Carlo (MCMC) method.

We provide a theoretical analysis showing that (i) the posterior distribution of the category probabilities targeted with Bayesian estimation converges to the true probabilities even for approximate posterior sampling, and (ii) AdOBEst-LDP eventually selects the optimal subset for its LDP mechanism with high probability if posterior sampling is performed exactly. We also present numerical results to validate the estimation accuracy of AdOBEst-LDP. Our comparisons show its superior performance against non-adaptive and semi-adaptive competitors across different privacy levels and distributional parameters.

As we mentioned before, this chapter is based on a published paper in which the author of this dissertation is one of the co-authors (along with Sinan Yıldırım). The proofs in the Appendix of this chapter were done by Sinan Yıldırım. The other parts of this work were made by the combined efforts of both co-authors.

5.1 Introduction

Frequency estimation is the focus of many applications that involve personal and private categorical data. Suppose a type of sensitive information is represented as a random variable X with a categorical distribution denoted by $\text{Cat}(\theta)$, where θ is a K-dimensional probability vector. As real-life examples, this could be the distribution of the types of a product bought by the customers of an online shopping company, responses to a poll question like "Which party will you vote for in the next elections?", occupational affiliations of the people who visit the website of a governmental agency, and so on.

In this chapter, we propose an adaptive and online algorithm to estimate θ in a *Local Differential Privacy* (LDP) framework where X is unobserved and instead, we have access to a randomized response Y derived from X. In the LDP framework, a central aggregator receives each user's randomized (privatized) data to be used for inferential tasks. In that sense, LDP differs from global DP (Dwork, 2006b) where the aggregator privatizes operations on the sensitive dataset after it collects the sensitive data without noise. Hence LDP can be said to provide a stricter form of privacy and is used in cases where the aggregator may not be trustable (Kasiviswanathan et al., 2011). Below, we give a more formal definition of ϵ-LDP as a property that concerns a randomized mechanism.

Definition 4 (Local differential privacy)**.** *A randomized mechanism $\mathcal{M} : \mathcal{X} \mapsto \mathcal{Y}$ satisfies ϵ-LDP if the following inequality holds for any pairs of inputs $x, x' \in \mathcal{X}$, and for any output (response) $y \in \mathcal{Y}$:*

$$e^{-\epsilon} \leq \frac{\mathbb{P}(\mathcal{M}(x) = y)}{\mathbb{P}(\mathcal{M}(x') = y)} \leq e^{\epsilon}.$$

The definition of LDP is almost the same as that of global DP. The main difference is that, in the global DP, inputs x, x' are two datasets that differ in only one individual's record, whereas in LDP, x, x' are two different data points from $\mathcal{X}$.

In Definition 4, $\epsilon \geq 0$ is the privacy parameter. A smaller ϵ value provides stronger privacy. One main challenge in most differential privacy settings is to decide on the randomized mechanism. In the case of LDP, this is how an individual data point X should be randomized. For a given randomized algorithm, too little randomization may not guarantee the privacy of individuals, whereas too severe randomization deteriorates the utility of the output of the randomized algorithm. Balancing these conflicting objectives (privacy vs utility) is the main goal of the research on estima-

tion under privacy constraints.

In many cases, individuals' data points are collected sequentially. A basic example is opinion polling, where data is collected typically in time intervals of lengths in the order of hours or days. Personal data entered during registration is another example. For example, a hospital can collect patients' categorical data as they visit the hospital for the first time.

While sequential collection of individual data may make the estimation task under the LDP constraint harder, it may also offer an opportunity to adapt the randomized mechanism in time to improve the estimation quality. Motivated by that, in this chapter, we address the problem of online Bayesian estimation of a categorical distribution (θ) under ϵ-LDP, while at the same time choosing the randomization mechanism adaptively so that the utility is improved continually in time.

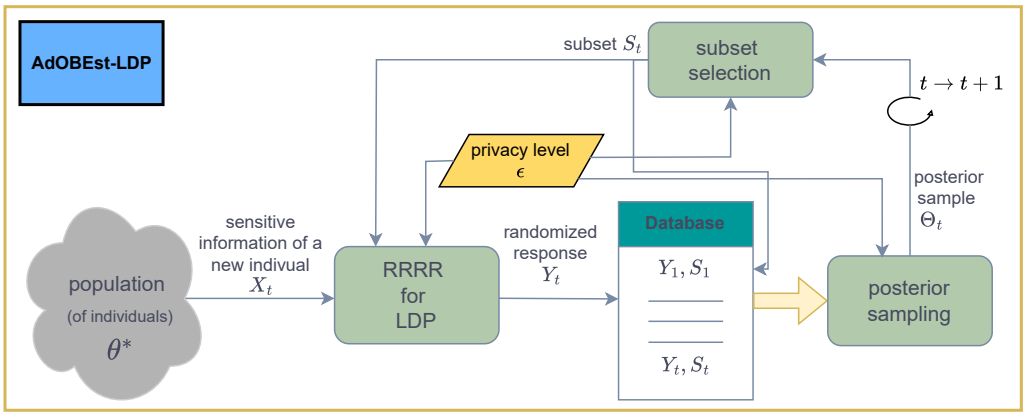


Figure 5.1 AdOBEst-LDP: A framework for Adaptive and Online Bayesian Estimation of categorical distributions with Local Differential Privacy.

Contribution: This chapter presents AdOBEst-LDP: a new methodological framework for Adaptive Online Bayesian Frequency Estimation with LDP. A flowchart diagram of AdOBEst-LDP is given in Figure 5.1 to expose the reader to the main idea of the framework. The main idea of AdOBEst-LDP is to collect future privatized categorical data with high estimation utility based on the knowledge extracted from the previously collected privatized categorical data. To achieve this goal, AdOBEst-LDP continually adapts its randomized response mechanism to the estimation of θ.

The development of AdOBEst-LDP offers three main contributions to the LDP literature.

- **A new randomized response mechanism:** AdOBEst-LDP uses a new adaptive *Randomly Restricted Randomized Response* mechanism (RRRR) to

produce randomized responses under ϵ-LDP. RRRR is a generalization of the standard randomized response mechanism in that it restricts the response to a *subset* of categories. This subset is selected such that the sensitive information X of the next individual is likely contained in that subset. To ensure this, the subset selection step uses two inputs: (i) a sample for θ drawn from the posterior distribution of θ conditional on the past data, (ii) a utility function that scores the informativeness of the randomized response obtained from RRRR when it is run with a given subset. To that end, we propose several utility functions to score the informativeness of the randomized response. The utility functions are based on well-known tools and metrics from probability and statistics, such as Fisher information (Alparslan & Yıldırım, 2022; Lopuhaä-Zwakenberg, Škorić & Li, 2022; Steinberger, 2024; Yıldırım, 2024), entropy, total variation distance, expected squared error, and probability of honest response, i.e., $Y = X$. We provide some insight into those utility functions both theoretically and numerically. Moreover, we also provide a computational complexity analysis for the proposed utility functions.

- **Posterior sampling:** We equip AdOBEst-LDP with a scalable posterior sampling method for parameter estimation. Bayesian estimation is a natural choice for inference when the data is corrupted or censored (Kim, Jung & Chung, 2011; Liu, Zhang, Jin & Pan, 2022; Lone, Panahi, Anwar & Shahab, 2024) and such modification can be statistically modeled. In differential privacy settings, too, Bayesian inference is widely employed (Alparslan & Yıldırım, 2022; Foulds, Geumlek & an Kamalika Chaudhuri, 2016; Karwa, Slavković & Krivitsky, 2014; Williams & Mcsherry, 2010) when the input data is shared with privacy-preserving noise. Standard MCMC methods, such as Gibbs sampling, have a computation complexity quadratic in the number of individuals whose data have been collected. As a remedy to this, similar to Mazumdar, Pacchiano, Ma, Bartlett & Jordan (2020), we propose a *stochastic gradient Langevin dynamics* (SGLD)-based algorithm to obtain approximate posterior samples (Welling & Teh, 2011a). By working on subsets of data, SGLD scales in time.

  The numerical experiments show that AdOBEst-LDP outperforms its non-adaptive counterpart when run with SGLD for posterior sampling. The results also suggest that the utility functions considered in this chapter are robust and perform well. The MATLAB code at `https://github.com/soneraydin/AdOBEst_LDP` can be used to reproduce the results obtained in this chapter.

- **Convergence results:** Finally, we provide a theoretical analysis of AdOBEst-LDP. We prove two main results:

(i) The targeted posterior distribution conditional on the generated observations by the adaptive scheme converges to the true parameter in probability in the number of observations, n. This convergence result owes mainly to the smoothness and a special form of concavity of the marginal log-likelihood function of the randomized responses. Another key factor is that the second moment of the sum up to time n of the gradient of this log-marginal likelihood is $\mathcal{O}(n)$.

(ii) If posterior sampling is performed exactly, the expected frequency of the algorithm choosing the best subset (according to the utility function) tends to 1 as n goes to ∞.

The theoretical results require fairly weak, realistic, and verifiable assumptions.

Outline: In Section 5.2, we discuss the earlier work related to ours. Section 5.3 presents LDP and the frequency estimation problem and introduces AdOBEst-LDP as a general framework. In Section 5.4, we delve deeper into the details of AdOBEst-LDP by first presenting RRRR, the proposed randomized response mechanism, then explaining how it chooses an 'optimal' subset of categories adaptively at each iteration. Section 5.4 also presents the utility metrics considered for choosing these subsets in this chapter. In Section 5.5, we provide the details of the posterior sampling methods considered in this chapter, particularly SGLD. The theoretical analysis of AdOBEst-LDP is provided in Section 5.6. Section 5.7 contains the numerical experiments. Finally, Section 5.8 provides some concluding remarks. All the proofs of the theoretical results are given in the appendices.

5.2 Related Literature

Frequency estimation under the LDP setting has been an increasingly popular research area in recent years. Along with its basic application (estimation of discrete probabilities from locally privatized data), it is also used for a wide range of other estimation and learning purposes such as estimation of confidence intervals and confidence sets for a population mean (Waudby-Smith, Wu & Ramdas, 2023), estimation or identification of heavy hitters (Zhu, Cao, Xue, Wu & Zhang, 2024) (Wang, Li, Zhong, Chen, Wang, Zhou, Peng, Qian, Du & Yang, 2024) (Jia & Gong,

2019), estimation of quantiles (Cormode & Bharadwaj, 2022), frequent itemset mining (Zhao, Zhao, Chen, Liu, Li & Zhang, 2023), estimation of degree distribution in social networks (Wang, Jiang, Peng & Li, 2024), distributed training of graph neural networks with categorical features and labels (Bhaila, Huang, Wu & Wu, 2024). The methods that are proposed for ϵ-LDP frequency estimation also form the basis of more complex inferential tasks (with some modifications on these methods), such as the release of 'marginals' (contingency tables) between multiple categorical features and their correlations, as in the work of Cormode, Kulkarni & Srivastava (2018).

AdOBEst-LDP employs RRRR as its randomized mechanism to produce randomized responses. RRRR is a modified version of the *Standard Randomized Response* mechanism (SRR) (also known as *generalized randomized response*, *k-randomized response*, and *direct encoding* in the literature.) Given X as its input, SRR outputs X with probability $\frac{e^\epsilon}{e^\epsilon+K-1}$, otherwise outputs one of the other categories at random. This is a well-studied mechanism in the DP literature, and the statistical properties of its basic version (such as its estimation variance) can be found in the works by (Wang et al., 2017) and (Wang et al., 2020). When K is large, the utility of SRR can be too low. RRRR in AdOBEst-LDP is designed to circumvent this problem by constraining its output to a subset of categories. Unlike SRR, the perturbation probability of responses in our algorithm changes adaptively, depending on the cardinality of the selected subset of categories (which we explain in detail in Section 5.4) for the privatization of X, and the cardinality of its complementary set.

The use of information metrics as utility functions in LDP protocols has been an active line of research in recent years. In the work of Kairouz, Oh & Viswanath (2016), information metrics like f-divergence and mutual information are used for selecting optimal LDP protocols. In the same vein, Steinberger (2024) uses Fisher Information as the utility metric for finding a nearly optimal LDP protocol for the frequency estimation problem, and Lopuhaä-Zwakenberg et al. (2022) uses it for comparing the utility of various LDP protocols for frequency estimation and finding the optimal one. In these works, the mentioned information metrics are used statically, i.e., to choose a protocol once and for all, for a given estimation task. The approaches in these works suffer from computational complexity for large values of K because the search space for optimal protocols there grows in the order of 2^K. In some other works, such as Wang, Huang, Wang, Nie, Xu, Yang, Li & Qiao (2016), a randomly sampled subset of size $k \leq K$ is used to improve the efficiency of this task, where the optimal k is determined by maximizing the *mutual information* between real data and the privatized data. However, this approach is also static as the optimal subset size k is selected only once, and the optimization procedure only determines k and not the subset itself. Unlike those static approaches, AdOBEst-LDP

dynamically uses the information metric (such as the Fisher Information matrix and the other alternatives in Section 5.4.3) to select the optimal subset at each time step. In addition, in the subset selection step of AdOBEst-LDP, only K candidate subsets are compared in terms of their utilities at each iteration, enabling computational tractability. This way of tackling the problem requires computing the given information metric for only K times at each iteration. We will provide further details of this approach in Section 5.4.3 and provide a computational complexity analysis in Section 5.4.4.

Another use of the Fisher Information in the LDP literature is for bounding the estimation error for a given LDP protocol. For example, Barnes, Chen & Özgür (2020) uses Fisher Information inside van Trees inequality, the Bayesian version of the Cramér-Rao bound (Gill & Levit, 1995), for bounding the estimation error of various LDP protocols for Gaussian mean estimation and frequency estimation. Again, their work provides rules for choosing optimal protocols for a given ϵ in a static way. As a similar example, Acharya, Canonne, Sun & Tyagi (2023) derives a general *information contraction bound* for parameter estimation problems under LDP and shows its relation to van Trees inequality as its special case. To our knowledge, our approach is the first one that adaptively uses a utility metric to dynamically update the inner workings of an LDP protocol for estimating categorical distributions.

The idea of building adaptive mechanisms for improved estimation under the LDP has been studied in the literature, although the focus and methodology of those works differ from ours. For example, Joseph, Kulkarni, Mao & Wu (2019) proposed a two-step adaptive method to estimate the unknown mean parameter of data from Gaussian distribution. In this method, the users are split into two groups, an initial mean estimate is obtained from the perturbed data of the first group, and the data from the second group is transformed adaptively according to that initial estimate. Similarly, Wei, Bao, Xiao, Yang & Ding (2024) proposed another two-step adaptive method for the mean estimation problem, in which the aggregator first computes a rough distribution estimate from the noisy data of a small sample of users, which is then used for adjusting the amount of perturbation for the data of remaining users. While Joseph et al. (2019); Wei et al. (2024) consider a two-stage method, AdOBEst-LDP seeks to adapt continually by updating its LDP mechanism each time an individual's information is collected. Similar to our work, Yıldırım (2024) has recently proposed an adaptive LDP mechanism for online parameter estimation for continuous distributions. The LDP mechanism of Yıldırım (2024) contains a truncation step with boundaries adapted to the estimate from the past data according to a utility function based on the Fisher information. Unfortunately, the

parameter estimation step of Yıldırım (2024) does not scale in time. Differently from Yıldırım (2024), AdOBEst-LDP focuses on categorical distributions, considers several other utility functions to update its LDP mechanism, employs a scalable parameter estimation step, and its performance is backed up with theoretical results.

5.3 Problem definition and general framework

Suppose we are interested in a discrete probability distribution $\mathcal{P}$ of a certain form of sensitive categorical information $X \in [K] := \{1, \dots, K\}$ of individuals in a population. Hence, $\mathcal{P}$ is a categorical distribution $\text{Cat}(\theta^*)$ with a probability vector

$$\theta^* := (\theta_1^*, \dots, \theta_K^*) \in \Delta,$$

where Δ is the $(K-1)$-dimensional probability simplex,

$$\Delta := \left\{ \theta \in \mathbb{R}^K : \sum_{k=1}^{K} \theta_k = 1 \text{ and } \theta_k \geq 0 \text{ for } k \in [K] \right\}.$$

We assume a setting where individuals' sensitive data are collected *privately* and *sequentially in time*. The privatization is performed via a randomized algorithm that, upon taking a category index in $[K]$ as an input, returns a random category index in $[K]$ such that the whole data collection process is ϵ-LDP. (See Definition 4.) Let X_t and Y_t be the private information and randomized responses of individual t, respectively. According to Definition 4 for LDP, the following inequality must be satisfied for all triples $(x, x', y) \in [K]^3$ for the randomized mechanism to be ϵ-LDP.

$$\mathbb{P}(Y_t = y | X_t = x) \leq e^{\epsilon} \mathbb{P}(Y_t = y | X_t = x'). \tag{5.1}$$

The inferential goal is to estimate θ^* sequentially based on the responses $Y_1, Y_2, \dots$, and the mechanisms $\mathcal{M}_1, \mathcal{M}_2, \dots$ that are used to generate those responses. Specifically, Bayesian estimation is considered, whereby the target is the posterior distribution, denoted by $\Pi(\mathrm{d}\theta | Y_{1:n}, \mathcal{M}_{1:n})$, given a prior probability distribution with pdf $\eta(\theta)$ on Δ.

This chapter concerns the Bayesian estimation of θ while adapting the randomized mechanism to improve the estimation utility continually. We propose a general framework called AdOBEst-LDP, in which the randomized mechanism at time t

Algorithm 7 AdOBEst-LDP: Adaptive Online Bayesian Estimation with LDP

Initialization: Start with an initial estimator $\Theta_0 = \theta_{\text{init}}$.

for $t = 1, 2, \ldots$ **do**

Step 1: Adapting the LDP mechanism: Based on Θ_{t-1}, determine the ϵ-LDP mechanism $\mathcal{M}_t$ for the next individual according to a utility metric.

Step 2: LDP response generation The sensitive information X_t of individual t is shared as Y_t using the ϵ-LDP mechanism $\mathcal{M}_t$.

Step 3: Draw a sample (approximately) from the posterior distribution

$$\Theta_t \sim \Pi(\cdot | Y_{1:t}, \mathcal{M}_{1:t}).$$

is adapted to the data collected until time $t-1$. AdOBEst-LDP is outlined in Algorithm 7.

Algorithm 7 is fairly general, and it does not describe how to choose the ϵ-LDP mechanism $\mathcal{M}_t$ at time t, nor does it provide the details of the posterior sampling. However, it is still worth making some critical observations about the nature of the algorithm. Firstly, at time t the selection of the ϵ-LDP mechanism in Step 1 relies on the posterior sample Θ_{t-1}, which serves as an *estimator* of the true parameter θ^* based on the past observations. As we shall see in Section 5.4, at Step 1 the 'best' ϵ-LDP mechanism is chosen from a set of candidate LDP mechanisms according to a utility function. This step is relevant only when Θ_{t-1} is a reliable estimator of θ^*. In other words, Step 1 'exploits' the estimator Θ_{t-1}. Moreover, the random nature of posterior sampling prevents having too much confidence in the current estimator Θ_{t-1} and enables a certain degree of 'exploration.' In conclusion, Algorithm 7 utilizes an 'exploration-exploitation' approach reminiscent of reinforcement learning. In particular, posterior sampling in Step 3 suggests a strong parallelism between AdOBEst-LDP and the well-known exploration-exploitation approach called Thompson sampling (Russo, Roy, Kazerouni, Osband & Wen, 2018).

The details of Steps 1-2 and Step 3 of Algorithm 7 are given in Sections 5.4 and 5.5, respectively.

5.4 Constructing informative randomized response mechanisms

In this section, we describe Steps 1-2 of AdOBEst-LDP in Algorithm 7 where the ϵ-LDP mechanism $\mathcal{M}_t$ is selected at time t based on the posterior sample Θ_t and

a randomized response is generated using $\mathcal{M}_t$. For ease of exposition, we will drop the time index t throughout the section and let $\Theta_{t-1} = \theta$.

Recall from Definition 4 that an ϵ-LDP randomized mechanism is associated with a conditional probability distribution that satisfies (5.1). An ϵ-LDP mechanism is not unique. One such mechanism is the standard randomized response mechanism (SRR). For subsequent use, it is convenient to define SRR generally: We let $\texttt{SRR}(X;\Omega,\epsilon)$ the output of SRR which operates on the set Ω with LDP parameter ϵ when the input is $X \in \Omega$. Then, we have

$$Y = \mathrm{SRR}(X;\Omega,\epsilon) = \begin{cases} X & \text{w.p. } e^\epsilon/(e^\epsilon + |\Omega| - 1) \\ \sim \mathrm{Uniform}(\Omega/\{X\}) & \text{else} \end{cases}. \tag{5.2}$$

We aim to develop an alternative randomized mechanism whose response Y is more informative about θ^* than the one generated as $Y = \mathrm{SRR}(X;[K],\epsilon)$. The main idea is as follows. Supposing that the posterior sample $\Theta_{t-1} = \theta$ is an accurate estimate of θ^*, it is reasonable to aim for the 'best' ϵ-LDP mechanism (among a range of candidates) which would maximize the (estimation) utility of Y if the true parameter were $\theta^* = \theta$. We follow this main idea to develop the proposed ϵ-LDP mechanism.

5.4.1 The randomly restricted randomized response (RRRR) mechanism

Given $\Theta_{t-1} = \theta \in \Delta$, an informative randomized response mechanism can be constructed by considering a *high-probability set* $S \subset [K]$ and a *low-probability set* $S^c = [K]/S$ for X (according to θ). Then, a sensible alternative to $\mathrm{SRR}(X;[K],\epsilon)$ would be to confine the randomized response to the set S (unioned by a random element from S^c to remain LDP). The expected benefit of this approach is due to (i) using less amount of randomization since $|S| < K$, and thus (ii) having an informative response when $X \in S$, which happens with a high probability. Based on this approach, we propose RRRR, whose precise steps are given in Algorithm 8.

RRRR has three algorithmic parameters: a subset S of $[K]$ and two privacy parameters ϵ_1 and ϵ_2 which operates on S and S^c, respectively. Theorem 6 states the necessary conditions for ϵ_1 and ϵ_2 for RRRR to be ϵ-LDP. A proof of Theorem 6 is given in Appendix C.1.1.

Algorithm 8 RRRR

Input: Sample space size K, a subset $S \subset [K]$, privacy parameters $\epsilon_1, \epsilon_2 > 0$, input $X \in [K]$

Output: Randomized response $Y \in [K]$

if $X \in S$ **then**

  Draw $R \sim \text{Uniform}(S^c)$.

  Set $Y = \texttt{SRR}(X; S \cup \{R\}, \epsilon_1)$ as in (5.2).

else

  Set $R = \texttt{SRR}(X; S^c, \epsilon_2)$ as in (5.2).

  Set $Y = \texttt{SRR}(R; S \cup \{R\}, \epsilon_1)$ as in (5.2).

return Y

Theorem 6. RRRR *is ϵ-DP if $\epsilon_1 \leq \epsilon$ and*

$$\epsilon_2 = \begin{cases} \min\left\{\epsilon, \ln \frac{|S^c|-1}{e^{\epsilon_1-\epsilon}|S^c|-1}\right\} & \text{for } \epsilon - \epsilon_1 < \ln|S^c| \text{ and } |S| > 0 \\ \epsilon & \text{else} \end{cases}. \tag{5.3}$$

Note that when $S = \emptyset$ and $\epsilon_2 = \epsilon$, RRRR reduces to SRR.

5.4.2 Choosing the privacy parameters ϵ_1, ϵ_2

We elaborate on the choice of ϵ_1 and ϵ_2 in the light of Theorem 6. In RRRR, the probability of an honest response, i.e., $X = Y$, given $X \in S$, is

$$\mathbb{P}(Y = X | X \in S) = \frac{e^{\epsilon_1}}{e^{\epsilon_1} + |S|},$$

which should be contrasted to $e^\epsilon/(e^\epsilon + K - 1)$, which would be the probability if $Y = \text{SRR}(X; [K], \epsilon)$. Anticipating that $\{X \in S\}$ is likely, one should at least aim for ϵ_1 that satisfies $\mathbb{P}(X = Y | X \in S) \geq e^\epsilon/(e^\epsilon + K - 1)$ for RRRR to be relevant. This is equivalent to

$$\epsilon_1 \geq \epsilon + \ln|S| - \ln(K-1). \tag{5.4}$$

Taking into account also the constraint that $\epsilon_1 \leq \epsilon$ (by Theorem 6), we suggest $\epsilon_1 = \kappa\epsilon$, where $\kappa \in (0,1)$ is a number close to 1, such as 0.9, to ensure (5.4) with a significant margin. (It is possible to choose $\kappa = 1$; however, again by Theorem 6, this requires that $\epsilon_2 = 0$, which renders Y completely uninformative when $X \notin S$.) In Section 5.7, we discuss the choice of κ in more detail.

For the next section, we assume a fixed $\kappa \in (0,1)$, and set $\epsilon_1 = \kappa\epsilon$; and we focus on the selection of S.

5.4.3 Subset selection for RRRR

Let $\texttt{RRRR}(X; S, \epsilon)$ be the random output of RRRR that achieves ϵ-LDP by using the subset S and the privacy parameters $\epsilon_1 = \kappa\epsilon$ and ϵ_2 as in (5.3) when the input is X. Furthermore, let $U(\theta, S, \epsilon)$ be the (inferential) 'utility' of $Y = \texttt{RRRR}(X; S, \epsilon)$ when $X \sim \text{Cat}(\theta)$. One would like to choose S that maximizes $U(\theta, S, \epsilon)$. (One could also seek to optimize κ in $\epsilon_1 = \kappa\epsilon$, too, however with the expense of additional computation.)

However, since there are $2^K - 1$ feasible choices for S, one must confine the search space for S in practice. As discussed above, RRRR becomes most relevant when the set S is a high-probability set. Therefore, for a given θ, we confine the choices for S to

$$S_{k,\theta} := \{\sigma_\theta(1), \sigma_\theta(2), \ldots, \sigma_\theta(k)\}, \quad k = 1, \ldots, K. \tag{5.5}$$

where $\sigma_\theta := (\sigma_\theta(1), \ldots, \sigma_\theta(K))$ be the permutation vector for θ so that $\theta_{\sigma_\theta(1)} \geq \ldots \geq \theta_{\sigma_\theta(K)}$.

Then the subset selection problem can be formulated as finding

$$k^* = \arg \max_{k \in \{0, \ldots, K-1\}} U(\theta, S_{k,\theta}, \epsilon). \tag{5.6}$$

The alternatives in (5.5) can be justified. Since $S_{k,\theta}$ contains the indices of the k highest-valued components of θ^*, it is expected to cover a large portion of the total probability for X. This can be the case even for a small value of k relative to K when the components of θ^* are not evenly distributed. Also, the alternatives cover the basic SRR, which is obtained with $k = 0$ (leading to $S = \emptyset$ and $\epsilon_2 = \epsilon$).

In the subsequent sections, we present six different utility functions $U(\theta, S, \epsilon)$ and justify their relevance to estimation; the usefulness of the proposed functions is also demonstrated in the numerical experiments.

5.4.3.1 Fisher information matrix

The first utility function under consideration is based on the Fisher information matrix at θ according to the distribution of Y given θ. It is well-known that the inverse of the Fisher information matrix sets the Cramer-Rao lower bound for the variance of an unbiased estimator. Hence, the Fisher information can be regarded as a reasonable metric to quantify the information contained in Y about θ. This approach is adopted in Lopuhaä-Zwakenberg et al. (2022); Steinberger (2024) for LDP applications for estimating discrete distributions, and Alparslan & Yıldırım (2022); Yıldırım (2024) in similar problems involving parametric continuous distributions.

For a given $\theta \in \Delta$, let $F(\theta; S, \epsilon)$ be the Fisher information matrix evaluated at θ when $X \sim \mathrm{Cat}(\theta)$ and $Y = \texttt{RRRR}(X; S, \epsilon)$. Let

$$g_{S,\epsilon}(y|x) := \mathbb{P}(Y = y | X = x)$$

when $Y = \texttt{RRRR}(X; S, \epsilon)$. The following result states $F(\theta; S, \epsilon)$ in terms of $g_{S,\epsilon}$ and θ. The result is derived in Lopuhaä-Zwakenberg et al. (2022); we also give a simple proof in Appendix C.1.2. Note that $F(\theta; S, \epsilon)$ is $(K-1) \times (K-1)$ since θ has $K-1$ free components and $\theta_K = 1 - \sum_{i=1}^{K-1} \theta_i$.

Proposition 2. *The Fisher information matrix for RRRR is given by*

$$F(\theta; S, \epsilon) = A_{S,\epsilon}^\top D_\theta^{-1} A_{S,\epsilon}, \tag{5.7}$$

where $A_{S,\epsilon}$ is a $K \times (K-1)$ matrix whose entries are $A_{S,\epsilon}(i,j) := g_{S,\epsilon}(i|j) - g_{S,\epsilon}(i|K)$ and D_θ is a $K \times K$ diagonal matrix with elements $D_\theta(i,i) := \sum_{j=1}^K g_{S,\epsilon}(i|j)\theta_j$.

We define the following utility function based on the Fisher information

$$U_1(\theta, S, \epsilon) := -\mathrm{Tr}\left[F^{-1}(\theta; S, \epsilon)\right]. \tag{5.8}$$

This utility function depends on the Fisher information differently from Lopuhaä-Zwakenberg et al. (2022); Steinberger (2024), who considered the determinant of the FIM as the utility function. The rationale behind (5.8) is that the for an unbiased estimator $\hat{\theta}(Y)$ of θ^* based on $Y = \texttt{RRRR}(X; S, \epsilon)$, the expected mean squared error is bounded by $E_{\theta^*}[\|\hat{\theta}(Y) - \theta^*\|^2] \leq \mathrm{Tr}\left[F^{-1}(\theta^*; S, \epsilon)\right]$. For the utility function in (5.8) to be well-defined, the FIM needs to be invertible. Proposition 3, proven in Appendix C.1.2, states that this is indeed the case.

Proposition 3. *$F(\theta; S, \epsilon)$ in (5.7) is invertible for all $\theta \in \Delta$, $S \subset [K]$, and $\epsilon_1, \epsilon_2 > 0$.*

5.4.3.2 Entropy of randomized response

For discrete distributions, entropy measures *uniformity*. Hence, in the LDP framework, a lower entropy for the randomized response Y implies a more informative Y. Based on that observation, a utility function can be defined as the negative entropy of the marginal distribution of Y,

$$U_2(\theta, S, \epsilon) := \sum_{y=1}^{K} \ln h_{S,\epsilon}(y|\theta) h_{S,\epsilon}(y|\theta),$$

where $h_{S,\epsilon}(y|\theta)$ is the marginal probability of $Y = y$ given θ,

$$h_{S,\epsilon}(y|\theta) := \sum_{x=1}^{K} g_{S,\epsilon}(y|x)\theta_x.$$

5.4.3.3 Total variation distance

The TV distance between two discrete probability distributions μ, ν on $[K]$ is given by

$$\mathrm{TV}(\mu, \nu) := \frac{1}{2} \sum_{k=1}^{K} |\mu(x) - \nu(x)|.$$

We consider two utility functions based on TV distance. The first function arises from the observation that a more informative response Y generally leads to a larger change in the posterior distribution of X given Y, θ,

$$p_{S,\epsilon}(x|y,\theta) := \frac{\theta_x \cdot g_{S,\epsilon}(y|x)}{h_{S,\epsilon}(y|\theta)}, \quad x = 1, \ldots, K, \tag{5.9}$$

relative to its prior $\mathrm{Cat}(\theta)$. The expected amount of change can be formulated as the expectation of the TV distance between the prior and posterior distributions with respect to the marginal distribution of Y given θ. Then, a utility function can be defined as

$$\begin{aligned} U_3(\theta, S, \epsilon) &:= \mathbb{E}_\theta \left[\mathrm{TV}(p_{S,\epsilon}(\cdot|Y,\theta), \mathrm{Cat}(\theta)) \right] \\ &= \frac{1}{2} \sum_{x=1}^{K} \sum_{y=1}^{K} \left| g_{S,\epsilon}(y|x)\theta_x - h_{S,\epsilon}(y|\theta)\theta_x \right|. \end{aligned}$$

Another utility function is related to the TV distance between the marginal probability distributions of X given θ and Y given θ. Since X is more informative about θ than the randomized response Y, the mentioned TV distance is desired to be as

small as possible. Hence, a utility function may be formulated as

$$\begin{aligned} U_4(\theta, S, \epsilon) &:= -\mathrm{TV}(h_{S,\epsilon}(\cdot|\theta), \mathrm{Cat}(\theta)) \\ &= -\frac{1}{2}\sum_{i=1}^{K} |h_{S,\epsilon}(i|\theta) - \theta_i|. \end{aligned}$$

5.4.3.4 Expected mean squared error

One can also wish to choose S such that the Bayesian estimator of X given Y has the lowest expected squared error. Specifically, given $k \in [K]$ let e_k be a $K \times 1$ vector of 0s except that $e_k = 1$. A utility function can be defined based on that as

$$U_5(\theta, S, \epsilon) := -\arg\min_{\widehat{e_X}} \mathbb{E}_\theta \left[\|e_X - \widehat{e_X}(Y)\|^2 \right], \tag{5.10}$$

where $\mathbb{E}_\theta \left[\|e_X - \widehat{e_X}(Y)\|^2 \right]$ is the mean squared error for the estimator $\widehat{e_X}$ of e_X given Y when $X \sim \mathrm{Cat}(\theta)$ and $Y = \texttt{RRRR}(X; S, \epsilon)$, which is known to be minimized when $\widehat{e_X}$ is the Bayesian estimator of e_X. Proposition 4 provides an explicit formula for this utility function. A proof is given in Appendix C.1.2.

Proposition 4. *For the utility function in* (5.10)*, we have*

$$U_5(\theta, S, \epsilon) = \sum_{y=1}^{K} \sum_{x=1}^{K} \frac{g_{S,\epsilon}(y|x)^2 \theta_x^2}{h_{S,\epsilon}(y|\theta)} - 1.$$

5.4.3.5 Probability of honest response

Our last alternative for the utility function is a simple yet intuitive one, which is the probability of an honest response, i.e.,

$$U_6(\theta, S, \epsilon) := \mathbb{P}_\theta(Y = X|S). \tag{5.11}$$

This probability is explicitly given by

$$\begin{aligned}\mathbb{P}_\theta(Y = X|S) &= \mathbb{P}(Y = X|X \in S)\mathbb{P}_\theta(X \in S) + \mathbb{P}(Y = X|X \notin S)\mathbb{P}_\theta(X \notin S) \\ &= \frac{e^{\epsilon_1}}{e^{\epsilon_1} + |S|}\left(\sum_{i \in S}\theta_i + \frac{e^{\epsilon_2}}{e^{\epsilon_2} + K - |S| - 1}\sum_{i \notin S}\theta_i\right).\end{aligned}$$

Recall that, for computational tractability, we confined the possible sets for S to the subsets $\{\sigma_\theta(1), \ldots, \sigma_\theta(k)\}$, $k = 0, \ldots, K-1$ and select S by solving the maximization problem in (5.6). Remarkably, if $U_6(\theta, S, \epsilon)$ is used for the utility function, the restricted maximization (5.6) is equivalent to *global* maximization, i.e., finding the best S among all the 2^K possible subsets S. We state this as a theorem and prove it in Appendix C.1.2.

Theorem 7. *For the utility function $U_6(\theta, S, \epsilon)$ in* (5.11) *and $S_{k,\theta}$s in* (5.5)*, we have*

$$\max_{k=0,\ldots,K-1} U_6(\theta, S_{k,\theta}, \epsilon) = \max_{S \subset [K]} U_6(\theta, S, \epsilon).$$

5.4.3.6 Semi-adaptive approach

We also consider a semi-adaptive approach which uses a fixed parameter $\alpha \in (0, 1)$ to select the smallest $S_{k,\theta}$ in (5.5) such that $\mathbb{P}_\theta(X \in S_{k,\theta}) \geq \alpha$, that is, $S = \{\sigma_\theta(1), \ldots, \sigma_\theta(k^*)\}$ is taken such that

$$\mathbb{P}_\theta(X \in \{\sigma_\theta(1), \ldots, \sigma_\theta(k^* - 1)\}) < \alpha \text{ and } \mathbb{P}_\theta(X \in \{\sigma_\theta(1), \ldots, \sigma_\theta(k^*)\}) \geq \alpha.$$

Again, the idea is to randomize the most likely values of X with a high accuracy. The approach forms the subset S by including values for X in descending order of their probabilities (given by θ) until the cumulative probability exceeds α. In that way, it is expected to have set S that is small-sized (especially when θ is unbalanced) and captures the most likely values of X. The resulting S has varying cardinality depending on the sampled θ at the current time step.

We call this approach "semi-adaptive" because, while it still adapts to θ, it uses the fixed parameter α. As we will see in Section 5.7, the best α depends on various parameters such as ϵ, K, and the degree of evenness in θ.

	Fisher	Entropy	TV_1	TV_2	MSE	$\mathbb{P}_\theta(Y=X)$	Semi-adaptive
Computing utility	$\mathcal{O}(K^3)$	$\mathcal{O}(K^2)$	$\mathcal{O}(K^2)$	$\mathcal{O}(K^2)$	$\mathcal{O}(K^2)$	$\mathcal{O}(K)$	NA
Choosing S	$\mathcal{O}(K^4)$	$\mathcal{O}(K^3)$	$\mathcal{O}(K^3)$	$\mathcal{O}(K^3)$	$\mathcal{O}(K^3)$	$\mathcal{O}(K)$	$\mathcal{O}(K)$

Table 5.1 Computational complexity of utility functions and choosing S

5.4.4 Computational complexity of utility functions

We now provide the computational complexity analysis of the utility metrics presented in Section 5.4.3.1-5.4.3.5, and that of the semi-adaptive approach in Section 5.4.3.6, as a function of K. The first row of Table 5.1 shows the computational complexities of calculating the utility function for a fixed S, and the second row shows the complexities of choosing the best S according to (5.6). To find (5.6), the utility function generally needs to be calculated K times, which explains the additional K factor in the computational complexities in the second row.

The least demanding utility function is U_6, that is based on $\mathbb{P}_\theta(Y=X)$, whose complexity is $\mathcal{O}(K)$. Moreover, finding the best S can also be done in $\mathcal{O}(K)$ time because one can compute this utility metric for all $k=0,\dots,K-1$ by starting with $S=\emptyset$ and expanding it incrementally. Also note that the semi-adaptive approach does not use a utility metric and finding k^* can be done in $\mathcal{O}(K)$ time by summing the components of θ from largest to smallest until the cumulative sum exceeds the given α parameter. So, its complexity is $\mathcal{O}(K)$.

For all these approaches, it is additionally required to sort θ beforehand, which is an $\mathcal{O}(K\ln K)$ operation with an efficient sorting algorithm like *merge sort*.

In practice, one can choose among these utility functions depending on the nature of the application. When the number of categories K or the arrival rate of sensitive data is large, we suggest using U_6 or a semi-adaptive approach. When K and the arrival rate of the personal data are both small, the more computationally demanding utility functions can also be used.

Example 1 (**Numerical illustration**). *We close this section with an example that shows the benefit of* `RRRR` *and the role of S. We consider θ values such that θ_i/θ_{i+1} is constant for $i=1,\dots,K-1$. The ratio θ_i/θ_{i+1} controls the degree of 'evenness' in θ: The smaller ratio indicates a more evenly distributed θ. Note that θ is already ordered in this example; hence, we consider using $S=\{1,\dots,k\}$ which has the k most likely values for X according to θ. Also, for a given ϵ, we fix $\epsilon_1=0.9\epsilon$ and set ϵ_2 according to* (5.3).

Figure 5.2 shows, for a fixed ϵ and $K=20$, and various values of k, the probability

of the randomized response being equal to the sensitive information, i.e., $\mathbb{P}_\theta(Y = X)$ *vs* θ_i/θ_{i+1} *when* $S = \{1, \ldots, k\}$ *in RRRR. (Recall that this probability corresponds to* $U_6(\theta, S, \epsilon)$*.) Comparing this probability with* $e^\epsilon/(e^\epsilon + K - 1)$*, the probability obtained with* $Y = SRR(X; [K], \epsilon)$*, it can be observed that RRRR can do significantly better than SRR if* k *can be chosen suitably. The plots demonstrate that the "suitable"* k *depends on* θ*: While the best* k *tends to be larger for more even* θ*, small* k *becomes the better choice for non-even* θ *(large* θ_i/θ_{i+1}*). This is because, when* θ_i/θ_{i+1} *is large, the probability is concentrated on just a few components, and* S *with a small* k *captures most of the probability. Moreover, the plots for* $\epsilon = 1$ *and* $\epsilon = 5$ *also show the effect of the level of privacy. In more challenging scenarios where* ϵ *is smaller, the gain obtained by RRRR compared to SRR is bigger.*

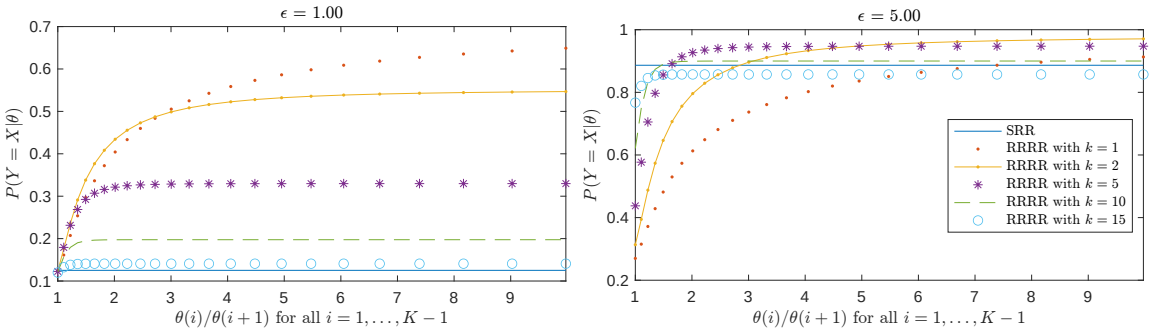


Figure 5.2 $\mathbb{P}_\theta(Y = X)$ vs θ_i/θ_{i+1} for all $i = 1, \ldots, K-1$ with $K = 20$. Left: $\epsilon = 1$, Right: $\epsilon = 5$.

5.5 Posterior sampling

Steps 1-2 of AdOBEst-LDP in Algorithm 7 were detailed in the previous section. In this section, we provide the details of Step 3.

Step 3 of AdOBEst-LDP requires sampling from the posterior distribution $\Pi(\cdot|Y_{1:n}, S_{1:n})$ of θ given $Y_{1:n}$ and $S_{1:n}$ for $n \geq 1$, where S_t is the subset selected at time t to generate Y_t from X_t. Let $\pi(\theta|Y_{1:n}, S_{1:n})$ denote the pdf of $\Pi(\cdot|Y_{1:n}, S_{1:n})$. Given $Y_{1:n} = y_{1:n}$ and $S_{1:n} = s_{1:n}$, the posterior density can be written as

$$\pi(\theta|y_{1:n}, s_{1:n}) \propto \eta(\theta) \prod_{t=1}^{n} h_{s_t,\epsilon}(y_t|\theta). \tag{5.12}$$

Note that the right-hand side does not include a transition probability for S_t's because the sampling procedure of S_t given $Y_{1:t-1}$ and $S_{1:t-1}$ does *not* depend on θ^*.

Furthermore, we assume that the prior distribution $\eta(\theta)$ is a Dirichlet distribution $\theta \sim \mathrm{Dir}(\rho_1, \ldots, \rho_K)$ with prior hyperparameters $\rho_k > 0$, for $k = 1, \ldots, K$.

Unfortunately, the posterior distribution in (5.12) is intractable. Therefore, we resort to approximate sampling approaches using MCMC. Below, we present two MCMC methods, namely SGLD and Gibbs sampling.

5.5.1 Stochastic gradient Langevin dynamics

SGLD is an asymptotically exact gradient-based MCMC sampling approach that enables the use of subsamples of size $m \ll t$. A direct application of SGLD to generate samples for θ from the posterior distribution in (5.12) is difficult. This is because θ lives in the probability simplex Δ, which makes the task of keeping the iterates for θ inside Δ challenging. We overcome this problem by defining the surrogate variables $\phi_1, \ldots, \phi_K$ with

$$\phi_k \overset{\text{ind.}}{\sim} \mathrm{Gamma}(\rho_k, 1), \quad k = 1, \ldots, K,$$

and the mapping from ϕ to θ as

$$\theta(\phi)_k := \frac{\phi_k}{\sum_{j=1}^K \phi_j}, \quad k = 1, \ldots, K. \tag{5.13}$$

It is well-known that the resulting $(\theta_1, \ldots, \theta_K)$ has a Dirichlet distribution $\mathrm{Dir}(\rho_1, \ldots, \rho_K)$, which is exactly the prior distribution $\eta(\theta)$. Therefore, this change of variables preserves the originally constructed probabilistic model. Moreover, since $\phi = (\phi_1, \ldots, \phi_K)$ takes values in $[0, \infty)^K$, we run SGLD for ϕ, where the j'th update is

(5.14)
$$\phi^{(j)} = \left| \phi^{(j-1)} + \frac{\gamma_n}{2} \left(\nabla_\phi \ln p(\phi^{(j-1)}) + \frac{n}{m} \sum_{i=1}^m \nabla_\phi \ln p_{s_{u_i}, \epsilon}(y_{u_i} | \phi^{(j-1)}) \right) + \gamma_n W_j \right|,$$
$$W_j \sim \mathcal{N}(0, I_K).$$

where $u = (u_1, \ldots, u_m)$ is a random subsample of $\{1, \ldots, n\}$. In (5.14), the 'new' prior and likelihood functions are

$$p(\phi) := \prod_{k=1}^{K} \text{Gamma}(\phi_k; \alpha_i, 1), \quad p_{s,\epsilon}(y|\phi) := h_{s,\epsilon}(y|\theta(\phi)). \tag{5.15}$$

The reflection in (5.14) via taking the component-wise absolute value is necessary because each $\phi_k^{(j)}$ must be positive. Step 3 of Algorithm 7 can be approximated by running SGLD for some $M > 0$ iterations. To exploit the SGLD updates from the previous time, one should start the updates at time n by setting the initial value for ϕ to the last SGLD iterate at time $n-1$.

The next proposition provides the explicit formulae for the gradients of the log-prior and the log-likelihood of ϕ in (5.14). A proof is given in Appendix C.2.

Proposition 5. *For $p(\phi)$ and in $p(y|\phi)$ in (5.15), we have*

$$[\nabla_\phi \ln p(\phi)]_i = \frac{\alpha_i - 1}{\phi_i} - 1, \quad [\nabla_\phi \ln p(y|\phi)]_i = \sum_{k=1}^{K-1} J(i,k) \frac{g_{S,\epsilon}(y|k) - g_{S,\epsilon}(y|K)}{h_{S,\epsilon}(y|\theta(\phi))},$$

where J is a $K \times (K-1)$ Jacobian matrix whose (i,j)th element is

$$J(i,j) = \mathbb{I}(i=j) \frac{1}{\sum_{k=1}^{K} \phi_k} - \frac{\phi_j}{\left(\sum_{k=1}^{K} \phi_k\right)^2}.$$

5.5.2 Gibbs sampling

An alternative to SGLD is the Gibbs sampler, which operates on the joint posterior distribution of θ and $X_{1:n}$ given $Y_{1:n} = y_{1:n}$ and $S_{1:n} = s_{1:n}$,

$$p(\theta, x_{1:n}|y_{1:n}, s_{1:n}) \propto \eta(\theta) \left[\prod_{t=1}^{n} \theta_{x_t} g_{s_t,\epsilon}(y_t|x_t) \right].$$

The full conditional distributions of $X_{1:n}$ and θ are tractable. Specifically, for $X_{1:n}$, we have

$$p(x_{1:n}|y_{1:n}, s_{1:n}, \theta) = \prod_{t=1}^{n} p_{s_t,\epsilon}(x_t|y_t, \theta), \tag{5.16}$$

where $p_{s_t,\epsilon}(x_t|y_t, \theta)$ is defined in (5.9). Therefore, (5.16) is a product of n categorical distributions, each with support $[K]$. Furthermore, the full conditional distribution

of θ is a Dirichlet distribution due to the conjugacy between the categorical and the Dirichlet distributions. Specifically,

$$p(\theta|x_{1:n}, y_{1:n}, s_{1:n}) = \mathrm{Dir}(\theta|\rho_1^{\mathrm{post}}, \ldots, \rho_K^{\mathrm{post}}),$$

where the hyperparameters of the posterior distribution are given by $\rho_k^{\mathrm{post}} := \rho_k + \sum_{t=1}^{n} \mathbb{I}(x_t = k)$ for $k = 1, \ldots, K$.

Computational load at time t of sampling from t distributions in (5.16), is proportional to tK, which renders the computational complexity of Gibbs sampling $\mathcal{O}(n^2K)$ after n time steps. This can be computationally prohibitive when n gets large.

5.6 Theoretical analysis

We address two questions concerning AdOBEst-LDP in Algorithm 7 when it is run with RRRR whose subset is selected as described in Section 5.4.3. (i) Does the targeted posterior distribution based on the observations generated by Algorithm 7 converge to the true value θ^*? (ii) How frequently does Algorithm 7 with RRRR select the optimum subset S according to the chosen utility function?

5.6.1 Convergence of the posterior distribution

We begin by developing the joint probability distribution of the random variables involved in AdOBEst-LDP.

- Given $Y_{1:n}$ and $S_{1:n}$, the posterior distribution $\Pi(\cdot|Y_{1:n}, S_{1:n})$ is defined such that for any measurable set $A \subseteq \Delta$, the posterior probability of $\{\theta \in A\}$ is given by

$$\Pi(A|Y_{1:n}, S_{1:n}) := \frac{\int_A \eta(\theta) \prod_{t=1}^{n} h_{S_t,\epsilon}(Y_t|\theta) \mathrm{d}\theta}{\int_\Delta \eta(\theta) \prod_{t=1}^{n} h_{S_t,\epsilon}(Y_t|\theta) \mathrm{d}\theta}. \tag{5.17}$$

- Let $Q(\cdot|Y_{1:n}, S_{1:n}, \Theta_{n-1})$ be the probability distribution corresponding to the posterior sampling process for Θ_n. Note that if exact posterior sampling were

used, we would have $Q(A|Y_{1:n}, S_{1:n}, \Theta_{n-1}) = \Pi(A|Y_{1:n}, S_{1:n})$; however, when approximate sampling techniques are used to target Π, such as SGLD or Gibbs sampling, the equality does not hold in general.

- For $\theta \in \Delta$, let

$$S^*_\theta := \{\sigma_\theta(1), \ldots, \sigma_\theta(k^*_\theta)\}, \quad \text{with} \quad k^*_\theta := \arg \max_{k \in \{0,\ldots,K-1\}} U(\theta, S_{k,\theta}, \epsilon),$$

be the best subset according to θ, where $S_{k,\theta} = \{\sigma_\theta(1), \ldots, \sigma_\theta(k)\}$ is defined in (5.5). Given $\Theta_{1:t-1}$ and $Y_{1:t}$, S_t depends only on Θ_{t-1} and it is given by $S_t = S^*_{\Theta_{t-1}}$.

Combining all, the joint law of $S_{1:n}, Y_{1:n}$ can be expressed as

(5.18)
$$P_{\theta^*}(S_{1:n}, Y_{1:n}) := \prod_{t=1}^{n} h_{S_t,\epsilon}(Y_t|\theta^*) \left[\int_\Delta \mathbb{I}(S_t = S_{k^*,\theta_{t-1}}) Q(\mathrm{d}\theta_{t-1}|Y_{1:t-1}, S_{1:t-1}, \theta_{t-2}) \right],$$

where we use the convention that $Q(\mathrm{d}\theta_0|Y_{1:0}, S_{1:0}, \theta_{-1}) = \delta_{\theta_{\text{init}}}(\mathrm{d}\theta_0)$ for an initial value $\theta_{\text{init}} \in \Delta$.

The posterior probability in (5.17) is a random variable with respect to P_{θ^*} defined in (5.18). Theorem 8 establishes that under the fairly mild Assumption 1 on the prior, the $\Pi(\cdot|Y_{1:n}, S_{1:n})$ converges to θ^* regardless of the choice of Q for posterior sampling.

Assumption 1. *There exist finite positive constants $d > 0$ and $B > 0$ such that $\eta(\theta)/\eta(\theta') < B$ for all $\theta, \theta' \in \Delta$ whenever $\|\theta' - \theta^*\| < d$.*

Theorem 8. *Under Assumption 1, there exists a constant $c > 0$ such that, for any $0 < a < 1$ and the sequence of sets*

$$\Omega_n = \{\theta \in \Delta : \|\theta - \theta^*\|^2 \leq cn^{-a}\},$$

the sequence of probabilities

$$\lim_{n\to\infty} \Pi(\Omega_n|Y_{1:n}, S_{1:n}) \overset{P_{\theta^*}}{\to} 1,$$

regardless of the choice of Q.

A proof is given in Appendix C.3.2, where the constant c in the sets Ω_n is explicitly given.

5.6.2 Selecting the best subset

Let $S^* := S^*_{\theta^*}$ be the best subset at θ^*. In this part, we prove that if posterior sampling is performed exactly, the best subset is chosen with an expected long-run frequency of 1. Our result relies on some mild assumptions.

Assumption 2. *The components of θ^* are strictly ordered, that is, $\theta_{\sigma_{\theta^*}(1)} > \ldots > \theta_{\sigma_{\theta^*}(K)}$.*

Assumption 3. *Given any $S \subset [K]$ and $\epsilon > 0$, $U(\theta, S, \epsilon)$ is a continuous function of θ with respect to the L_2-norm.*

Assumption 4. *The solution of (5.6) is unique at θ^*.*

Assumption 2 is required to avoid technical issues regarding the uniqueness of S^*. Assumptions 3 and 4 impose a certain form of regularity on the utility function.

Theorem 9. *Suppose Assumptions 1-4 hold and Θ_ts are generated by exact sampling, that is, $Q(A|Y_{1:t}, S_{1:t}) = \Pi(A|Y_{1:t}, S_{1:t})$ for all measurable $A \subseteq \Delta$. Then,*

$$\lim_{n\to\infty} P_{\theta^*}(S_n = S^*) \to 1. \tag{5.19}$$

As a corollary, S^* is selected with an expected long-run frequency of 1, that is,

$$\lim_{n\to\infty} \frac{1}{n} \sum_{t=1}^{n} E_{\theta^*}\left[\mathbb{I}(S_t = S^*)\right] = 1. \tag{5.20}$$

The result in (5.20) can be likened to sublinear regret from the reinforcement learning theory.

5.7 Numerical results

We tested[1] the performance of AdOBEst-LDP when the subset S in RRRR is determined according to a utility function in Section 5.4.3. We compared AdOBEst-LDP when combined with each of the utility functions defined in Sections 5.4.3.1-5.4.3.5 with its non-adaptive counterpart when SRR is used to generate Y_t at all steps. We also included the semi-adaptive subset selection method in Section 5.4.3.6 into the comparison. For the semi-adaptive approach, we obtained results for five different

[1]The MATLAB code at `https://github.com/soneraydin/AdOBEst_LDP` can be used to reproduce the results obtained in this chapter.

values of its α parameter, namely $\alpha \in \{0.2, 0.6, 0.8, 0.9, 0.95\}$.

We ran each method for 50 Monte Carlo runs. Each run contained $T = 500K$ time steps. For each run, the sensitive information is generated as $X_t \overset{\text{i.i.d.}}{\sim} \text{Cat}(\theta^*)$ where θ^* itself was randomly drawn from Dirichlet$(\rho, \ldots, \rho)$. Here, the parameter ρ was used to control the unevenness among the components of θ^*. (Smaller ρ leads to more uneven components in general). At each time step, Step 3 of Algorithm 7 was performed by running $M = 20$ updates of an SGLD-based MCMC kernel as described Section 5.5.1. In SGLD, we took the subsample size $m = 50$ and the step-size parameter $a = \frac{0.5}{t}$ at time step t. This type of polynomially decaying step sizes for SGLD are often suggested in the literature. For example, (Welling & Teh, 2011b) show that this type of step sizes ensures both that the algorithm can reach high-probability regions and it converges to the posterior mode. This choice of step size worked well in our practice, and it did not require any tuning. Prior hyperparameters for the gamma distribution were taken $\rho_0 = 1_K$. The posterior sample Θ_t was taken as the last iterate of those SGLD updates. Only for the last time step, $t = T$, the number of MCMC iterations was taken 2000 to reliably calculate the final estimate $\hat{\theta}$ of θ by averaging the last 1000 of those 2000 iterates. (This average is the MCMC approximation of the posterior mean of θ given $Y_{1:T}$ and $S_{1:T}$.) We compared the mean posterior estimate of θ and the true value, and the performance measure was taken as the TV distance between $\text{Cat}(\theta^*)$ and $\text{Cat}(\hat{\theta})$, that is,

$$\frac{1}{2}\sum_{i=1}^{K} |\hat{\theta}_i - \theta_i|. \tag{5.21}$$

Finally, the comparison among the methods was repeated for all the combinations $(K, \epsilon, \kappa, \rho)$ of $K \in \{10, 20\}$, $\epsilon \in \{0.5, 1, 5\}$, $\kappa \in \{0.8, 0.9\}$, and $\rho \in \{0.01, 0.1, 1\}$.

The accuracy results for the methods in comparison are summarized in Figures 5.3 and 5.4 in terms of the error given in (5.21). The box plots are centered at the error median, and the whiskers stretch from the minimum to the maximum over the 50 MC runs, excluding the outliers. When the medians are compared, the fully adaptive algorithms, which use a utility function to select S_t, yield comparable results to the best semi-adaptive approach in both figures. As one may expect, the non-adaptive approach yielded the worst results in general, especially in the high-privacy regimes (smaller ϵ) and uneven θ^* (smaller ρ). We also observe that, while most utility metrics are generally robust, the one based on FIM seems sensitive to the choice of ϵ_1 parameter. This can be attributed to the fact that the FIM approaches singularity when ϵ_2 is too small, which is the case if ϵ_1 is chosen too close to ϵ. Supporting this, we see that when $\epsilon_1 = 0.8\epsilon$, the utility metric based on FIM becomes more robust. Another remarkable observation is that the utility function based on the

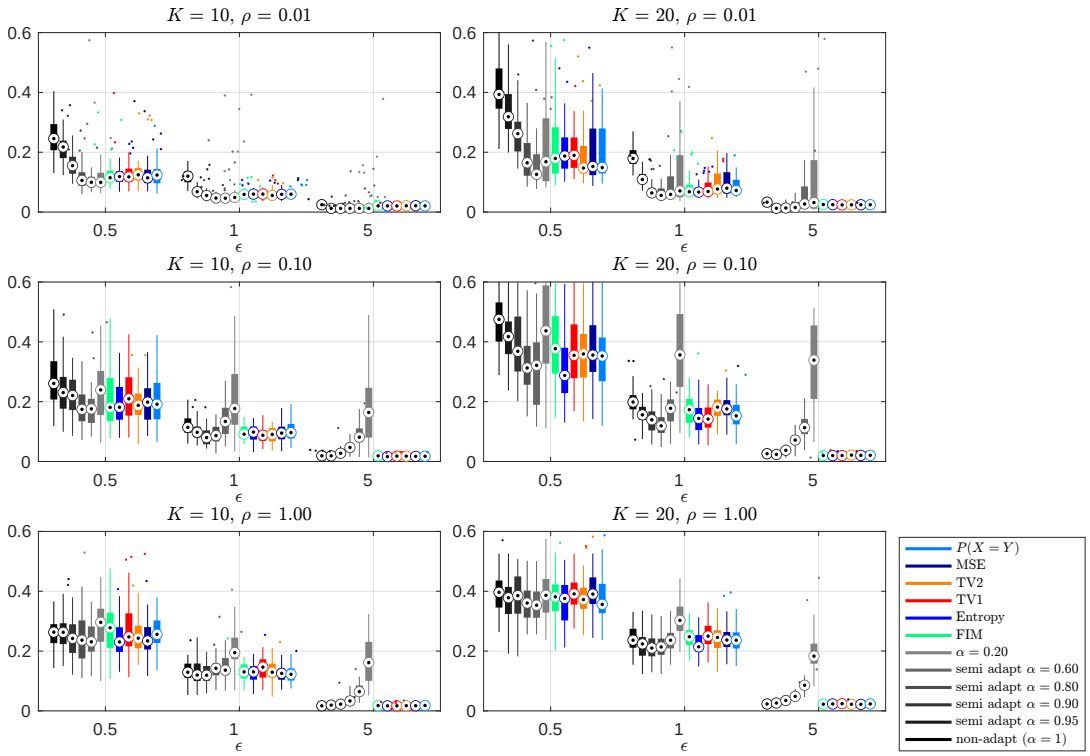


Figure 5.3 TV distance in (5.21) for $K \in \{10, 20\}$, $\epsilon_1 = 0.8\epsilon$

probability of honest response, U_6, has competitive performance despite being the lightest utility metric in computational complexity. Finally, while the semi-adaptive approach is computationally less demanding than most fully adaptive versions, the results show it can dramatically fail if its α hyperparameter is not tuned properly. In contrast, the fully adaptive approaches adapt well to ϵ or ρ and do not need additional tuning.

In addition to the error graphs, the heat maps in Figures 5.5 and 5.6 show the effect of parameters ρ and ϵ on the average cardinality of the subsets S chosen by each algorithm (again, averaged over 50 Monte Carlo runs). According to these figures, increasing the value of ρ causes an increase in the cardinalities of subsets chosen by each algorithm (except the nonadaptive one since it uses all K categories rather than a smaller subset). This is expected since higher ρ values cause $\text{Cat}(\theta^*)$ to be closer to the uniform distribution, thus causing X to be more evenly distributed among the categories. Moreover, for small ρ, increasing the value of ϵ causes a decrease in the cardinalities of these subsets, which can be attributed to a higher ϵ, leading to a more accurate estimation. When we compare the utility functions for the adaptive approach among themselves, we observe that for $\epsilon_1 = 0.8\epsilon$, the third utility function (TV1) uses the subsets with the largest cardinality (on average). However, when we increase the ϵ_1 value to $\epsilon_1 = 0.9\epsilon$, the second utility function (FIM) uses the subsets with the largest cardinality. This might be due to the sensitivity of the FIM-based

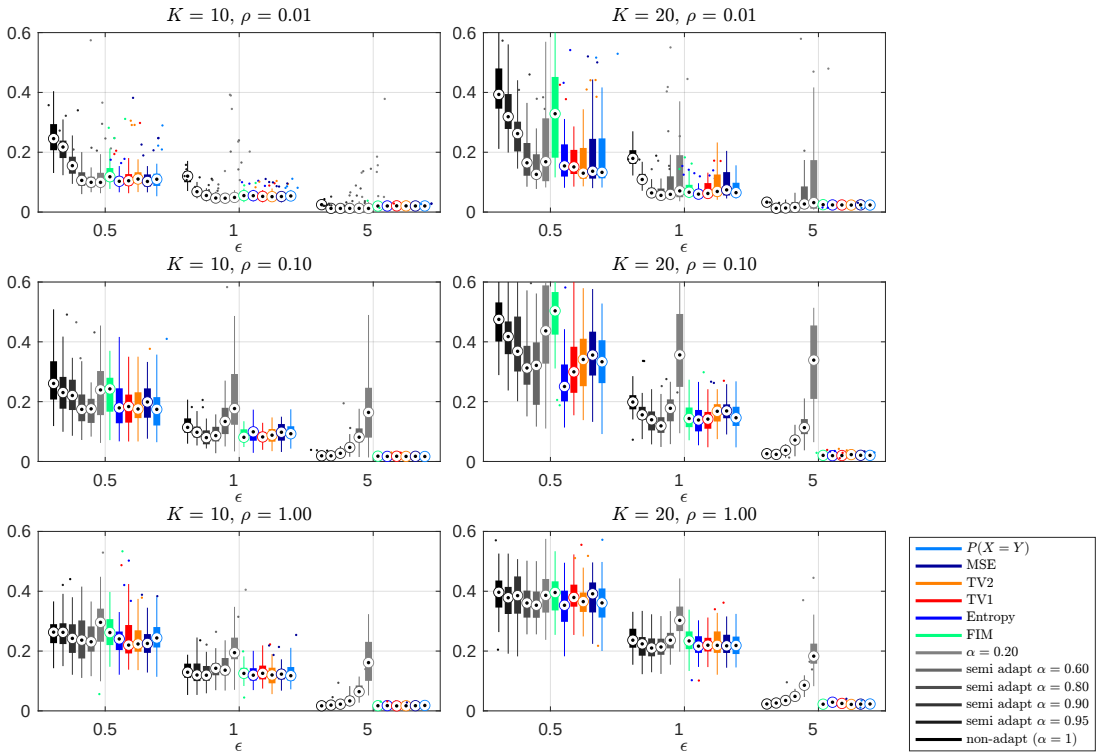


Figure 5.4 TV distance in (5.21) for $K \in \{10, 20\}$, $\epsilon_1 = 0.9\epsilon$

utility function to the choice of ϵ_1 parameter that we mentioned before, which affects the invertibility of the Fisher information matrix when ϵ_1 is too close to ϵ.

5.8 Conclusion

In this chapter, we proposed a new adaptive framework, AdOBEst-LDP, for online estimation of the distribution of categorical data under the ϵ-LDP constraint. AdOBEst-LDP, run with RRRR for randomization, encompasses both privatization of the sensitive data and accurate Bayesian estimation of population parameters from privatized data in a dynamic way. Our privatization mechanism (RRRR) is distinguished from the baseline approach (SRR) in a way that it operates on a smaller subset of the sample space rather than the entire sample space. We employed an adaptive approach to dynamically adjust the subset at each iteration, based on the knowledge about θ^* obtained from the past data. The selection of these subsets was guided by various alternative utility functions that we used throughout this chapter. For the posterior sampling of θ at each iteration, we employed an efficient SGLD-based sampling scheme on a constrained region, namely the K-dimensional

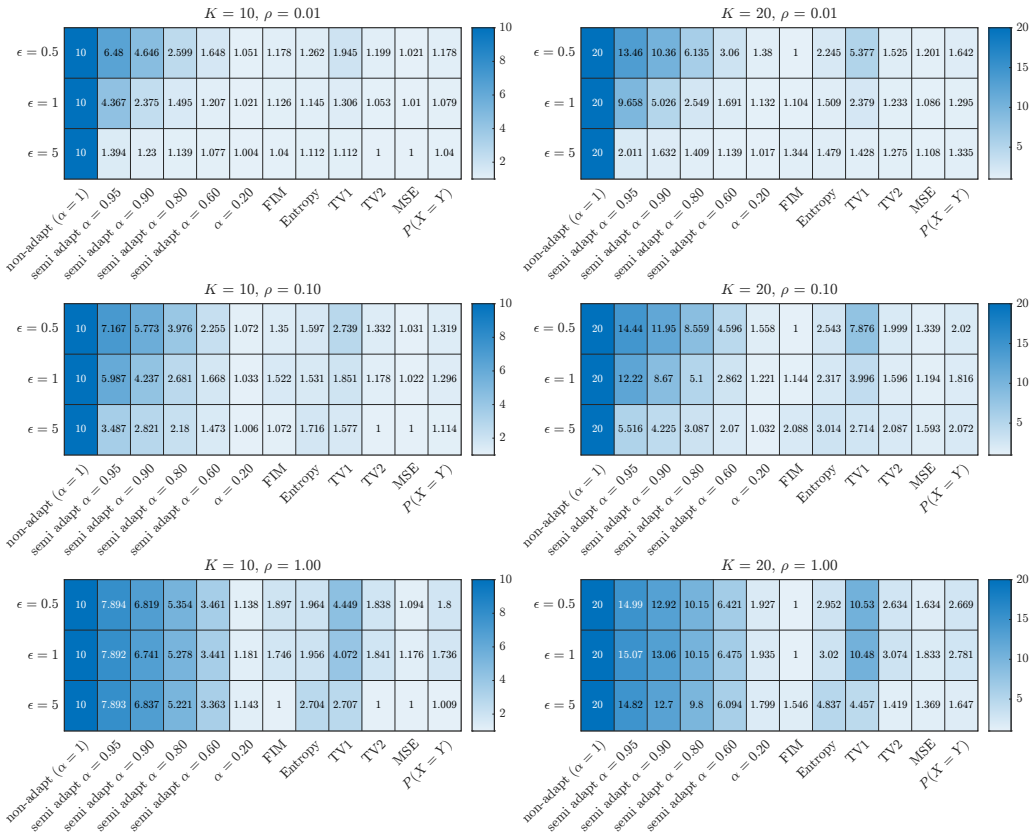


Figure 5.5 Average cardinalities of the subsets selected by each method, for $K \in \{10, 20\}$, $\epsilon_1 = 0.8\epsilon$

probability simplex. We distinguished this scheme from Gibbs sampling, which uses all of the historical data and is not scalable to large datasets.

In the numerical experiments, we demonstrated that AdOBEst-LDP can estimate the population distribution more accurately than the non-adaptive approach under experimental settings with various privacy levels ϵ and degrees of evenness among the components of θ^*. While the performance of AdOBEst-LDP is generally robust for all the utility functions considered in this chapter, the utility function based on the probability of honest response can be preferred due to its much lower computational complexity than the other utility functions. Our experiments also showed that the accuracy of the adaptive approach is comparable to that of the semi-adaptive approach. However, the semi-adaptive approach requires adjusting its parameter α carefully, which makes it challenging to use.

In a theoretical analysis, we showed that, regardless of whether the posterior sampling is conducted exactly or approximately, the posterior distribution targeted in AdOBEst-LDP converges to the true population parameter θ^*. We also showed that, under exact posterior sampling, the best subset given utility function is selected with probability 1 in the long run.

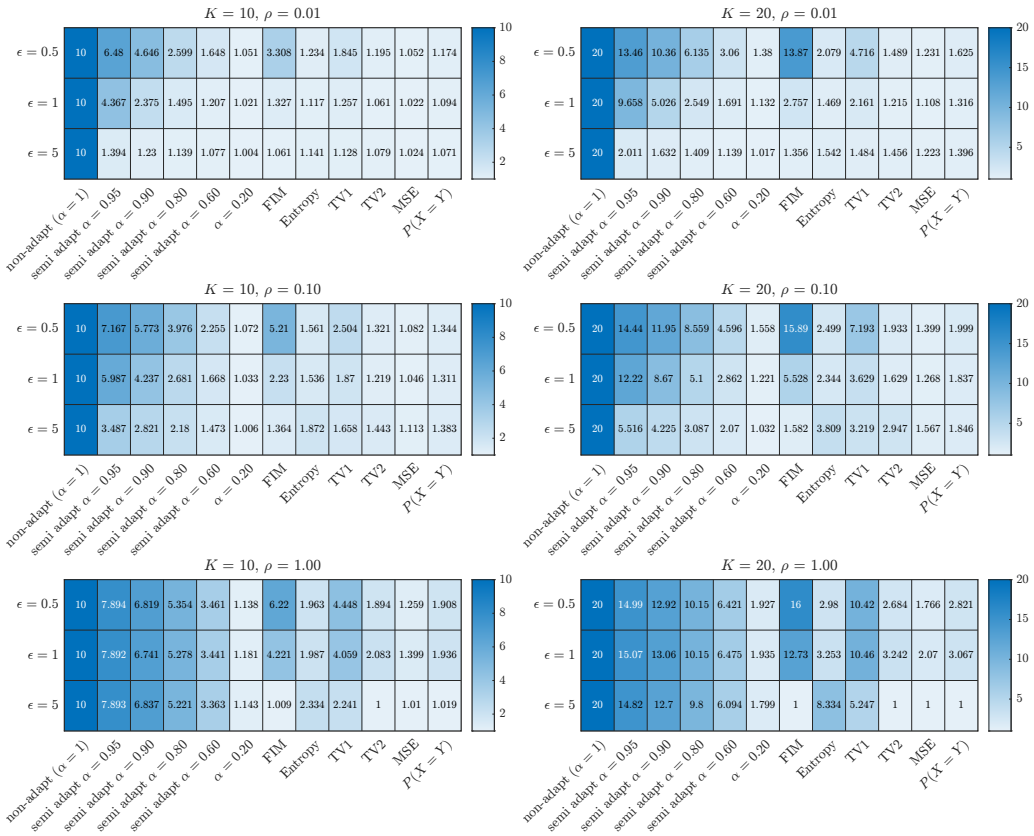


Figure 5.6 Average cardinalities of the subsets selected by each method, for $K \in \{10, 20\}$, $\epsilon_1 = 0.9\epsilon$

It is important to note that the observations $\{Y_t\}_{t\geq 1}$ generated by AdOBEst-LDP are dependent. Therefore, the theoretical analysis presented in Section 5.6 can also be seen as a contribution to the literature on the convergence of posterior distributions with dependent data. Additionally, we have already highlighted an analogy between AdOBEst-LDP and Thompson sampling (Russo et al., 2018). Both methods involve posterior sampling, and the subset selection step in AdOBEst-LDP can be viewed as analogous to the action selection step in reinforcement learning schemes. In this regard, we believe that the theoretical results may also inspire future research on the convergence of dynamic reinforcement learning algorithms, especially those based on Thompson sampling.

Categorical distributions serve as useful non-parametric discrete approximations of continuous distributions. As a potential future direction, AdOBEst-LDP could be adapted for non-parametric density estimation. A key challenge in this context would be determining how to partition the support domain of the data.

Along similar lines, one can propose fitting a mixture of Gaussians (under LDP) to approximate a more complex continuous density. In this mixture model, the mixture weights would correspond to θ_i's in our original categorical density estimation prob-

lem, but now for each category i, one would also need to estimate the location and scale parameters (μ_i and Σ_i, respectively) of the entities in that category. In that case, the SGLD algorithm that we used for posterior sampling may not be efficient enough to estimate all these parameters, especially when μ_i and Σ_i are multivariate. For this purpose, *variational Bayesian* methods could be more efficient to update all parameters. In other words, one could replace the SGLD step for posterior sampling with variational updates for each parameter, which may handle convergence of the parameters to their posterior modes more efficiently in higher dimensions.

RRRR is a practical LDP mechanism with a subset parameter that adapts based on past data. It has been shown to outperform SRR when leveraging the knowledge of θ^*. However, in this work, it is *not* proven that RRRR is the *optimal* ϵ-LDP mechanism with respect to the utility functions considered. While the optimal ϵ-LDP mechanism could be identified numerically by solving a constrained optimization problem—where the utility function is maximized under the LDP constraint—it may not have a closed-form solution for complex utility functions. A promising direction for future research would be to compare the optimal ϵ-LDP mechanism with the ϵ-LDP RRRR mechanism by analyzing their transition probability matrices and assessing the suboptimality of RRRR. Additionally, insights from the optimal ϵ-LDP mechanism could inspire the development of new, tractable, and approximately optimal ϵ-LDP mechanisms.

Further Comparison with State-of-the-Art Methods

Let $P_{xy} = P(X = Y)$ denote the entries of a stochastic matrix P that is used for creating the randomized response Y, and θ_x denote the current probability estimate of X. Then, the generic form of the optimization model to determine the optimal mechanism is as follows:

$$\begin{aligned} \underset{P_{xx}}{\text{maximize}} \quad & U(P_{xx}; \theta_x) \\ \text{subject to} \quad & P_{xy} \leq e^{\epsilon} P_{x'y}, \quad \forall x, x', y \in \{1, \dots K\} \\ & P_{xy} \geq 0 \\ & \textstyle\sum_{y=1}^{K} P_{xy} = 1. \end{aligned}$$

Here, one would determine the optimal mechanism by maximizing a given utility function $U(P_{xx}; \theta_x)$, subject to ϵ-LDP constraints and probability simplex constraints.

To demonstrate empirically that RRRR (while being suboptimal) outperforms SRR significantly, we conducted some Monte Carlo experiments where we compare RRRR, SRR and the "optimal mechanism" with respect to two objective

functions: (i) Maximizing the linear objective function $U_{LP}(P_{xx};\theta_x) = \sum_{x=1}^{K} \theta_x P_{xx}$ which is based on the probability of honest response as in the utility function U_6 that we used before, (ii) Maximizing the (negative) cross-entropy utility function $U_{CE}(P_{xx};\theta_x) = \sum_{x=1}^{K} \theta_x \log(P_{xx})$ which is equivalent to minimizing the distance between true θ and its estimate. We conducted these experiments for the combinations of $\epsilon \in \{0.1, 0.5, 1\}$, $\rho \in \{0.5, 1, 2\}$, $\kappa \in \{0.8, 0.9\}$, $K = 20$ values with 50 MC runs for each combination. The results are summarized in Figures 5.7 - 5.10. According to these plots, RRRR outperforms SRR most of the time in high-privacy regimes (smaller ϵ values), while it is suboptimal in comparison to the optimal mechanism. Here, each optimal mechanism is found by solving the given constrained optimization problem (by using CVXR library in R) whose number of constraints are in the order of K^3; so, for $K = 20$, the number of constraints are in the order of 8000. Even for such a small value of K, solution of each of these optimization problems takes around 40 minutes on a laptop with with 16 GB RAM and Intel Core i7-10510U CPU with clock rate 1.80 GHz, whereas RRRR yields a good approximate solution in less than a second. So, our proposed mechanism would be very useful in applications where the individuals' data arrives at a quick pace and the frequency estimations are required to be updated in accordance with that pace.

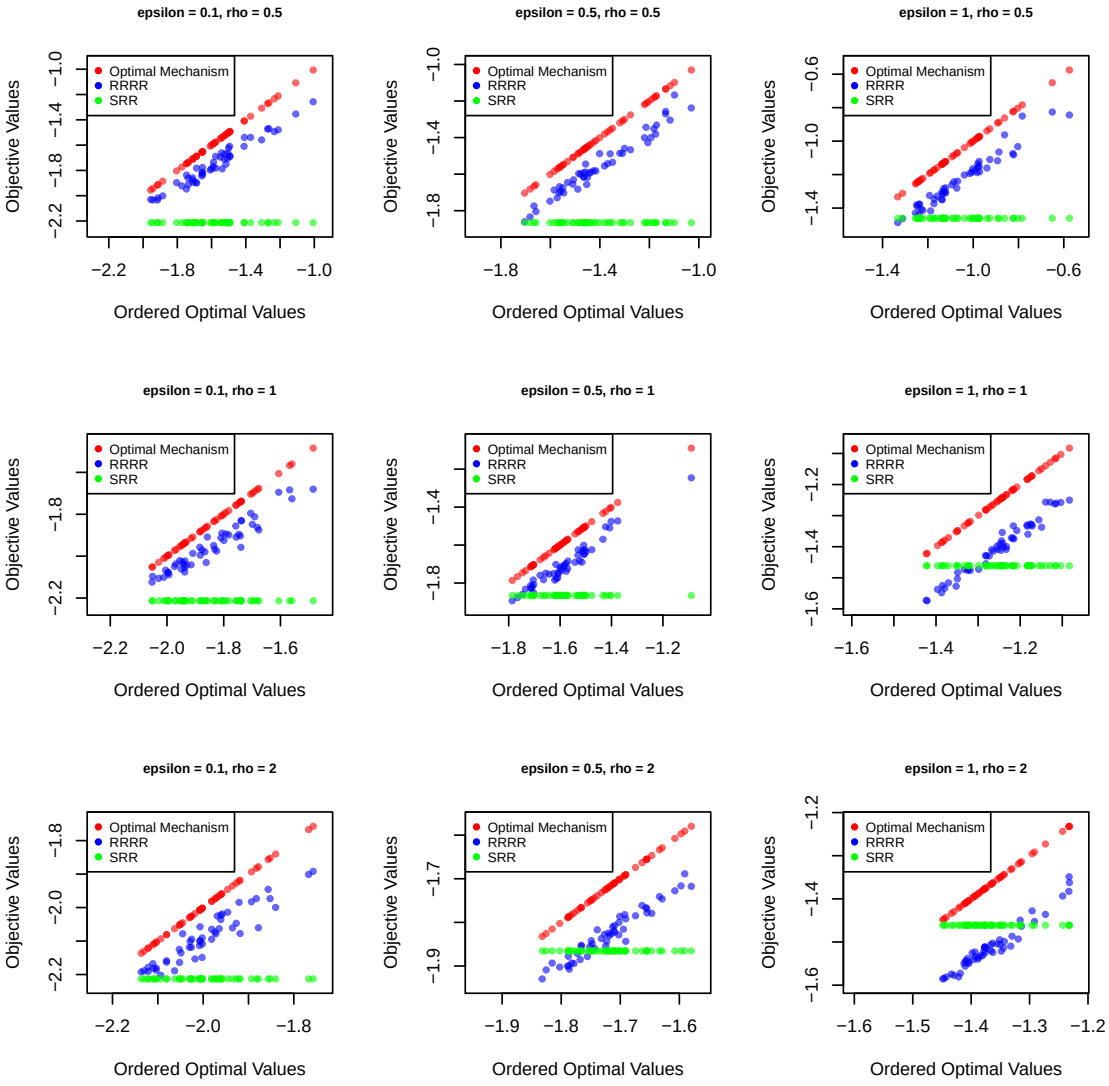


Figure 5.7 Comparison of RRRR, SRR, and the optimal mechanism for $\kappa = 0.8$ and the cross-entropy utility function

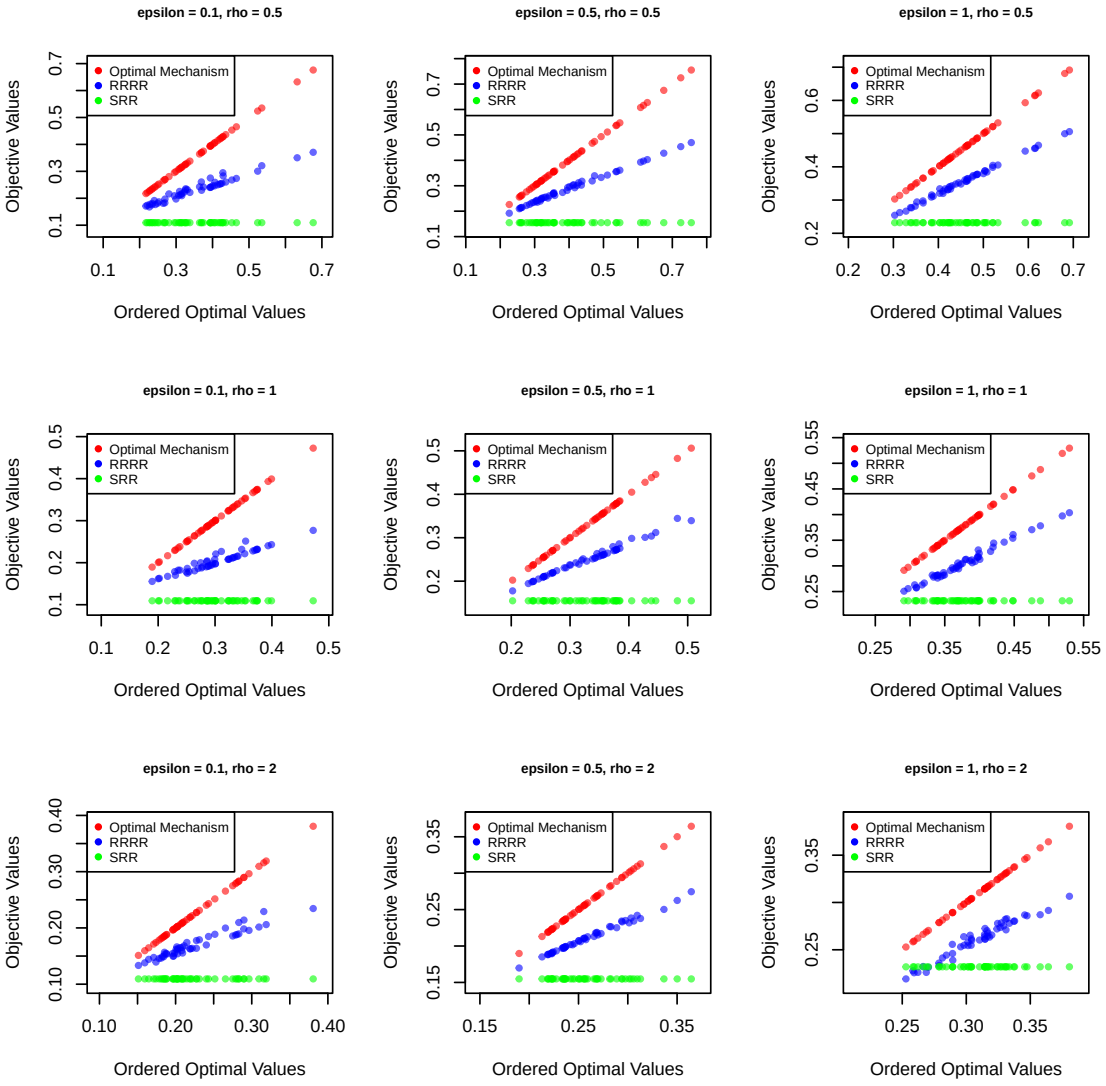


Figure 5.8 Comparison of RRRR, SRR, and the optimal mechanism for $\kappa = 0.8$ and the linear utility function

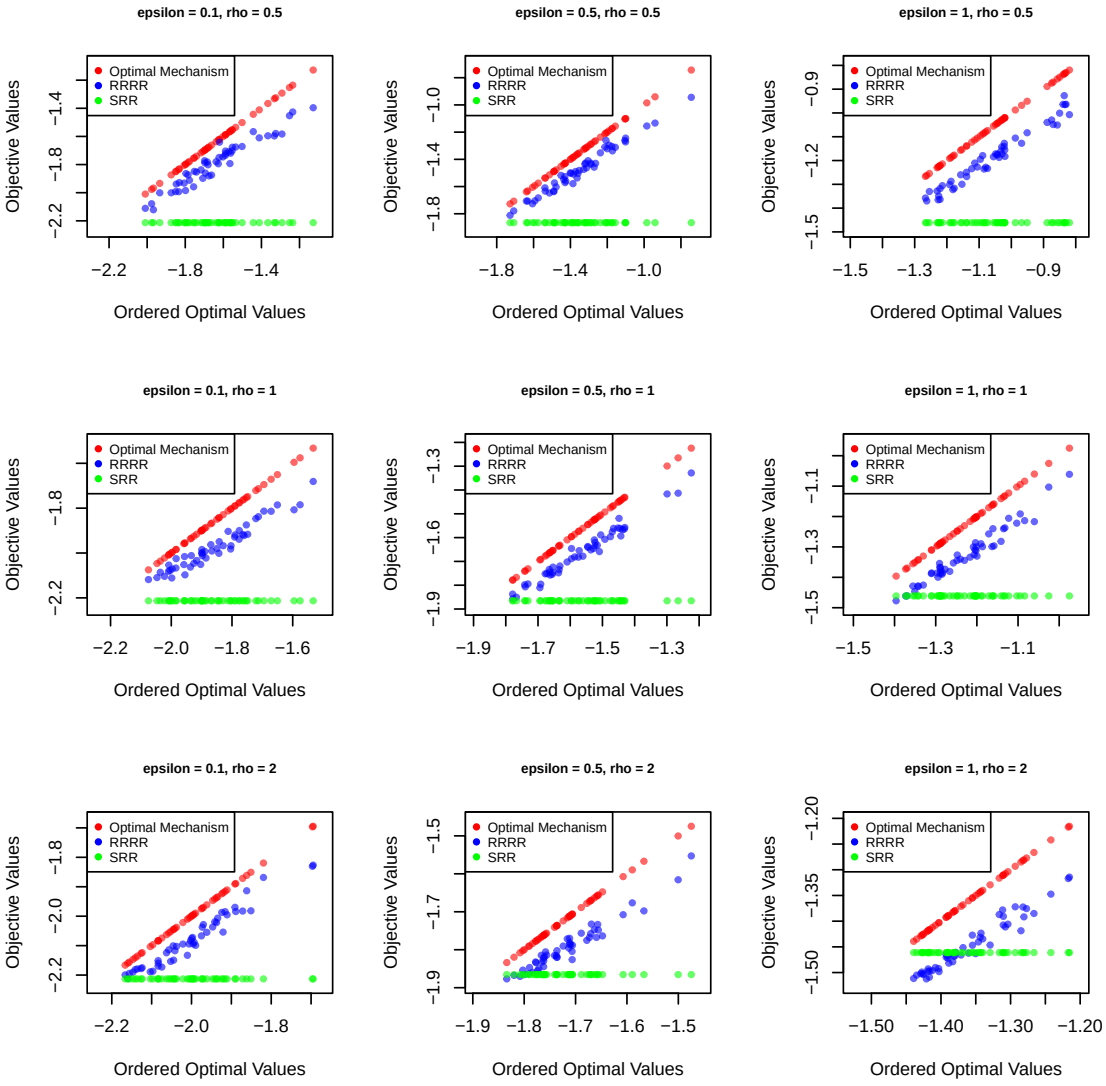


Figure 5.9 Comparison of RRRR, SRR, and the optimal mechanism for $\kappa = 0.9$ and the cross-entropy utility function

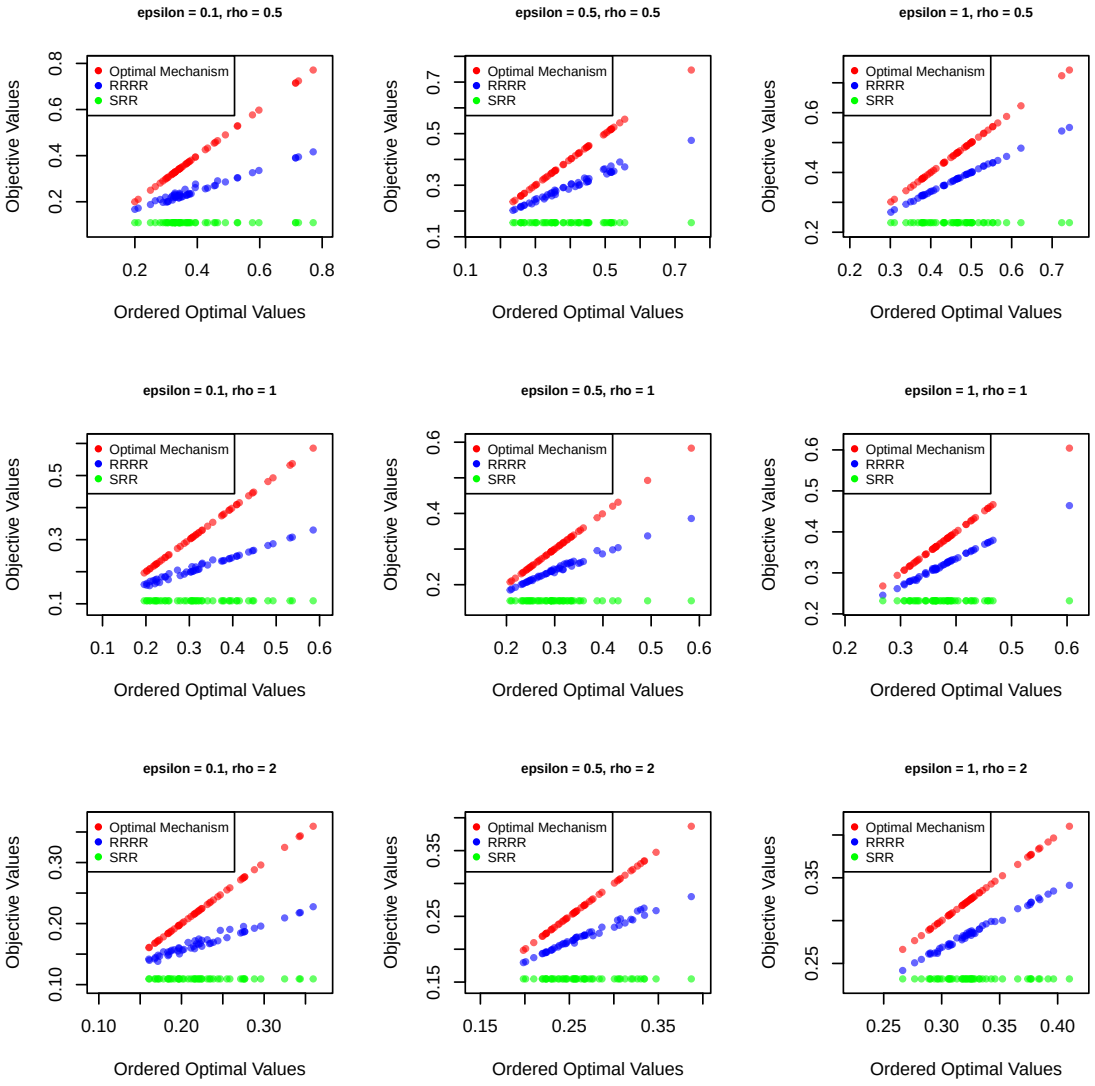


Figure 5.10 Comparison of RRRR, SRR, and the optimal mechanism for $\kappa = 0.9$ and the linear utility function

6. CONCLUSION

Here, we briefly summarize our main conclusions and discussions for each chapter, and add some further comments on potential lines of improvement. More detailed treatments of these points are already included in the conclusion parts of each corresponding chapter.

Chapter 2: In this chapter, we proposed a subsampling-based algorithm for hyper-parameter tuning in regularized linear models. Due to its simplicity, it efficiently finds a good-enough (suboptimal) solution for this problem. On the one hand, its performance is comparable to that of grid-search-based cross-validation. On the other hand there is still a lot of room for improvement for this algorithm. One of the possible directions for improvement is to find a rule for selecting the subsample size and the number of subsamples, as we mentioned before. Another possible direction is to investigate its relation to jackknife estimator further. This might yield a reliable correction term to better estimate the variance of the model parameters. If at least one of these potential improvements come to fruition, one can also consider extending this approach to nonlinear machine learning models.

Chapter 3: In this chapter, the EM algorithm that we proposed to fit a mixture of m-estimators yielded significantly better results than OLS, in terms of test accuracy. However, its performance over each standalone m-estimator seems to be marginally better. Its performance might be further improved by finding a method that handles both robustness and regularization problems simultaneously, by incorporating the methodology of Chapter 2 into this chapter.

Chapter 4: In this chapter, we proposed noisy versions of count-sketch that satisfy (global) differential privacy for large data streams. After investigating the pros and cons of median-perturbation and cell-perturbation approaches for static and dynamic cases, we concluded that, in the static case both have equivalent performance guarantees (in terms of the increase of noise variance), whereas in the dynamic case the Algorithm 6 that uses and keeps the noise in cells is more favorable, since its noise variance increases only linearly. Additionally, we proposed some possible fu-

ture directions of research, including the use of subsamples from a given data stream, applying the same methodology on count-min sketches, extending our methods into pan-privacy, and trying some other randomization mechanisms that are alternative to Laplace mechanism (such as Gaussian mechanism).

Chapter 5: In this chapter, we proposed AdOBEst-LDP algorithm which is a new adaptive framework for online estimation of the distribution parameter of categorical data, while satisfying the ϵ-LDP constraint. Our algorithm, used along with our proposed randomization mechanism RRRR, handles both privatization of the sensitive data and accurate Bayesian estimation of population parameters from privatized data in a dynamic way.

We demonstrated that AdOBEst-LDP outperforms non-adaptive mechanism, and is comparable to semi-adaptive mechanism in terms of accuracy, and argued that it is computationally more efficient than its semi-adaptive counterpart.

We also proved that, the posterior distribution targeted in AdOBEst-LDP converges to the true population parameter θ^*, regardless of whether the posterior sampling is done exactly or approximately. We also proved that our method chooses the best subset in the long run with probability 1, given any utility function that we used, under exact posterior sampling.

In a theoretical analysis, we showed that, regardless of whether the posterior sampling is conducted exactly or approximately, the posterior distribution targeted in AdOBEst-LDP converges to the true population parameter θ^*. We also showed that, under exact posterior sampling, the best subset given utility function is selected with probability 1 in the long run.

Our theoretical analyses in this chapter can be regarded as a contribution to the literature on the convergence of posterior distributions with dependent data, and we noted the analogies between our approach and some other dynamic algorithms in reinforcement learning (especially Thompson sampling) which can motivate potential new research on the convergence properties of these algorithms.

We also suggested some other potential lines of research related to our algorithm. For example, AdOBEst-LDP might be modified to tackle non-parametric density estimation. As another open problem, RRRR could be modified to better approximate the "optimal mechanism" for a given utility function, without explicitly solving a constrained optimization problem.

BIBLIOGRAPHY

Acharya, J., Canonne, C. L., Sun, Z., & Tyagi, H. (2023). Unified lower bounds for interactive high-dimensional estimation under information constraints. In Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., & Levine, S. (Eds.), *Advances in Neural Information Processing Systems*, volume 36, (pp. 51133–51165)., New Orleans, US. Curran Associates, Inc.

Alparslan, B. & Yıldırım, S. (2022). Statistic selection and MCMC for differentially private Bayesian estimation. *Statistics and Computing*, *32*(5), 66.

Arslan, O. & Billor, N. (2000). Robust liu estimator for regression based on an m-estimator. *Journal of applied statistics*, *27*(1), 39–47.

Bai, X., Yao, W., & Boyer, J. E. (2012). Robust fitting of mixture regression models. *Computational Statistics & Data Analysis*, *56*(7), 2347–2359.

Balle, B., Barthe, G., & Gaboardi, M. (2018). Privacy amplification by subsampling: Tight analyses via couplings and divergences. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, (pp. 6280?6290)., Red Hook, NY, USA. Curran Associates Inc.

Balu, R. & Furon, T. (2016). Differentially private matrix factorization using sketching techniques. In *Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security*, (pp. 57–62). ACM.

Barnes, L. P., Chen, W.-N., & Özgür, A. (2020). Fisher information under local differential privacy. *IEEE Journal on Selected Areas in Information Theory*, *1*(3), 645–659.

Bhaila, K., Huang, W., Wu, Y., & Wu, X. (2024). Local differential privacy in graph neural networks: a reconstruction approach. In *Proceedings of the 2024 SIAM International Conference on Data Mining (SDM)*, (pp. 1–9)., Texas, US. SIAM, SIAM.

Bischl, B., Binder, M., Lang, M., Pielok, T., Richter, J., Coors, S., Thomas, J., Ullmann, T., Becker, M., Boulesteix, A.-L., et al. (2023). Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, *13*(2), e1484.

Chaari, L., Batatia, H., Dobigeon, N., & Tourneret, J.-Y. (2014). A hierarchical sparsity-smoothness Bayesian model for $\ell_0 + \ell_1 + \ell 2$ regularization. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, (pp. 1901–1905). IEEE.

Charikar, M., Chen, K., & Farach-Colton, M. (2002). Finding frequent items in data streams. In *Proceedings of the 29th International Colloquium on Automata, Languages and Programming*, ICALP '02, (pp. 693–703)., Berlin, Heidelberg. Springer-Verlag.

Charikar, M., Chen, K., & Farach-Colton, M. (2004). Finding frequent items in data streams. *Theoretical Computer Science*, *312*(1), 3 – 15. Automata, Languages and Programming.

Cormode, G. & Bharadwaj, A. (2022). Sample-and-threshold differential privacy: Histograms and applications. In *International Conference on Artificial Intelligence and Statistics*, (pp. 1420–1431)., Valencia, Spain. PMLR.

Cormode, G., Kulkarni, T., & Srivastava, D. (2017). Constrained differential privacy for count data. *ArXiv*, *abs/1710.00608*.

Cormode, G., Kulkarni, T., & Srivastava, D. (2018). Marginal release under local differential privacy. In *Proceedings of the 2018 International Conference on Management of Data*, (pp. 131–146).

Cormode, G. & Muthukrishnan, S. (2005). An improved data stream summary: the count-min sketch and its applications. *Journal of Algorithms*, *55*(1), 58 – 75.

Cormode, G., Procopiuc, C., Srivastava, D., & Tran, T. T. L. (2012). Differentially private summaries for sparse data. In *Proceedings of the 15th International Conference on Database Theory*, ICDT '12, (pp. 299–311)., New York, NY, USA. ACM.

De Menezes, D., Prata, D. M., Secchi, A. R., & Pinto, J. C. (2021). A review on robust m-estimators for regression analysis. *Computers & Chemical Engineering*, *147*, 107254.

Deisenroth, M. P., Faisal, A. A., & Ong, C. S. (2020). *Mathematics for machine learning*. Cambridge University Press.

Dobson, A. J. & Barnett, A. G. (2018). *An introduction to generalized linear models*. CRC press.

Doğru, F. Z. & Arslan, O. (2021). Robust mixture regression modeling based on the generalized m (gm)-estimation method. *Communications in Statistics-Simulation and Computation*, *50*(9), 2643–2665.

Dwork, C. (2006a). Differential privacy. In Bugliesi, M., Preneel, B., Sassone, V., & Wegener, I. (Eds.), *Automata, Languages and Programming*, (pp. 1–12)., Berlin, Heidelberg. Springer Berlin Heidelberg.

Dwork, C. (2006b). Differential privacy. In *International colloquium on automata, languages, and programming*, (pp. 1–12). Springer.

Dwork, C. (2008). Differential privacy: A survey of results. In *International Conference on Theory and Applications of Models of Computation*, (pp. 1–19). Springer.

Dwork, C., Naor, M., Pitassi, T., Rothblum, G., & Yekhanin, S. (2010). Pan-private streaming algorithms. In *Proceedings of The First Symposium on Innovations in Computer Science (ICS 2010)* (Proceedings of The First Symposium on Innovations in Computer Science (ICS 2010) ed.). Tsinghua University Press.

Dwork, C., Naor, M., Pitassi, T., & Rothblum, G. N. (2010). Differential privacy under continual observation. In *Proceedings of the Forty-Second ACM Symposium on Theory of Computing*, STOC '10, (pp. 715?724)., New York, NY, USA. Association for Computing Machinery.

Foulds, J., Geumlek, J., & an Kamalika Chaudhuri, M. W. (2016). On the theory and practice of privacy-preserving Bayesian data analysis. Technical report, arxiv:1603.07294.

Freeman, P. R. (1979). Algorithm as 145: Exact distribution of the largest multinomial frequency. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, *28*(3), 333–336.

Gill, R. D. & Levit, B. Y. (1995). Applications of the van Trees Inequality: A Bayesian Cramér-Rao Bound. *Bernoulli*, *1*(1/2), 59–79.

Hastie, T., Tibshirani, R., & Wainwright, M. (2015). Statistical learning with sparsity. *Monographs on statistics and applied probability*, *143*(143), 8.

Hastie, T., Tibshirani, R., & Wainwright, M. (2019). *Statistical learning with spar-*

sity: the lasso and generalizations. Chapman and Hall/CRC.

Hoerl, A. E., Kannard, R. W., & Baldwin, K. F. (1975). Ridge regression: some simulations. *Communications in Statistics-Theory and Methods*, *4*(2), 105–123.

Hoerl, A. E. & Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, *12*(1), 55–67.

Huang, L., Wang, H., & Zheng, A. (2014). The m-estimator for functional linear regression model. *Statistics & Probability Letters*, *88*, 165–173.

James, G. (2013). An introduction to statistical learning.

Jia, J. & Gong, N. Z. (2019). Calibrate: Frequency estimation and heavy hitter identification with local differential privacy via incorporating prior knowledge. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*, (pp. 2008–2016). IEEE.

Joseph, M., Kulkarni, J., Mao, J., & Wu, S. Z. (2019). Locally private Gaussian estimation. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., & Garnett, R. (Eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Kairouz, P., Oh, S., & Viswanath, P. (2016). Extremal mechanisms for local differential privacy. *The Journal of Machine Learning Research*, *17*(1), 492–542.

Karwa, V., Slavković, A. B., & Krivitsky, P. (2014). Differentially private exponential random graphs. In Domingo-Ferrer, J. (Ed.), *Privacy in Statistical Databases*, (pp. 143–155)., Cham. Springer International Publishing.

Kasiviswanathan, S. P., Lee, H. K., Nissim, K., Raskhodnikova, S., & Smith, A. (2011). What can we learn privately? *SIAM Journal on Computing*, *40*(3), 793–826.

Kim, C., Jung, J., & Chung, Y. (2011). Bayesian estimation for the exponentiated weibull model under type ii progressive censoring. *Statistical Papers*, *52*(1), 53–70.

Kochenderfer, M. J. & Wheeler, T. A. (2019). *Algorithms for optimization*. Mit Press.

Kuhn, M. & Silge, J. (2022). *Tidy Modeling with R*. " O'Reilly Media, Inc.".

Li, T., Liu, Z., Sekar, V., & Smith, V. (2019). Privacy for free: Communication-efficient learning with differential privacy using sketches. *arXiv preprint arXiv:1911.00972*.

Lin, Y. & Lee, D. D. (2006). Bayesian l 1-norm sparse learning. In *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, volume 5, (pp. V–V). IEEE.

Liu, T., Zhang, L., Jin, G., & Pan, Z. (2022). Reliability assessment of heavily censored data based on e-bayesian estimation. *Mathematics*, *10*(22).

Lone, S. A., Panahi, H., Anwar, S., & Shahab, S. (2024). Inference of reliability model with burr type xii distribution under two sample balanced progressive censored samples. *Physica Scripta*, *99*(2), 025019.

Lopuhaä-Zwakenberg, M., Škorić, B., & Li, N. (2022). Fisher information as a utility metric for frequency estimation under local differential privacy. In *Proceedings of the 21st Workshop on Privacy in the Electronic Society*, (pp. 41–53).

MacKay, D. J. (1995). Probable networks and plausible predictions-a review of practical bayesian methods for supervised neural networks. *Network: computation in neural systems*, *6*(3), 469.

Mazumdar, E., Pacchiano, A., Ma, Y.-A., Bartlett, P. L., & Jordan, M. I. (2020). On approximate Thompson sampling with Langevin algorithms. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org.

Melis, L., Danezis, G., & Cristofaro, E. D. (2016). Efficient private statistics with succinct sketches. In *23rd Annual Network and Distributed System Security Symposium, NDSS 2016, San Diego, California, USA, February 21-24, 2016*. The Internet Society.

Mir, D., Muthukrishnan, S., Nikolov, A., & Wright, R. N. (2011). Pan-private algorithms via statistics on sketches. In *Proceedings of the Thirtieth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '11, (pp. 37?48)., New York, NY, USA. Association for Computing Machinery.

Mishra, N. & Sandler, M. (2006). Privacy via pseudorandom sketches. In *Proceedings of the Twenty-Fifth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '06, (pp. 143–152)., New York, NY, USA. Association for Computing Machinery.

Monreale, A., Wang, W. H., Pratesi, F., Rinzivillo, S., Pedreschi, D., Andrienko, G., & Andrienko, N. (2013). Privacy-preserving distributed movement data aggregation. In *Geographic Information Science at the Heart of Europe* (pp. 225–245). Springer.

Murphy, K. P. (2023). *Probabilistic machine learning: Advanced topics*. MIT press.

Neal, R. M. (1995). *BAYESIAN LEARNING FOR NEURAL NETWORKS*. PhD thesis, Citeseer.

Noor-Ul-Amin, M., Asghar, S. U. D., Sanaullah, A., & Shehzad, M. A. (2018). Redescending m-estimator for robust regression. *Journal of Reliability and Statistical Studies*, 69–80.

Pelekis, C. & Ramon, J. (2017). Hoeffding's inequality for sums of dependent random variables. *Mediterranean Journal of Mathematics*, *14*(6), 243.

Qian, K. (2017). On the determination of proper regularization parameter: α-weighted ble via a-optimal design and its comparison with the results derived by numerical methods and ridge regression. B.S. thesis.

Ramakrishna, M. V. (1988). An exact probability model for finite hash tables. In *Proceedings. Fourth International Conference on Data Engineering*, (pp. 362–368).

Robert, C. P., Casella, G., & Casella, G. (2010). *Introducing monte carlo methods with r*, volume 18. Springer.

Russo, D., Roy, B. V., Kazerouni, A., Osband, I., & Wen, Z. (2018). A tutorial on thompson sampling. *Foundations and Trends in Machine Learning*, *11*(1), 1–96.

Shao, J. & Wu, C. J. (1989). A general theory for jackknife variance estimation. *The annals of Statistics*, 1176–1197.

Sparka, H., Tschorsch, F., & Scheuermann, B. (2018). P2kmv: A privacy-preserving counting sketch for efficient and accurate set intersection cardinality estimations. Cryptology ePrint Archive, Report 2018/234.

Steinberger, L. (2024). Efficiency in local differential privacy.

Susanti, Y., Pratiwi, H., Sulistijowati, S., Liana, T., et al. (2014). M estimation, s estimation, and mm estimation in robust regression. *International Journal of Pure and Applied Mathematics*, *91*(3), 349–360.

Tak, H., Ellis, J. A., & Ghosh, S. K. (2019). Robust and accurate inference via a mixture of gaussian and student'st errors. *Journal of Computational and Graphical Statistics*, *28*(2), 415–426.

Tipping, M. E. (2001). Sparse bayesian learning and the relevance vector machine. *Journal of machine learning research*, *1*(Jun), 211–244.

von Voigt, S. N. & Tschorsch, F. (2019). Rrtxfm: Probabilistic counting for differentially private statistics. In *IACR Cryptol. ePrint Arch.*

Wang, M., Jiang, H., Peng, P., & Li, Y. (2024). Accurately estimating frequencies of relations with relation privacy preserving in decentralized networks. *IEEE Transactions on Mobile Computing*, *23*(05), 6408–6422.

Wang, S., Huang, L., Wang, P., Nie, Y., Xu, H., Yang, W., Li, X.-Y., & Qiao, C. (2016). Mutual information optimally local private discrete distribution estimation.

Wang, S., Li, Y., Zhong, Y., Chen, K., Wang, X., Zhou, Z., Peng, F., Qian, Y., Du, J., & Yang, W. (2024). Locally private set-valued data analyses: Distribution and heavy hitters estimation. *IEEE Transactions on Mobile Computing [preprint]*, 1–14.

Wang, T., Blocki, J., Li, N., & Jha, S. (2017). Locally differentially private protocols for frequency estimation. In *26th USENIX Security Symposium (USENIX Security 17)*, (pp. 729–745).

Wang, T., Lopuhaä-Zwakenberg, M., Li, Z., Skoric, B., & Li, N. (2020). Locally differentially private frequency estimation with consistency. In *27th Annual Network and Distributed System Security Symposium, NDSS 2020*. Cited by: 33; All Open Access, Bronze Open Access, Green Open Access.

Waudby-Smith, I., Wu, S., & Ramdas, A. (2023). Nonparametric extensions of randomized response for private confidence sets. In *International Conference on Machine Learning*, (pp. 36748–36789). PMLR.

Wei, F., Bao, E., Xiao, X., Yang, Y., & Ding, B. (2024). Aaa: an adaptive mechanism for locally differential private mean estimation.

Welling, M. & Teh, Y. W. (2011a). Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, (pp. 681–688)., Madison, WI, USA. Omnipress.

Welling, M. & Teh, Y. W. (2011b). Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, (pp. 681–688). Citeseer.

Williams, O. & Mcsherry, F. (2010). Probabilistic inference and differential privacy. In Lafferty, J., Williams, C., Shawe-Taylor, J., Zemel, R., & Culotta, A. (Eds.), *Advances in Neural Information Processing Systems*, volume 23. Curran Associates, Inc.

Wipf, D. & Nagarajan, S. (2010). Iterative reweighted ℓ_1 and ℓ_2 methods for finding sparse solutions. *IEEE Journal of Selected Topics in Signal Processing*, *4*(2), 317–329.

Xiang, D. (2020). *Fully Bayesian Penalized Regression with a Generalized Bridge Prior*. PhD thesis, University of Minnesota.

Yıldırım, S. (2024). Differentially private online bayesian estimation with adaptive truncation. *Turkish Journal of Electrical Engineering and Computer Sciences*, *32*(2), 34–50.

Zhang, Z. (1997). Parameter estimation techniques: A tutorial with application to conic fitting. *Image and vision Computing*, *15*(1), 59–76.

Zhao, D., Zhao, S.-Y., Chen, H., Liu, R.-X., Li, C.-P., & Zhang, X.-Y. (2023). Hadamard encoding based frequent itemset mining under local differential privacy. *Journal of Computer Science and Technology*, *38*(6), 1403–1422.

Zhu, Y., Cao, Y., Xue, Q., Wu, Q., & Zhang, Y. (2024). Heavy hitter identification over large-domain set-valued data with local differential privacy. *IEEE Transactions on Information Forensics and Security*, *19*, 414–426.

Zou, T., Li, X., Liang, X., & Wang, H. (2021). On the subbagging estimation for massive data. *arXiv preprint arXiv:2103.00631*.

A. Supplementary Material for Chapter 2

A.1 Derivation of the Analytical Solution for Ridge Regression

Consider the ridge estimate obtained from the training set, $\hat{\beta} = (X_R^T X_R + vI)^{-1} X_R^T Y_R$, where I is an identity matrix. The model is $Y = X\beta + e$, where the exact β is unknown $e \sim \mathcal{N}(0, \sigma I)$ and σ is the unknown standard deviation of the residuals (e). For convenience, we assume that columns of X are standardized and Y is centered; in this case, the intercept term will be zero, so it is not included in β, and will not be penalized. For an estimator $\hat{\beta}$, the total test error is

$$\|Y_S - X_S\hat{\beta}\|^2 = Y_S^T Y_S + \hat{\beta}^T X_S^T X_S \hat{\beta} - 2Y_S^T X_S \hat{\beta}.$$

At the same time, we have

$$\|Y_S - X_S\hat{\beta}\|^2 = \|X_S(\beta - \hat{\beta}) + e\|^2 = (\hat{\beta} - \beta)^T X_S^T X_S (\hat{\beta} - \beta) - e^T X_S^T X_S (\hat{\beta} - \beta) + e^T e$$

The last term does not depend on β or its estimator. Also, note that $\hat{\beta}$ depends on the training data, therefore independent of X_S and Y_S. Therefore, the expected total test error with respect to the distribution of X_S, Y_S is, up to an additive constant,

$$\frac{1}{n}(\hat{\beta} - \beta)^T E[X_S^T X_S](\hat{\beta} - \beta)$$

The expectation of $X_S^T X_S$ is nS where $S = E[xx^T]$ and the random variable x^T represents a row of X. Therefore, we have

$$MSE(\beta; X, y) = (\hat{\beta} - \beta)^T S (\hat{\beta} - \beta) = \mathrm{tr}((\hat{\beta} - \beta)(\hat{\beta} - \beta)^T S).$$

This is the conditional expectation of MSE given the training data X_R and Y_R. Specifically, $\hat{\beta}$ depends on X_R and Y_R. Here, X_R and Y_R are also random variables, sampled from a population of X and Y. Therefore, for an overall performance measure, we need to consider the expectation of $MSE(\beta; X, Y)$. We can decompose the MSE into its components, namely, squared bias and variance, as follows.

Call $U = X^T X$ and $H = U + vI$. The bias of $\hat{\beta}$ is

$$E[\hat{\beta} - \beta] = (H^{-1}U - I)\beta,$$

And the variance of $\hat{\beta}$ is

$$Cov(\hat{\beta}) = \sigma^2 H^{-1} U H^{-1}.$$

Therefore,

$$E[(\hat{\beta} - \beta)(\hat{\beta} - \beta)^T] = (H^{-1}U - I)\beta\beta^T(H^{-1}U - I) + \sigma^2 H^{-1} U H^{-1}.$$

Some lemmas that can be useful: $H^{-1} = (U + vI)^{-1} = \frac{1}{v}(\frac{U}{v} + I)^{-1} = \frac{1}{v} v U^{-1}(vU^{-1} + I)^{-1} = U^{-1}(vU^{-1} + I)^{-1} = \frac{1}{v} U^{-1}(U^{-1} + \frac{1}{v} I)^{-1}$. For a small scalar $a \ll Q, M$, by Taylor series approximation, we have

$$(Q + aM)^{-1} \approx Q^{-1} - aQ^{-1}MQ^{-1}.$$

Apply this to $H = (U + \frac{v}{n}(nI))$ to get

$$H^{-1} \approx U^{-1} - \frac{v}{n} U^{-1} n I U^{-1} = U^{-1} - vU^{-2}.$$

With this, we have

$$H^{-1}U - I = U^{-1}U - \frac{v}{n} U^{-1} n I U^{-1} U - I \approx -vU^{-1}.$$

Hence, the bias term is

$$(H^{-1}U - I)\beta\beta^T(H^{-1}U - I) \approx v^2 U^{-1}\beta\beta^T U^{-1}.$$

Also, we have

$$H^{-1}UH^{-1} \approx (U^{-1} - vU^{-2})U(U^{-1} - vU^{-2}) = (I - vU^{-1})(U^{-1} - vU^{-2}) = (I - vU^{-1})^2 U^{-1}$$

Putting everything together,

$$MSE(\beta; X, y) \approx \mathrm{tr}\left\{S\left[\sigma^2(I - vU^{-1})^2U^{-1} + v^2U^{-1}\beta\beta^TU^{-1}\right]\right\} \tag{A.1}$$

$$= \mathrm{tr}\left\{\left[\sigma^2(I - vU^{-1})^2U^{-1}S + v^2U^{-1}\beta\beta^TU^{-1}S\right]\right\}. \tag{A.2}$$

Using the approximation $U \approx nS$,

$$MSE(\beta) := E[MSE(\beta; X, y)] \approx \frac{1}{n}\mathrm{tr}\left\{\left[\sigma^2(I - \frac{v}{n}S^{-1})^2 + \frac{v^2}{n}S^{-1}\beta\beta^T\right]\right\}$$

$$= \frac{1}{n}\mathrm{tr}\left\{\left[\sigma^2(I + \frac{v^2}{n^2}S^{-2} - 2\frac{v}{n}S^{-1}) + \frac{v^2}{n}S^{-1}\beta\beta^T\right]\right\}. \tag{A.3}$$

The derivative is equal to

$$\frac{dMSE(\beta)}{dv} = \frac{2}{n^2}\left[v\left(\frac{\sigma^2\mathrm{tr}S^{-2}}{n} + \mathrm{tr}S^{-1}\beta\beta^T\right) - \sigma^2\mathrm{tr}S^{-1}\right].$$

Optimal v is

$$v_n = \frac{\mathrm{tr}S^{-1}}{\frac{1}{n}\mathrm{tr}S^{-2} + \frac{1}{\sigma^2}\mathrm{tr}(S^{-1}\beta\beta^T)} = \frac{\mathrm{tr}S^{-1}}{\frac{1}{n}\mathrm{tr}S^{-2} + \frac{1}{\sigma^2}\beta^TS^{-1}\beta}.$$

This converges to a constant

$$\lim_{n\to\infty} v_n = \sigma^2\frac{\mathrm{tr}S^{-1}}{\mathrm{tr}(S^{-1}\beta\beta^T)} = \sigma^2\frac{\mathrm{tr}S^{-1}}{\beta^TS^{-1}\beta}.$$

This solution yields a scalar value for λ (denoted as v_n above).

A.2 Additional Experiments for Ridge Regression

In table A.1, we compare the closed-form approximate solution that we found for ridge regression, with CV-based method and subsampling-based method for datasets which were randomly generated with different settings. Here (N, d) denotes the number of rows and columns of X, respectively; *normal* denotes that X was generated from standard normal distribution, *norm-IW* denotes that X again comes from normal distribution, but this time its covariance matrix is drawn from *Inverse Wishart* distribution in a way that its features are set to be highly correlated among themselves. $sd(noise)$ denotes the standard deviation of noise, and *Sparsity of* (β)

denotes the ratio of sparse elements in the actual β vector. The results are averaged over 100 runs with random partitions of training and testing sets.

Table A.1 Closed-form Ridge Results

(N, d)	**X distr.**	**sd(noise)**	**Sparsity of β**	MSE_{OLS}	MSE_{CV}	$MSE_{Subsampling}$	MSE_{Closed}	$Time_{CV}$	$Time_{Subsampling}$	$Time_{Closed}$
2000, 100	normal	100	0.25	10553.8081	10350.5265	10555.1587	10379.8920	0.3395	0.0485	0.0479
2000, 100	normal	100	0.5	11024.5700	10703.4329	11016.2349	10774.0981	0.3426	0.0489	0.0465
2000, 100	normal	100	0.75	10749.2366	10451.3828	10745.2048	10516.1456	0.3862	0.0543	0.0514
2000, 100	normal	10	0.25	110.3539	111.1651	114.8423	110.3388	0.4324	0.0561	0.0561
2000, 100	normal	10	0.5	100.6333	101.3003	106.1821	100.5961	0.4566	0.0581	0.0566
2000, 100	normal	10	0.75	108.8399	109.5543	115.2301	108.8011	0.6772	0.0867	0.0815
1000, 100	normal	100	0.25	12360.2334	11502.5767	12324.3266	11744.1897	0.5367	0.0518	0.0466
1000, 100	normal	100	0.5	11762.5437	10822.2656	11727.7128	11143.1831	0.6437	0.0636	0.0539
1000, 100	normal	100	0.75	11505.5780	10386.9769	11466.8181	10816.0886	0.4163	1.2343	0.0372
1000, 100	normal	10	0.25	119.6367	120.0197	135.8895	119.3979	0.4705	0.0488	0.0401
1000, 100	normal	10	0.5	114.0341	114.5024	131.4961	113.9083	0.7155	0.0677	0.0538
1000, 100	normal	10	0.75	113.6015	113.9641	128.1552	113.4321	0.7051	0.0656	0.0548
2000, 100	norm-IW	100	0.25	10817.5296	10271.7158	10662.1778	10807.9304	0.6703	0.0748	0.0741
2000, 100	norm-IW	100	0.5	10843.9970	10212.3545	10696.4575	10819.1558	0.7299	0.0779	0.0755
2000, 100	norm-IW	100	0.75	10758.2274	10155.4350	10606.1876	10737.5505	0.7203	0.0717	0.0708
2000, 100	norm-IW	10	0.25	107.2539	118.8977	104.8767	107.0886	0.4125	0.0516	0.0496
2000, 100	norm-IW	10	0.5	106.4656	111.7489	104.4660	106.2962	0.8345	0.0957	0.0905
2000, 100	norm-IW	10	0.75	107.0267	107.1411	104.1973	106.7551	0.6220	0.0685	0.0673
1000, 100	norm-IW	100	0.25	10440.8775	9326.8004	10050.4255	10391.1914	0.6746	0.0527	0.0475
1000, 100	norm-IW	100	0.5	10741.2805	9391.5584	10301.8075	10683.3390	0.7898	0.0634	0.0517
1000, 100	norm-IW	100	0.75	12062.7190	10741.6351	11677.5667	12048.2106	1.0214	0.0623	0.0541
1000, 100	norm-IW	10	0.25	117.6934	117.7080	109.5994	117.3095	0.6992	0.0604	0.0547
1000, 100	norm-IW	10	0.5	115.1880	111.4499	106.1806	115.1547	0.7127	0.0638	0.0527
1000, 100	norm-IW	10	0.75	120.9487	110.0401	110.6490	120.2324	0.3788	0.0370	0.0325

According to these experiments both subsampling-based solution and closed-form solution yielded good-enough results in a much shorter time, in comparison to CV.

B. Additional Proofs for Chapter 4

B.1 Proof of Lemma 1

Proof. (Lemma 1) Given a stream, let c_k^X be the true count for the k'th query at time t_k. Also, let $\tilde{C}_0^X = 0$ and $\tilde{C}_k^X$ be the reported noisy count for the k'th query at time t_k (capital letter is used to emphasize the randomness in $\tilde{C}_k^X$). Note that $\tilde{C}_k^X - \tilde{C}_{k-1}^X$, $k \geq 1$, are i.i.d. with $\tilde{C}_k^X - \tilde{C}_{k-1}^X - (c_k^X - c_{k-1}^X) \sim \text{Laplace}(1/\epsilon)$. For any $n > 0$, the joint density of $\tilde{C}_0^X, \ldots, \tilde{C}_n^X$ at the values $\tilde{c}_0, \ldots, \tilde{c}_n$ is

$$p_X(\tilde{c}_0, \ldots, \tilde{c}_n) = \prod_{k=1}^{n} \frac{\epsilon}{2} \exp\{-\epsilon |(\tilde{c}_k - \tilde{c}_{k-1}) - (c_k^X - c_{k-1}^X)|\}$$

For neighbour X and X', $(c_k^X - c_{k-1}^X)$ and $(c_k^{X'} - c_{k-1}^{X'})$ differ by 1 for only one $k \geq 1$. For that k. We have

$$|(\tilde{c}_k - \tilde{c}_{k-1}) - (c_k^X - c_{k-1}^X)| - |(\tilde{c}_k - \tilde{c}_{k-1}) - (c_k^{X'} - c_{k-1}^{X'})| \leq 1$$

As a result, $e^{-\epsilon} \leq p_X(\tilde{c}_0, \ldots, \tilde{c}_n)/p_{X'}(\tilde{c}_0, \ldots, \tilde{c}_n) \leq e^{\epsilon}$. □

B.1.1 Proof of Theorem 5

Theorem 5 is based on a standard result on the median of probabilistically bounded random variables, which is restated here with some adaptation to our setting.

Lemma 2. *Let $X_1, \ldots, X_d$ be random variables satisfying*

$$\mathbb{P}(|X_i - \mu| > \kappa) < \lambda_i, \quad i = 1, \ldots, d$$

and let $\lambda = \sum_{i=1}^{d} \lambda_i / d$. *If* $\lambda < 1/2$, *then,*

$$\begin{aligned}\mathbb{P}(|\tilde{X} - \mu| > \kappa) &\leq e^{-\frac{d}{2}(1-2\lambda)}[2(1-\lambda)]^{d/2} \\ &\leq \exp\left\{-\frac{d}{8}\frac{(1-2\lambda)^2}{1-\lambda}\right\}.\end{aligned}$$

where $\tilde{X}$ *is the median of* $X_1, \ldots, X_d$.

Proof. Let $Y_i = \mathbb{I}(|X_i - \mu| < \kappa)$ for $i = 1, \ldots, d$, and $Y = \sum_i Y_i$. Then $\mathbb{E}(Y_i) > 1 - \lambda_i$ and $\mathbb{E}(Y) = d(1-\lambda)$. The event $|\tilde{X} - \mu| > \kappa$ implies that at least $d/2$ of $X_1, \ldots, X_d$ are outside $(\mu - \kappa, \mu + \kappa)$, which is equivalent to $Y < d/2$. Also,

$$\begin{aligned}\{Y < d/2\} &\Leftrightarrow \left\{Y < \mathbb{E}(Y)\left[1 - \left(1 - \frac{d}{2\mathbb{E}(Y)}\right)\right]\right\} \\ &\Rightarrow \left\{Y < \mathbb{E}(Y)\left[1 - \left(1 - \frac{d}{2d(1-\lambda)}\right)\right]\right\} \\ &\Leftrightarrow \{Y < \mathbb{E}(Y)\,(1-\delta)\}\end{aligned}$$

where $\delta = \frac{1-2\lambda}{2(1-\lambda)}$. Therefore, by the Chernoff bound, we have

$$\begin{aligned}\mathbb{P}(|\tilde{X} - \mu| > \kappa) &\leq \left(\frac{e^{-\delta}}{(1-\delta)^{1-\delta}}\right)^{E(Y)} \\ &\leq \left(\frac{e^{-\delta}}{(1-\delta)^{1-\delta}}\right)^{d(1-\lambda)} = e^{-\frac{d}{2}(1-2\lambda)}[2(1-\lambda)]^{d/2}\end{aligned}$$

where the first line is by Chernoff, the second line is due to the fact that $\mathbb{E}(Y) > d(1-\lambda)$ and the base of the exponentiation is bounded by 1. □

We can use Lemma 2 to prove Theorem 5 for the use-and-keep method for dynamic queries in Algorithm 6.

Proof. (Theorem 5) In Algorithm 6, if the d cells corresponding to the queried element x are used $u_1 - 1, \ldots, u_d - 1$ times prior to the query, they will have been used $u_1, \ldots, u_d$ just before the response. Therefore, the values in the related cells can be written as $C(i, j_i^x) = C_0(i, j_i^x) + V_i$ for $i = 1, \ldots, d$, where $C_0(i, j_i^x)$ is the cell value we would have if the regular count sketch were used and $V_i = \sum_{k=1}^{u_i} V_{i,k}$ where each $V_{i,j}$ are independent with $V_{i,j} \sim \text{Laplace}(d/\epsilon)$. We have $\mathbb{E}(C(i, j_i^x)) = f_x$ and $\sigma_i^2 = \text{var}(C(i, j_i^x))$ is bounded by

$$\sigma_i^2 \leq \frac{||f||_2^2}{w} + \frac{u_i d^2}{\epsilon^2}.$$

where the first term is due to hashing. For $\kappa^2 > 2\max_i \sigma_i^2$, using the Chebyshev's bound, we have

$$\mathbb{P}(|\hat{C}_i - f_x| > \kappa) \leq \frac{\sigma_i^2}{\kappa^2}, \quad i = 1, \ldots, d.$$

By Lemma 2, an upper bound on the error of the median $\hat{f}_x^k = \text{median}(C(1, j_1^x), \ldots, C(1, j_d^x))$ is given by

$$\mathbb{P}(|\hat{f}_x - f_x| > \kappa) \leq e^{-\frac{d}{2}(1-2\lambda)}[2(1-\lambda)]^{d/2}$$

where $\lambda = \frac{1}{d}\sum_{i=1}^d \frac{\sigma_i^2}{\kappa^2} = \frac{||f||_2^2}{w\kappa^2} + \frac{d}{\epsilon^2\kappa^2}\sum_{i=1}^d u_i$ as claimed. □

B.1.2 Error bounds for output perturbation

Let f_x^c denote the response which is returned by standard count sketch for element x, and $\hat{f}_x^\epsilon$ be the value obtained by adding a Laplace noise to f_x^c to preserve ϵ privacy. Also, define $||f||_2^2 := \sum_x f_x^2$. The first result on the error is a bound for the probability of a fixed error.

Theorem 10. *Let $\kappa > 2||f||_2/w$ be a constant. Then, we have $\mathbb{P}(|\hat{f}_x^\epsilon - f_x| > \kappa) < \lambda$ where, with $\lambda^* = ||f_{-x}||_2/(w\kappa^*)$,*

$$\lambda = \min_{\frac{2||f||_2}{w} < \kappa^* < \kappa} e^{-\frac{d}{2}[(1-2\lambda^*) - \ln(2(1-\lambda^*))]} + e^{-\epsilon(\kappa - \kappa^*)}$$

Proof. For any $2||f_{-x}||_2/w < \kappa^* < \kappa$, the simple triangular inequality for probability statements can be applied as

$$\mathbb{P}(|\hat{f}_x^\epsilon - f_x| > \kappa) < \mathbb{P}(|\hat{f}_x - f_x| > \kappa - \kappa^*) + \mathbb{P}(|\hat{f}_x^\epsilon - \hat{f}_x| > \kappa^*).$$

While the first probability can be bounded using Lemma 2, the second one can be calculated exactly using the cdf of the Laplace distribution. This inequality holds for any $\kappa > \kappa^* > 2||f_{-x}||_2/w$, hence the minimum. □

Now, let us fix λ and examine how the amount of error varies with respect to d, with probability $1 - \lambda$.

Corollary 1. *For a noisy median response that satisfies ϵ_0-DP and $\lambda > (e/2)^{-d/2}$, we have $\mathbb{P}(|\hat{f}_x^\epsilon - f_x| > \kappa) < \lambda$ where*

$$\kappa = \max_{(e/2)^{-d/2} < \lambda^* < \lambda} \frac{||f||_2}{\sqrt{w\lambda_0}} - \frac{1}{\epsilon_0 \ln(\lambda - \lambda^*)}.$$

where λ_0 *is such that* $e^{-\frac{d}{2}[(1-2\lambda_0)-\ln(2(1-\lambda_0))} = \lambda^*$.

C. Proofs for Chapter 5

C.1 Proofs for the Proposed Mechanism

C.1.1 Proofs for LDP of RRRR

Proof of Theorem 6. Let $k = |S|$. We can write as

$$g_{S,\epsilon}(y|x) = \begin{cases} \frac{e^{\epsilon_1}}{e^{\epsilon_1}+k} & x \in S, y \in S, x = y \\ \frac{1}{e^{\epsilon_1}+k} & x \in S, y \in S, x \neq y \\ \frac{1}{K-k}\frac{1}{e^{\epsilon_1}+k} & x \in S, y \notin S \\ \frac{1}{e^{\epsilon_1}+k} & x \notin S, y \in S \\ \frac{e^{\epsilon_2}}{e^{\epsilon_2}+K-k-1}\frac{e^{\epsilon_1}}{e^{\epsilon_1}+k} & x \notin S, y \notin S, x = y \\ \frac{1}{e^{\epsilon_2}+K-k-1}\frac{e^{\epsilon_1}}{e^{\epsilon_1}+k} & x \notin S, y \notin S, x \neq y \end{cases}. \tag{C.1}$$

We will show that when ϵ_1, ϵ_2 are chosen according to the theorem,

$$e^{-\epsilon} \leq \frac{g_{S,\epsilon}(y|x)}{g_{S,\epsilon}(y|x')} \leq e^{\epsilon} \tag{C.2}$$

for all possible $x, x', y \in [K]$. When $S = \emptyset$, the proof is trivial; we focus on the non-trivial case $S \neq \emptyset$. For the non-trivial case, the transition probability $g_{S,\epsilon}(y|x)$ requires checking the ratio in (C.2) in 10 different cases for x, x', y concerning their interrelation.

(C1) $x \in S$, $x' \notin S$, $y \in S$, $y = x$. We have

$$\frac{g_{S,\epsilon}(y|x)}{g_{S,\epsilon}(y|x')} = \frac{\frac{e^{\epsilon_1}}{e^{\epsilon_1}+k}}{\frac{1}{e^{\epsilon_1}+k}} = e^{\epsilon_1}.$$

Since $\epsilon_1 \leq \epsilon$, (C.2) holds.

(C2) $x \in S$, $x' \notin S$, $y \in S$, $y \neq x$. We have

$$\frac{g_{S,\epsilon}(y|x)}{g_{S,\epsilon}(y|x')} = \frac{\frac{1}{e^{\epsilon_1}+k}}{\frac{1}{e^{\epsilon_1}+k}} = 1,$$

which trivially implies (C.2).

(C3) $x \in S$, $x' \notin S$, $y \notin S$, $y = x'$. We need

$$\frac{g_{S,\epsilon}(y|x)}{g_{S,\epsilon}(y|x')} = \frac{\frac{1}{K-k}\frac{1}{e^{\epsilon_1}+k}}{\frac{e^{\epsilon_2}}{e^{\epsilon_2}+K-k-1}\frac{e^{\epsilon_1}}{e^{\epsilon_1}+k}} = \frac{e^{\epsilon_2}+K-k-1}{(K-k)e^{\epsilon_1+\epsilon_2}}.$$

We can show that $\frac{g_{S,\epsilon}(y|x)}{g_{S,\epsilon}(y|x')} \leq 1 \leq e^{\epsilon}$ already holds since

$$\frac{e^{\epsilon_2}+K-k-1}{(K-k)e^{\epsilon_1+\epsilon_2}} = \frac{(K-k-1)+e^{\epsilon_2}}{(K-k-1)e^{\epsilon_1+\epsilon_2}+e^{\epsilon_1+\epsilon_2}},$$

and the first and the second terms in the numerator are smaller than those in the denominator, respectively. For the other side of the inequality,

$$\frac{e^{\epsilon_2}+K-k-1}{(K-k)e^{\epsilon_1+\epsilon_2}} \geq e^{-\epsilon}$$

requires

$$e^{\epsilon_2} \leq \frac{K-k-1}{e^{\epsilon_1-\epsilon}(K-k)-1}$$

whenever $e^{\epsilon_1-\epsilon}(K-k)-1 > 0$, which is the condition given in the theorem.

(C4) $x \in S$, $x' \notin S$, $y \notin S$, $y \neq x'$. We need

$$\frac{g_{S,\epsilon}(y|x)}{g_{S,\epsilon}(y|x')} = \frac{\frac{1}{K-k}\frac{1}{e^{\epsilon_1}+k}}{\frac{1}{e^{\epsilon_2}+K-k-1}\frac{e^{\epsilon_1}}{e^{\epsilon_1}+k}} = \frac{e^{\epsilon_2}+(K-k)-1}{(K-k)e^{\epsilon_1}}.$$

Since $\epsilon_2 \leq \epsilon$, we have

$$e^{\epsilon_2} \leq (K-k)(e^{\epsilon+\epsilon_1}-1)+1.$$

Hence

$$\frac{g_{S,\epsilon}(y|x)}{g_{S,\epsilon}(y|x')} \leq \frac{(K-k)(e^{\epsilon+\epsilon_1}-1)+1+K-k-1}{(K-k)e^{\epsilon_1}} \leq e^{\epsilon},$$

Hence, we proved the right-hand side inequality. For the left-hand side, we have

$$\frac{e^{\epsilon_2}+K-k-1}{(K-k)e^{\epsilon_1}} = \frac{(e^{\epsilon_2}-1)+K-k}{(K-k)e^{\epsilon_1}} \geq \frac{K-k}{(K-k)e^{\epsilon_1}} = e^{-\epsilon_1} \geq e^{-\epsilon}$$

since $\epsilon_2 \geq 0$ and $\epsilon_1 \leq \epsilon$.

(C5) $x, x' \in S$, $y \in S$, $y = x$. We have

$$\frac{g_{S,\epsilon}(y|x)}{g_{S,\epsilon}(y|x')} = \frac{e^{\epsilon_1}/(e^{\epsilon_1}+k)}{1/(e^{\epsilon_1}+k)} = e^{\epsilon_1}.$$

Since $\epsilon_1 \leq \epsilon$, (C.2) holds.

(C6) $x, x' \in S$, $y \in S$, $y \neq x$ and $y \neq x'$. We have

$$\frac{g_{S,\epsilon}(y|x)}{g_{S,\epsilon}(y|x')} = \frac{1/(e^{\epsilon_1}+k)}{1/(e^{\epsilon_1}+k)} = 1.$$

So (C.2) trivially holds.

(C7) $x, x' \in S$, $y \notin S$. We have

$$\frac{g_{S,\epsilon}(y|x)}{g_{S,\epsilon}(y|x')} = \frac{\frac{1}{K-k}\frac{1}{e^{\epsilon_1}+k}}{\frac{1}{K-k}\frac{1}{e^{\epsilon_1}+k}} = 1.$$

So, (C.2) trivially holds.

(C8) $x, x' \notin S$, $y \notin S$, $y = x$. We have

$$\frac{g_{S,\epsilon}(y|x)}{g_{S,\epsilon}(y|x')} = \frac{e^{\epsilon_2}/(e^{\epsilon_2}+K-k-1)e^{\epsilon_1}/(e^{\epsilon_1}+k)}{1/(e^{\epsilon_2}+K-k-1)e^{\epsilon_1}/(e^{\epsilon_1}+k)} = e^{\epsilon_2}.$$

Since $\epsilon_2 \leq \epsilon$, (C.2) holds.

(C9) $x, x' \notin S$, $y \notin S$, $y \neq x$, $y \neq x'$. We have

$$\frac{g_{S,\epsilon}(y|x)}{g_{S,\epsilon}(y|x')} = \frac{1/(e^{\epsilon_2}+K-k-1)e^{\epsilon_1}/(e^{\epsilon_1}+k)}{1/(e^{\epsilon_2}+K-k-1)e^{\epsilon_1}/(e^{\epsilon_1}+k)} = 1.$$

So, (C.2) trivially holds.

(**C10**) $x, x' \notin S$, $y \in S$. We have

$$\frac{g_{S,\epsilon}(y|x)}{g_{S,\epsilon}(y|x')} = \frac{1/(e^{\epsilon_1} + |S|)}{1/(e^{\epsilon_1} + |S|)} = 1.$$

So (C.2) trivially holds.

We conclude the proof by noting that any other case left out is symmetric in (x, x') to one of the covered cases and, therefore, does not need to be checked separately. □

C.1.2 Proofs about utility functions

Proof of Proposition 2. Given $\theta \in \Delta$, let ϑ be the $(K-1) \times 1$ column vector such that $\vartheta_i = \theta_i$ for $i = 1, \ldots, K-1$. We can write the Fisher information matrix in terms of the score vector as follows.

$$F(\theta; S, \epsilon) = \mathbb{E}_Y \left[\nabla_\vartheta \ln h_{S,\epsilon}(Y|\theta) \nabla_\vartheta \ln h_{S,\epsilon}(Y|\theta)^\top \right] = \sum_{y=1}^{K} h_{S,\epsilon}(y|\theta) \left[\nabla_\vartheta \ln h_{S,\epsilon}(y|\theta) \nabla_\vartheta \ln h_{S,\epsilon}(y|\theta)^\top \right].$$

Noting that

$$h_{S,\epsilon}(y|\theta) = \sum_{k=1}^{K-1} g_{S,\epsilon}(y|k)\vartheta_k + g_{S,\epsilon}(y|K) \left(1 - \sum_{k=1}^{K-1} \vartheta_k \right),$$

the score vector can be derived as

$$[\nabla_\vartheta \ln h_{S,\epsilon}(y|\theta)]_k = \frac{g_{S,\epsilon}(y|k) - g_{S,\epsilon}(y|K)}{h_{S,\epsilon}(y|\theta)}, \quad k = 1, \ldots, K-1. \tag{C.3}$$

As the $K \times (K-1)$ matrix $A_{S,\epsilon}$ defined as $A(i,j) = g(i|j) - g(i|K)$, we can rewrite (C.3) as $[\nabla_\vartheta \ln h_{S,\epsilon}(y|\theta)]_k = A_{S,\epsilon}(y,k)/h_{S,\epsilon}(y|\theta)$. Let a_y be the y'th row of $A_{S,\epsilon}$, and recall that D_θ is defined as a diagonal matrix with $1/h_{S,\epsilon}(j|\theta)$ being the j'th element in the diagonal. Then, the Fisher information matrix is

$$F(\theta; S, \epsilon) = \sum_{y=1}^{K} \frac{a_y^\top}{h_{S,\epsilon}(y|\theta)} \frac{a_y}{h_{S,\epsilon}(y|\theta)} h_{S,\epsilon}(y|\theta) = \sum_{y=1}^{K} a_y^\top \frac{1}{h_{S,\epsilon}(y|\theta)} a_y = A_{S,\epsilon}^\top D_\theta A_{S,\epsilon},$$

as claimed. □

Next, we prove that $F(\theta; S, \epsilon)$ is invertible. Let $G_{S,\epsilon}$ be the $K \times K$ matrix whose

elements are

$$G_{S,\epsilon}(i,j) = g_{S,\epsilon}(i|j), \quad i,j = 1,\dots,K. \tag{C.4}$$

To prove that $F(\theta; S, \epsilon)$ is invertible, we first prove the intermediate result that $G_{S,\epsilon}$ is invertible.

Lemma 3. *$G_{S,\epsilon}$ is invertible for all $S \subset [K]$ and $\epsilon > 0$.*

Proof. It suffices to prove that of $G_{S,\epsilon}$ is invertible for $S = \{1, 2, \dots, k\}$ and for all $k \in \{0, \dots, K-1\}$. For other S, $G_{S,\epsilon}$ can be obtained by permutation. Fix k and let $S = \{1, 2, \dots, k\}$. It can be verified by inspection that $G_{S,\epsilon}$ is a block matrix as

$$G_{S,\epsilon} = \begin{bmatrix} a_1 I_k + a_2 1_k 1_k^\top & b 1_k 1_{K-k}^\top \\ c 1_{K-k} 1_k^\top & d_1 I_{K-k} + d_2 1_{K-k} 1_{K-k}^\top \end{bmatrix},$$

where I_n is the identity matrix of size n and 1_n is the column vector of 1's of size n. The constants a_1, a_2, b, c, d_1, d_2 are given as

$$\begin{aligned} a_1 &= \frac{e^{\epsilon_1}}{k + e^{\epsilon_1}} - a_2, \quad a_2 = \frac{1}{k + e^{\epsilon_1}}, \quad b = \frac{1}{k + e^{\epsilon_1}}, \quad c = \frac{1}{K-k} a_2 \\ d_1 &= \frac{e^{\epsilon_2}}{e^{\epsilon_2} + K - k - 1} \frac{e^{\epsilon_1}}{k + e^{\epsilon_1}} - d_2, \quad d_2 = \frac{1}{e^{\epsilon_2} + K - k - 1} \frac{e^{\epsilon_1}}{k + \epsilon^1}. \end{aligned}$$

Also, note that since $\epsilon_1 > 0$ and $\epsilon_2 > 0$, a_1 and d_1 (whenever it is defined) are strictly positive.

The case $k = 0$ is trivial since then $G_{S,\epsilon} = d_1 I_K + d_2 1_K 1_K^\top$ is invertible. Hence, we focus on the case $0 < k < K$. For this case, firstly, note that the matrices on the diagonal are invertible. So, by Weinstein–Aronszajn identity, for $G_{S,\epsilon}$ to be invertible, it suffices to show that the matrix

$$M = a_1 I_k + a_2 1_k 1_k^\top - b 1_k 1_{K-k}^\top (d_1 I_{K-k} + d_2 1_{K-k} 1_{K-k}^\top)^{-1} c 1_{K-k} 1_k^\top$$

is invertible. Using the Woodbury matrix identity, the matrix M can be expanded as

$$\begin{aligned} M &= a_1 I_k + a_2 1_k 1_k^\top - b 1_k 1_{K-k}^\top \left(\frac{I_{K-k}}{d_1} - \frac{1}{d_1} 1_{K-k} \left(\frac{1}{d_2} + 1_{K-k}^\top \frac{1}{d_1} 1_{K-k} \right)^{-1} 1_{K-k}^\top \frac{1}{d_1} \right) c 1_{K-k} 1_k^\top \\ &= a_1 I_k + a_2 1_k 1_k^\top - \frac{bc}{d_1} 1_k 1_{K-k}^\top 1_{K-k} 1_k^\top + \frac{bc}{d_1^2} \left(\frac{1}{d_2} + \frac{K-k}{d_1} \right)^{-1} 1_k 1_{K-k}^\top 1_{K-k} 1_{K-k}^\top 1_{K-k} 1_k^\top \\ &= a_1 I_k + \left[a_2 - \frac{(K-k)bc}{d_1} + \frac{bc}{d_1^2} \left(\frac{1}{d_2} + \frac{K(K-k)}{d_1} \right)^{-1} \right] 1_k 1_k^\top. \end{aligned}$$

Inside the square brackets is a scalar, therefore, M in question is the sum of an identity matrix and a rank-1 matrix, which is invertible. Hence, $G_{S,\epsilon}$ is invertible. □

Proof of Proposition 3. Note that $A_{S,\epsilon} = G_{S,\epsilon}J$, where the $K \times (K-1)$ matrix J satisfies $J(i,i) = 1$ and $J(i,K) = -1$ for $i = 1,\ldots,K$, and $J(i,j) = 0$ otherwise. Since $G_{S,\epsilon}$ is invertible, it is full rank. Also, the columns of $A_{S,\epsilon}$, denoted by c_i^A, $i = 1,\ldots,K-1$ are given by

$$c_1^A = c_1^G - c_K^G, \quad \ldots, \quad c_{K-1}^A = c_{K-1}^G - c_K^G,$$

where c_i^G is the i'th column of $G_{S,\epsilon}$ for $i = 1,\ldots,K$. Observe that c_i^A, $i = 1,\ldots,K-1$ are linearly independent since any linear combination of those columns is in the form of

$$\sum_{i=1}^{K-1} a_i c_i^A = \sum_{i=1}^{K-1} a_i c_i^G - \left(\sum_{i=1}^{K-1} a_i\right) c_K^G.$$

Since the columns of $G_{S,\epsilon}$ are linearly independent, the linear combination above becomes 0 only if $a_1 = \ldots = a_{K-1} = 0$. This shows that the columns of $A_{S,\epsilon}$ are also linearly independent. Thus, we conclude that $A_{S,\epsilon}$ has rank $K-1$. Finally, since D_θ is diagonal with positive diagonal entries, $A_{S,\epsilon}^\top D_\theta A_{S,\epsilon} = A_{S,\epsilon}^\top D_\theta^{1/2} D_\theta^{1/2} A_{S,\epsilon}$ is positive definite, hence invertible. □

The following proof contains a derivation of the utility function based on the MSE of the Bayesian estimator of X given Y.

Proof of Proposition 4. It is well-known that the expectation in (5.10) is minimized when $\widehat{e_X} = \nu(Y) := \mathbb{E}_\theta[e_X|Y]$, i.e. the posterior expectation of e_X given Y. That is,

$$\min_{\widehat{e_X}} \mathbb{E}_\theta\left[\|e_X - \widehat{e_X}(Y)\|^2\right] = \mathbb{E}_\theta\left[\|e_X - \nu(Y)\|^2\right].$$

For the squared norm inside the expectation, we have

$$\begin{aligned}\|e_X - \nu(Y)\|^2 &= (1 - v(Y)_X)^2 + \sum_{k\neq X} v(Y)_k^2 \\ &= 1 + \nu(Y)_X^2 - 2\nu(Y)_X + \sum_{k\neq X} v(Y)_k^2 \\ &= 1 - 2\nu(Y)_X + \sum_{k=1}^{K} v(Y)_k^2. \qquad \text{(C.5)}\end{aligned}$$

The expectation of the last term in (C.5) is

$$\begin{aligned}\mathbb{E}_\theta\left[\sum_{k=1}^{K} v(Y)_k^2\right] &= \sum_y h_{S,\epsilon}(y|\theta)\sum_{x=1}^{K} p_{S,\epsilon}(x|y,\theta)^2 \\ &= \sum_{y=1}^{K}\sum_{x=1}^{K} h_{S,\epsilon}(y|\theta)p_{S,\epsilon}(x|y,\theta)^2 \\ &= \sum_{y=1}^{K}\sum_{x=1}^{K} \frac{g_{S,\epsilon}(y|x)^2\theta_x^2}{h_{S,\epsilon}(y|\theta)}. \end{aligned} \tag{C.6}$$

For the expectation of the second term in (C.5), we have

$$\mathbb{E}_\theta\left[\nu(Y)_X\right] = \sum_{x,y} p_{S,\epsilon}(x|y,\theta)p_{S,\epsilon}(x,y|\theta),$$

where $p(x,y|\theta)$ denotes the joint probability of X,Y given θ. Substituting $p(x,y|\theta) = p_{S,\epsilon}(x|y,\theta)h_{S,\epsilon}(y|\theta)$ into the equation above, we get

$$\begin{aligned}\mathbb{E}_\theta\left[\nu(Y)_X\right] &= \sum_{x=1}^{K}\sum_{y=1}^{K} h_{S,\epsilon}(y|\theta)p_{S,\epsilon}(x|y,\theta)^2. \\ &= \sum_{x=1}^{K}\sum_{y=1}^{K} \frac{g_{S,\epsilon}(y|x)^2\theta_x^2}{h_{S,\epsilon}(y|\theta)}, \end{aligned} \tag{C.7}$$

which is equal to what we get in (C.6). Substituting (C.6) and (C.7) into (C.5), we obtain

$$\begin{aligned}\mathbb{E}_\theta\left[\|e_X - \nu(Y)\|^2\right] &= 1 - 2\sum_{x=1}^{K}\sum_{y=1}^{K} \frac{g_{S,\epsilon}(y|x)^2\theta_x^2}{h_{S,\epsilon}(y|\theta)} + \sum_{x=1}^{K}\sum_{y=1}^{K} \frac{g_{S,\epsilon}(y|x)^2\theta_x^2}{h_{S,\epsilon}(y|\theta)} \\ &= 1 - \sum_{x=1}^{K}\sum_{y=1}^{K} \frac{g_{S,\epsilon}(y|x)^2\theta_x^2}{h_{S,\epsilon}(y|\theta)}. \end{aligned}$$

Finally, using the definition $U_5(\theta,S,\epsilon) = -\min_{\widehat{e_X}} \mathbb{E}_\theta\left[\|e_X - \widehat{e_X}(Y)\|^2\right]$, we conclude the proof. □

Proof of Theorem 7. The global maximization of U_6 over the set of all the subsets $S \subset [K]$ can be decomposed as

$$\max_{S\subset[K]} U_6(\theta,S,\epsilon) = \max_{k\in\{0,\ldots,K-1\}} \left\{ \max_{S\subset[K]:|S|=k} U_6(\theta,S,\epsilon) \right\}. \tag{C.8}$$

This inner maximization is equivalent to fixing the cardinality of S to k and finding

the best S with cardinality k. Now, the utility function can be written as

$$\begin{aligned} U_6(\theta, S, \epsilon) &= \frac{e^{\epsilon_1}}{e^{\epsilon_1}+k}\left(\sum_{i\in S}\theta_i + \frac{e^{\epsilon_2}}{e^{\epsilon_2}+K-k-1}\sum_{i\notin S}\theta_i\right) \\ &= \left(\frac{e^{\epsilon_1}}{e^{\epsilon_1}+k}\right)\sum_{i\in S}\theta_i + \left(\frac{e^{\epsilon_1}}{e^{\epsilon_1}+k}\right)\left(\frac{e^{\epsilon_2}}{e^{\epsilon_2}+K-k-1}\right)\sum_{i\notin S}\theta_i, \end{aligned}$$

where k appears in the first line since $|S| = k$. Note that $\sum_{i\in S}\theta_i$ and $\sum_{i\notin S}\theta_i$ sum to 1 and the constants in front of the first sum is larger than that of the second. Hence, we seek to maximize an expression in the form of

$$ax + b(1-x)$$

over a variable $x > 0$ when $a > b > 0$. This is maximized when $x > 0$ is taken as large as possible. Therefore, $U_6(\theta, S, \epsilon)$ is maximized when $\sum_{i\in S}\theta_i$ is made as large as possible under the constraint that $|S| = k$. Under this constraint this sum is maximized when S has the indices of the k largest components of θ, that is, when $S = S_{k,\theta} = \{\sigma_\theta(1), \ldots, \sigma_\theta(k)\}$.

Then, (C.8) reduces to $\max_{k=1,\ldots,K} U_6(\theta, S_{k,\theta}, \epsilon)$. Hence, we conclude. □

C.2 Proof for SGLD update

Proof of Proposition 5. For the *prior* component of the gradient, recall that we have $\phi = (\phi_1, \ldots, \phi_K)$, where

$$\phi_i \overset{\text{iid}}{\sim} \text{Gamma}(\rho_i, 1), \quad i = 1, \ldots, K.$$

Then, the marginal pdf of ϕ_i satisfies

$$\ln p(\phi_i) = (\rho_i - 1)\ln\phi_i - \ln\Gamma(\rho_i) - \phi_i, \quad i = 1, \ldots, K.$$

Taking the partial derivatives of $\ln p(\phi_i)$ with respect to ϕ_i, we have

$$[\nabla_\phi \ln p(\phi)]_i = \frac{\rho_i - 1}{\phi_i} - 1, \quad i = 1, \ldots, K.$$

For the *likelihood* component, given $\theta \in \Delta$, let the $(K-1)\times 1$ vector ϑ be the reparametrization of θ such that $\vartheta_i = \theta_i$ for $i = 1, \ldots, K-1$. Then, according to (5.13),

$$\vartheta_k = \frac{\phi_k}{\sum_{j=1}^K \phi_j}, \quad k = 1, \ldots, K-1. \tag{C.9}$$

Using the chain rule, we can write the gradient of the log-likelihood with respect to ϕ as

$$\nabla_\phi \ln p(y|\phi) = J \cdot \nabla_\vartheta \ln h_{S,\epsilon}(y|\theta).$$

where J is the $K \times (K-1)$ Jacobian matrix for the mapping from ϕ to ϑ in (C.9), whose (i,j)th element can be derived as

$$\begin{aligned} J(i,j) = \frac{\partial \vartheta_j}{\partial \phi_i} &= \frac{\partial}{\partial \phi_i} \frac{\phi_j}{\sum_{k=1}^K \phi_k} \\ &= \mathbb{I}(i=j) \frac{1}{\sum_{k=1}^K \phi_k} - \frac{\phi_j}{\left(\sum_{k=1}^K \phi_k\right)^2}. \end{aligned}$$

Using (C.3) for $\nabla_\vartheta \ln h_{S,\epsilon}(y|\theta)$, we complete the proof. □

C.3 Proofs for convergence and consistency results

C.3.1 Preliminary results

Lemma 4. *Given $\epsilon \geq 0$, there exists constants $0 < c \leq C < \infty$ such that for all $\theta \in \Delta$ and all $S \subset [K]$, we have*

$$c \leq g_{S,\epsilon}(y|x) \leq C, \quad c \leq h_{S,\epsilon}(y|\theta) \leq C.$$

Proof. The bounds for $g_{S,\epsilon}(y|x)$ can directly be verified from (C.1). Moreover,

$$c \leq \min_{i=1,\ldots,K} g_{S,\epsilon}(y|i) \leq h_{S,\epsilon}(y|\theta) = \sum_{i=1}^K g_{S,\epsilon}(y|i)\theta_i \leq \max_{i=1,\ldots,K} g_{S,\epsilon}(y|i) \leq C.$$

Hence, we conclude. □

Remark 1. *For the symbol θ, which is used for $K \times 1$ probability vectors in Δ, we will associate the symbol ϑ such that ϑ denotes the shortened vector of the first $K-1$ elements of θ. Accordingly, we will use ϑ, ϑ', ϑ^*, etc, to denote the shortened versions of θ, θ', θ^*, etc.*

Lemma 5. *For any $\theta, \theta' \in \Delta$, $y \in [K]$, and $S \subset [K]$, we have*

$$\nabla_\vartheta \ln h_{S,\epsilon}(y|\theta)^\top (\vartheta - \vartheta') = \frac{h_{S,\epsilon}(y|\theta) - h_{S,\epsilon}(y|\theta')}{h_{S,\epsilon}(y|\theta)}. \tag{C.10}$$

Proof. Recall from (C.3) that

$$[\nabla_\vartheta \ln h_{S,\epsilon}(y|\theta)]_i = \frac{g_{S,\epsilon}(y|i) - g_{S,\epsilon}(y|K)}{h_{S,\epsilon}(y|\theta)}, \quad i = 1, \ldots, K-1.$$

Hence,

$$\begin{aligned}
\nabla_\vartheta \ln h_{S,\epsilon}(y|\theta)^\top \vartheta &= \frac{1}{h_{S,\epsilon}(y|\theta)} \sum_{i=1}^{K-1} (g_{S,\epsilon}(y|i) - g_{S,\epsilon}(y|K))\vartheta_i \\
&= \frac{1}{h_{S,\epsilon}(y|\theta)} \left[\sum_{i=1}^{K-1} g_{S,\epsilon}(y|i)\vartheta_i - g_{S,\epsilon}(y|K) \sum_{i=1}^{K-1} \vartheta_i \right] \\
&= \frac{1}{h_{S,\epsilon}(y|\theta)} \left[\sum_{i=1}^{K-1} g_{S,\epsilon}(y,i)\theta_i - g_{S,\epsilon}(y|K)(1-\theta_K) \right] \\
&= \frac{1}{h_{S,\epsilon}(y|\theta)} \left[h_{S,\epsilon}(y|\theta) - g_{S,\epsilon}(y|K) \right],
\end{aligned}$$

and likewise $\nabla_\vartheta \ln h_{S,\epsilon}(y|\theta)^\top \vartheta' = \frac{1}{h_{S,\epsilon}(y|\theta)} \left[h_{S,\epsilon}(y|\theta') - g_{S,\epsilon}(y|K) \right]$. Taking the difference between $\nabla_\vartheta \ln h_{S,\epsilon}(y|\theta)^\top \vartheta$ and $\nabla_\vartheta \ln h_{S,\epsilon}(y|\theta)^\top \vartheta'$, we arrive at the result. □

Concavity of $\ln h_{S,\epsilon}(y|\theta)$: The following lemmas help with proving the concavity of $h_{S,\epsilon}(y|\theta)$ as a function of θ.

Lemma 6. *For $0 < b \le a \le 1$, we have $\ln \frac{a}{b} \ge \frac{a-b}{a} + \frac{(a-b)^2}{2}$.*

Proof. For $z > 0$, using the series based on the inverse hyperbolic tangent function, we can write

$$\ln z = 2 \sum_{k=0}^{\infty} \frac{1}{2k+1} \left(\frac{z-1}{z+1} \right)^{2k+1}.$$

Apply the expansion to $z = a/b$ when $a \ge b$. Noting that $(z-1)/(z+1) = (a-b)/(a+b)$,

$$\ln \frac{a}{b} \ge 2 \frac{a-b}{a+b} \ge \frac{2(a-b)}{2a} = \frac{a-b}{a}$$

where the difference is

$$\frac{2(a-b)}{a+b} - \frac{a-b}{a} = \frac{2a^2-2ab-a^2+b^2}{a(a+b)} = \frac{(a-b)^2}{(a+b)a} \geq \frac{(a-b)^2}{2}$$

since $0 < a, b \leq 1$. □

Lemma 7. *Let $0 < \alpha \leq 1$. For $x \geq 1$, we have $\frac{1}{\alpha}(x^\alpha - 1) \leq (x-1)$.*

Proof. Consider $f(x) = \frac{1}{\alpha}(x^\alpha - 1) - (x-1)$. We have $f(1) = 0$ and $f'(x) = x^{\alpha-1} - 1 \leq 0$ for $x \geq 1$. Hence, we conclude. □

Lemma 8. *Let $0 < a \leq b \leq 1$ and $0 < \alpha \leq 1$. Then, $b^\alpha - a^\alpha \geq \alpha(b-a)$.*

Proof. Fix a and let $b = a + x$ for $0 \leq x \leq 1-a$. Consider the function $f(x) = (a+x)^\alpha - a^\alpha - \alpha x$. We have $f(0) = 0$ and $f'(x) = \alpha(a+x)^{\alpha-1} - \alpha \geq 0$ over $0 \leq x \leq (1-a)$ since $a + x \leq 1$ and $\alpha - 1 \leq 0$. Hence, we conclude. □

Lemma 9. *Given $\epsilon > 0$, there exists $m_0 > 0$ such that, for all $S \subset [K]$ and $y \in [K]$, $\ln h_{S,\epsilon}(y|\theta)$ is a concave function of ϑ that satisfies*

$$\ln h_{S,\epsilon}(y|\theta) - \ln h_{S,\epsilon}(y|\theta') \geq \nabla_\vartheta \ln h_{S,\epsilon}(y|\theta)^\top (\vartheta - \vartheta') + m_0 (h_{S,\epsilon}(y|\theta) - h_{S,\epsilon}(y|\theta'))^2$$

for all $\theta, \theta' \in \Delta$.

Proof. We will look at the cases $h_{S,\epsilon}(y|\theta) \geq h_{S,\epsilon}(y|\theta')$ and $h_{S,\epsilon}(y|\theta) \leq h_{S,\epsilon}(y|\theta')$ separately.

3.1 Assume $h_{S,\epsilon}(y|\theta) \geq h_{S,\epsilon}(y|\theta')$. Using Lemma 6, we have

$$\begin{aligned}\ln h_{S,\epsilon}(y|\theta) - \ln h_{S,\epsilon}(y|\theta') &\geq \frac{h_{S,\epsilon}(y|\theta) - h_{S,\epsilon}(y|\theta')}{h_{S,\epsilon}(y|\theta)} + \frac{1}{2}(h_{S,\epsilon}(y|\theta) - h_{S,\epsilon}(y|\theta'))^2 \\ &= \nabla_\vartheta \ln h_{S,\epsilon}(y|\theta)^\top (\vartheta - \vartheta') + \frac{1}{2}(h_{S,\epsilon}(y|\theta) - h_{S,\epsilon}(y|\theta'))^2,\end{aligned}$$

where the last line follows from Lemma 5.

3.2 Assume $h_{S,\epsilon}(y|\theta) \leq h_{S,\epsilon}(y|\theta')$. Let $a = h_{S,\epsilon}(y|\theta)$ and $b = h_{S,\epsilon}(y|\theta')$. Let

$$\alpha = \min\left\{1, \frac{\ln 2}{\ln(C/c)}\right\},$$

where c and C are given in Lemma 4. This α ensures that $0 < \alpha \leq 1$ and

$b^\alpha/a^\alpha \leq 2$, so that we can use Taylor's expansion of $\ln \frac{b^\alpha}{a^\alpha}$ around 1 and have

$$\ln \frac{b}{a} = \frac{1}{\alpha} \ln \frac{b^\alpha}{a^\alpha} = \frac{1}{\alpha} \left[\sum_{k=1}^{\infty} (-1)^{k+1} \frac{1}{k} \left(\frac{b^\alpha}{a^\alpha} - 1 \right)^k \right].$$

Approximating the expansion up to its third term, we have the following upper bound on $\ln \frac{b}{a}$ as

$$\ln \frac{b}{a} \leq \frac{1}{\alpha} \left(\frac{b^\alpha}{a^\alpha} - 1 \right) - \frac{1}{2\alpha} \left(\frac{b^\alpha}{a^\alpha} - 1 \right)^2 + \frac{1}{3\alpha} \left(\frac{b^\alpha}{a^\alpha} - 1 \right)^3. \tag{C.11}$$

For the third term, we have

$$\frac{1}{3\alpha} \left(\frac{b^\alpha}{a^\alpha} - 1 \right)^3 = \frac{1}{3\alpha} \left(\frac{b^\alpha}{a^\alpha} - 1 \right)^2 \left(\frac{b^\alpha}{a^\alpha} - 1 \right) \leq \frac{1}{3\alpha} \left(\frac{b^\alpha}{a^\alpha} - 1 \right)^2,$$

since $1 \leq \frac{b^\alpha}{a^\alpha} \leq 2$. Substituting this into (C.11), the inequality can be continued as

$$\ln \frac{b}{a} \leq \frac{1}{\alpha} \left(\frac{b^\alpha}{a^\alpha} - 1 \right) - \frac{1}{6\alpha} \left(\frac{b^\alpha}{a^\alpha} - 1 \right)^2. \tag{C.12}$$

Using Lemma 7 to bound the first term in (C.12), we have

$$\ln \frac{b}{a} \leq \left(\frac{b}{a} - 1 \right) - \frac{1}{6\alpha} \left(\frac{b^\alpha}{a^\alpha} - 1 \right)^2. \tag{C.13}$$

Finally, using Lemma 8 we can lower-bound the second term in (C.12) as

$$\frac{b^\alpha}{a^\alpha} - 1 = \frac{b^\alpha - a^\alpha}{a^\alpha} \geq \frac{\alpha(b-a)}{a^\alpha} \geq \alpha(b-a),$$

where the last inequality follows from $a \leq 1$ and $\alpha > 0$. We end up with

$$\ln \frac{b}{a} \leq \left(\frac{b}{a} - 1 \right) - \frac{\alpha}{6} (b-a)^2.$$

Referring to the definitions of a and b, we have

$$\ln h_{S,\epsilon}(y|\theta') - \ln h_{S,\epsilon}(y|\theta) \leq \left(\frac{h_{S,\epsilon}(y|\theta')}{h_{S,\epsilon}(y|\theta)} - 1 \right) - \frac{\alpha}{6} (h_{S,\epsilon}(y|\theta) - h_{S,\epsilon}(y|\theta'))^2,$$

or, reversing the inequality,

(C.14)
$$\ln h_{S,\epsilon}(y|\theta) - \ln h_{S,\epsilon}(y|\theta') \geq \left(1 - \frac{h_{S,\epsilon}(y|\theta')}{h_{S,\epsilon}(y|\theta)}\right) + \frac{\alpha}{6}(h_{S,\epsilon}(y|\theta) - h_{S,\epsilon}(y|\theta'))^2.$$

Using (C.10) in Lemma 5, we rewrite (C.14) as

(C.15)
$$\ln h_{S,\epsilon}(y|\theta) - \ln h_{S,\epsilon}(y|\theta') \geq \nabla_\vartheta \ln h_{S,\epsilon}(y|\theta)^\top (\vartheta - \vartheta') + \frac{\alpha}{6}(h_{S,\epsilon}(y|\theta) - h_{S,\epsilon}(y|\theta'))^2,$$

which is the inequality we look for.

To cover both cases, take $m_0 = \min\{\frac{\alpha}{6}, \frac{1}{2}\}$. So, the proof is complete. □

Recalling that S_t is the selected subset at time t, define

(C.16)
$$V_t(\theta, \theta') := (h_{S_t,\epsilon}(Y_i|\theta) - h_{S_t,\epsilon}(Y_t|\theta'))^2.$$

The proof of Theorem 8 requires a probabilistic bound for $\sum_{t=1}^n V_t(\theta, \theta')$, which we provide next.

Lemma 10. *For all $\theta, \theta' \in \Delta$ and $t \geq 0$, $V_t(\theta, \theta') \leq \|\theta - \theta'\|^2$.*

Proof. Let the $1 \times K$ vector r_i be the ith row of $G_{S_t,\epsilon}$. Then, we obtain

$$\begin{aligned} V_t(\theta, \theta') = [h_{S_t,\epsilon}(y|\theta) - h_{S_t,\epsilon}(y|\theta')]^2 &= (r_i(\theta - \theta'))^2 \\ &= (\theta - \theta')^\top r_i^\top r_i (\theta - \theta') \\ &\leq \|\theta - \theta'\|^2, \end{aligned}$$

since every element of r_i is at most 1. □

Lemma 11. *For all $\theta, \theta' \in \Delta$ and $t \geq 1$, there exists a constant $c_u > 0$ such that*

$$E_{\theta^*}[V_t(\theta, \theta')] \geq c_u \|\theta - \theta'\|^2,$$

where E_{θ^*} is the expectation operator with respect to P_{θ^*} defined in (5.18)*, $\lambda_{\min}(A)$ is the minimum eigenvalue of the square matrix A and the matrix $G_{S,\epsilon}$ is defined in* (C.4)*.*

Proof. The overall expectation can be written as

$$E_{\theta^*}[V_t(\theta, \theta')] = \sum_{S \subset [K]} P_{\theta^*}(S_t = S) E_{\theta^*}[V_t(\theta, \theta') | S_t = S]$$

where the conditional expectation can be bounded as

$$E_{\theta^*}[V_t(\theta,\theta')|S_t = S] = \sum_{i=1}^{K}(h_{S,\epsilon}(i|\theta) - h_{S,\epsilon}(i|\theta'))^2 h_{S,\epsilon}(i|\theta^*)$$
$$\geq \sum_{i=1}^{K}(h_{S,\epsilon}(i|\theta) - h_{S,\epsilon}(i|\theta'))^2 c, \tag{C.17}$$

where the second line is due to Lemma 4. Further, let the $1 \times K$ vector r_i be the ith row of $G_{S,\epsilon}$. Then,

$$\begin{aligned}\sum_{i=1}^{K}(h_{S,\epsilon}(i|\theta) - h_{S,\epsilon}(i|\theta'))^2 &= \sum_{i=1}^{K}(r_i(\theta-\theta'))^2 \\ &= \sum_{i=1}^{K}(\theta-\theta')^\top r_i^\top r_i(\theta-\theta') \\ &= (\theta-\theta')^\top G_{S,\epsilon}^\top G_{S,\epsilon}(\theta-\theta') \\ &= \|G_{S,\epsilon}(\theta-\theta')\|^2 \\ &\geq \lambda_{\min}(G_{S,\epsilon}^\top G_{S,\epsilon})\|\theta-\theta'\|^2. \end{aligned} \tag{C.18}$$

Combining (C.17) and (C.18) and letting $c_u := c\min_{S\subset[K]}\lambda_{\min}(G_{S,\epsilon}^\top G_{S,\epsilon})$, we have

$$E_{\theta^*}[V_t(\theta,\theta')|S_t = S] \geq c_u\|\theta-\theta'\|^2. \tag{C.19}$$

for all S, which directly implies $E_{\theta^*}[V_t(\theta,\theta')] \geq c_u\|\theta-\theta'\|^2$ for the overall expectation. Finally, $c_u > 0$ since, by Lemma 3, every $G_{S,\epsilon}$ is invertible. □

Further, define

$$W_t(\theta,\theta') := 1 - \frac{V_t(\theta,\theta')}{\|\theta-\theta'\|^2}.$$

Lemma 12. *For any $0 < t_1 < \ldots < t_k$, we have $E_{\theta^*}\left[\prod_{i=1}^{k} W_{t_i}(\theta,\theta')\right] \leq (1-c_u)^k$.*

Proof. For simplicity, we drop (θ,θ') from the notation and denote the random variables in question as $W_{t_1},\ldots,W_{t_k}$. We can write

$$\begin{aligned}E_{\theta^*}\left(\prod_{i=1}^{k} W_{t_i}\right) &= E_{\theta^*}\left[E_{\theta^*}\left(\prod_{i=1}^{k} W_{t_i}\middle| W_{t_1},\ldots,W_{t_{k-1}}\right)\right] \\ &= E_{\theta^*}\left[\left(\prod_{i=1}^{k-1} W_{t_i}\right) E_{\theta^*}\left(W_{t_k}|W_{t_1},\ldots,W_{t_{k-1}}\right)\right]. \end{aligned} \tag{C.20}$$

By construction of $W_i(\theta,\theta')$, we have $E_{\theta^*}(W_i|S_i = S) \leq 1 - c_u$, which follows from

(C.19). Using this, the inner conditional expectation can be bounded as

$$E_{\theta^*}\left(W_{t_k}|W_{t_1},\ldots,W_{t_{k-1}}\right) = \sum_{S\subset[K]} P_{\theta^*}(S_{t_k}=S|W_{t_1},\ldots,W_{t_{k-1}})E_{\theta^*}(W_{t_k}|S_{t_k}=S)$$
$$\leq \sum_{S\subset[K]} P_{\theta^*}(S_{t_k}=S|W_{t_1},\ldots,W_{t_{k-1}})\,(1-c_u)$$
$$= (1-c_u)\,. \tag{C.21}$$

Combining (C.20) and (C.21), we have

$$E_{\theta^*}\left(\prod_{i=1}^{k} W_{t_i}\right) \leq (1-c_u)\,E_{\theta^*}\left(\prod_{i=1}^{k-1} W_{t_i}\right). \tag{C.22}$$

By Lemmas 10 and 11, we have $V_t(\theta,\theta') \leq \|\theta-\theta'\|^2$ and $E_{\theta^*}[V_t(\theta,\theta')] \geq c_u\|\theta-\theta'\|^2$. Thus, we necessarily have $c_u < 1$. Therefore, the recursion in (C.22) can be used until $k=1$ to obtain the desired result. □

Note that $W_t(\theta,\theta')$ is bounded as $0 \leq W_i(\theta,\theta') \leq 1$. We now quote a critical theorem from Pelekis & Ramon (2017, Theorem 3.2) regarding the sum of dependent and bounded random variables, which will be useful for bounding $\sum_{t=1}^{n} V_t(\theta,\theta')$.

Theorem 11. *(Pelekis & Ramon, 2017, Theorem 3.2) Let $W_1,\ldots,W_n$ be random variables, such that $0 \leq W_t \leq 1$, for $t=1,\ldots,n$. Fix a real number $\tau \in (0,n)$ and let k be any positive integer, such that $0 < k < \tau$. Then*

$$\mathbb{P}\left(\sum_{t=1}^{n} W_t \geq \tau\right) \leq \frac{1}{\binom{\tau}{k}} \sum_{A\subseteq\{1,\ldots,n\}:|A|=k} \mathbb{E}\left[\prod_{i\in A} W_i\right], \tag{C.23}$$

where $\binom{\tau}{k} = \frac{\tau(\tau-1)\ldots(\tau-k+1)}{k!}$.

In the following, we apply Theorem 11 for $\sum_{t=1}^{n} V_t(\theta,\theta')$.

Lemma 13. *For every $\theta,\theta' \in \Delta$ and $a \in (0,1)$,*

$$\lim_{n\to\infty} P_{\theta^*}\left(\frac{1}{n}\sum_{t=1}^{n} V_t(\theta,\theta') \leq ac_u\|\theta-\theta'\|^2\right) = 0.$$

Proof. For any integer $k < n(1-ac_u) < n$, we have

$$\begin{aligned}
P_{\theta^*}\left(\frac{1}{n}\sum_{t=1}^{n} V_t(\theta,\theta') \le ac_u\|\theta-\theta'\|^2\right) &= P_{\theta^*}\left(\|\theta-\theta'\|^2\frac{1}{n}\sum_{t=1}^{n}(1-W_t(\theta,\theta')) \le ac_u\|\theta-\theta'\|^2\right)\\
&= P_{\theta^*}\left(\frac{1}{n}\sum_{t=1}^{n}(1-W_t(\theta,\theta')) \le ac_u\right)\\
&= P_{\theta^*}\left(\sum_{t=1}^{n} W_t(\theta,\theta') \ge n\,(1-ac_u)\right)\\
&= P_{\theta^*}\left(\sum_{t=1}^{n} W_t(\theta,\theta') \ge n\,(1-ac_u)\right)\\
&\le \frac{1}{\binom{n(1-ac_u)}{k}}\sum_{A\subseteq\{1,\ldots,n\}:|A|=k} E_{\theta^*}\left[\prod_{i\in A} W_i(\theta,\theta')\right]\\
&\le \frac{1}{\binom{n(1-ac_u)}{k}}\binom{n}{k}(1-c_u)^k,
\end{aligned}$$

where the last two lines follow from Theorem 11 and Lemma 12, respectively. Select $k = k^* = \lceil n(1-a)\rceil$ and note that when $n > 1/(a-ac_u)$ one always has $k^* < n(1-ac_u)$. Then, for $n > 1/(a-ac_u)$,

$$\begin{aligned}
P_{\theta^*}\left(\frac{1}{n}\sum_{t=1}^{n} V_t(\theta,\theta') \le ac_u\|\theta-\theta'\|^2\right) &\le \frac{1}{\binom{n(1-ac_u)}{k^*}}\binom{n}{k^*}(1-c_u)^{k^*}\\
&= (1-c_u)^{k^*}\prod_{i=1}^{k^*}\frac{n-i+1}{n(1-ac_u)-i+1}. \qquad \text{(C.24)}
\end{aligned}$$

The right-hand side does not depend on θ,θ' and converges to 0. □

Smoothness of $\ln h_{S,\epsilon}(y|\theta)$: Next, we establish the L-smoothness of $h_{S,\epsilon}(y|\theta)$ as a function of θ for any $y \in [K]$ and $S \subset [K]$. Some technical lemmas are needed first.

Lemma 14. *For $x > 0$, $\ln(1+x) \ge x - \frac{1}{2}x^2$.*

Proof. The function $\ln(1+x) - x + 0.5x^2$ is 0 at $x = 0$ and its derivative $1/(1+x) - 1 + x = \frac{1}{1+x} - (1-x) = \frac{1-(1-x^2)}{1+x} = x^2/(1+x) > 0$ when $x > 0$. □

Lemma 15. *For $x > 0$, $\ln(x+1) \le 1 - \frac{1}{x+1} + \frac{1}{2}x^2$.*

Proof. The function $\ln(x+1) - 1 + \frac{1}{x+1} - \frac{1}{2}x^2$ is 0 at $x = 0$ and its derivative,

$$\frac{1}{x+1} - \frac{1}{(x+1)^2} - x = \frac{x - x(x+1)^2}{(x+1)^2} = \frac{x(1-(x+1)^2)}{(x+1)^2},$$

is negative for $x > 0$. □

Lemma 16. *There exists an $L_0 > 0$ such that, for all $S \subset [K]$ and $y \in [K]$, the function $\ln h_{S,\epsilon}(y|\theta)$ is an L_0-smooth function of θ. That is for all $\theta, \theta' \in \Delta$, we have*

$$\ln h_{S,\epsilon}(y|\theta) - \ln h_{S,\epsilon}(y|\theta') \leq \nabla_\vartheta \ln h_{S,\epsilon}(y|\theta)^\top (\vartheta - \vartheta') + L_0 \|\theta - \theta'\|^2.$$

Proof. Assume $h_{S,\epsilon}(y|\theta) \geq h_{S,\epsilon}(y|\theta')$. Using Lemma 15 with $x = \frac{h_{S,\epsilon}(y|\theta)}{h_{S,\epsilon}(y|\theta')} - 1 \geq 0$, we have

$$\begin{aligned}
\ln h_{S,\epsilon}(y|\theta) - \ln h_{S,\epsilon}(y|\theta') &\leq 1 - \frac{h_{S,\epsilon}(y|\theta')}{h_{S,\epsilon}(y|\theta)} + \frac{1}{2}\left(\frac{h_{S,\epsilon}(y|\theta)}{h_{S,\epsilon}(y|\theta')} - 1\right)^2 \\
&= \nabla_\vartheta \ln h_{S,\epsilon}(y|\theta)^\top (\vartheta - \vartheta') + \frac{1}{2}\left(\frac{h_{S,\epsilon}(y|\theta) - h_{S,\epsilon}(y|\theta')}{h_{S,\epsilon}(y|\theta')}\right)^2 \\
&\leq \nabla_\vartheta \ln h_{S,\epsilon}(y|\theta)^\top (\vartheta - \vartheta') + \frac{1}{2c^2}\left(h_{S,\epsilon}(y|\theta) - h_{S,\epsilon}(y|\theta')\right)^2 \qquad \text{(C.25)} \\
&\leq \nabla_\vartheta \ln h_{S,\epsilon}(y|\theta)^\top (\vartheta - \vartheta') + \frac{1}{2c^2}\|\theta - \theta'\|^2,
\end{aligned}$$

where the first term in the second line follows from Lemma 5, the third line follows from Lemma 4, and the last line follows from Lemma 10.

Now, assume $h_{S,\epsilon}(y|\theta) \leq h_{S,\epsilon}(y|\theta')$. By Lemma 14 with $x = \frac{h_{S,\epsilon}(y|\theta')}{h_{S,\epsilon}(y|\theta)} - 1 \geq 0$,

$$\ln h_{S,\epsilon}(y|\theta') - \ln h_{S,\epsilon}(y|\theta) \geq \left(\frac{h_{S,\epsilon}(y|\theta')}{h_{S,\epsilon}(y|\theta)} - 1\right) - \frac{1}{2}\left(\frac{h_{S,\epsilon}(y|\theta')}{h_{S,\epsilon}(y|\theta)} - 1\right)^2,$$

or, reversing the sign of inequality,

$$\begin{aligned}
\ln h_{S,\epsilon}(y|\theta) - \ln h_{S,\epsilon}(y|\theta') &\leq \left(1 - \frac{h_{S,\epsilon}(y|\theta')}{h_{S,\epsilon}(y|\theta)}\right) + \frac{1}{2}\left(\frac{h_{S,\epsilon}(y|\theta')}{h_{S,\epsilon}(y|\theta)} - 1\right)^2 \\
&\leq \nabla_\vartheta \ln h_{S,\epsilon}(y|\theta)^\top (\vartheta - \vartheta') + \frac{1}{2c^2}\left(h_{S,\epsilon}(y|\theta) - h_{S,\epsilon}(y|\theta')\right)^2,
\end{aligned}$$

where the third line follows from Lemma 4. Hence, we have arrived at the same inequality as (C.25). Hence, Lemma 16 holds with $L_0 = \frac{1}{2c^2}$. □

Second moment of the gradient at θ^*: Let the average log-marginal likelihoods be defined as

$$\Phi_n(\theta) := \frac{1}{n}\sum_{t=1}^{n} \ln h_{S_t,\epsilon}(Y_t|\theta), \quad n \geq 1. \qquad \text{(C.26)}$$

The following bound on the second moment of this average at θ^* will be useful.

Lemma 17. *For $\Phi_n(\theta)$ defined in* (C.26)*, we have*

$$E_{\theta^*}\left[\|\nabla_\vartheta \Phi_n(\theta^*)\|^2\right] \le \frac{1}{n}\max_{S\subset[K]} \mathrm{Tr}\left[F(\theta^*;S,\epsilon)\right].$$

Proof. First, we evaluate the mean at $\theta=\theta^*$.

$$E_{\theta^*}\left[\nabla_\vartheta \Phi_n(\theta^*)\right] = \frac{1}{n}\sum_{t=1}^{n} E_{\theta^*}\left[\nabla_\vartheta \ln h_{S_t,\epsilon}(Y_t|\theta^*)\right].$$

Focusing on a single term,

$$E_{\theta^*}\left[\nabla_\vartheta \ln h_{S_t,\epsilon}(Y_t|\theta^*)\right] = \sum_{S\subset[K]} P_{\theta^*}(S_t=S) E_{\theta^*}\left[\nabla_\vartheta \ln h_{S,\epsilon}(Y_t|\theta^*)|S_t=S\right].$$

Each term in the sum is equal to 0, since

$$E_{\theta^*}\left[\nabla_\vartheta \ln h_{S,\epsilon}(Y_t|\theta^*)|S_t=S\right] = \sum_{k=1}^{K} \nabla_\vartheta \ln h_{S,\epsilon}(k|\theta^*) h_{S,\epsilon}(k|\theta^*) = 0. \tag{C.27}$$

For the second moment at $\theta=\theta^*$,

$$\begin{aligned} E_{\theta^*}\left[\nabla_\vartheta \Phi_n(\theta^*)\nabla_\vartheta \Phi_n(\theta^*)^\top\right] =& \frac{1}{n^2}\sum_{t=1}^{n} E_{\theta^*}\left[\nabla_\vartheta \ln h_{S_t,\epsilon}(Y_t|\theta^*)\nabla_\vartheta \ln h_{S_t,\epsilon}(Y_t|\theta^*)^\top\right] \\ &+ \frac{2}{n^2}\sum_{t=1}^{n}\sum_{t'=1}^{t-1} E_{\theta^*}\left[\nabla_\vartheta \ln h_{S_t,\epsilon}(Y_t|\theta^*)\nabla_\vartheta \ln h_{S_{t'}}(Y_{t'}|\theta^*)^\top\right]. \end{aligned}$$

For the diagonal terms, for all $t=1,\dots,n$, we have

$$\begin{aligned} &E_{\theta^*}\left[\nabla_\vartheta \ln h_{S_t,\epsilon}(Y_t|\theta^*)\nabla_\vartheta \ln h_{S_t,\epsilon}(Y_t|\theta^*)^\top\right] \\ &\quad= \sum_{S\subset[K]} P_{\theta^*}(S_t=S) E_{\theta^*}\left[\nabla_\vartheta \ln h_{S,\epsilon}(Y_t|\theta^*)\nabla_\vartheta \ln h_{S,\epsilon}(Y_t|\theta^*)^\top|S_t=S\right] \\ &\quad= \sum_{S\subset[K]} P_{\theta^*}(S_t=S) F(\theta^*;S,\epsilon). \end{aligned}$$

For the cross terms, for $1\le t'<t\le n$,

$$\begin{aligned} E_{\theta^*}\left[\nabla_\vartheta \ln h_{S_t,\epsilon}(Y_t|\theta^*)\nabla_\vartheta \ln h_{S_{t'}}(Y_{t'}|\theta^*)^\top\right] &= E_{\theta^*}\left\{E_{\theta^*}\left[\nabla_\vartheta \ln h_{S_t,\epsilon}(Y_t|\theta^*)\nabla_\vartheta \ln h_{S_{t'}}(Y_{t'}|\theta^*)^\top|Y_{t'},S_{t'}\right]\right\} \\ &= E_{\theta^*}\left\{E_{\theta^*}\left[\nabla_\vartheta \ln h_{S_t,\epsilon}(Y_t|\theta^*)|Y_{t'},S_{t'}\right]\nabla_\vartheta \ln h_{S_t,\epsilon}(Y_{t'}|\theta^*)^\top\right\} \end{aligned}$$

The conditional expectation inside is zero, since, by (C.27),

$$E_{\theta^*}\left[\nabla_\vartheta \ln h_{S_t,\epsilon}(Y_t|\theta^*)|Y_{t'},A_{t'}\right] = \sum_{S\subset[K]} P_{\theta^*}(S_t=S|Y_{t'},S_{t'}) E_{\theta^*}\left[\nabla_\vartheta \ln h_{S,\epsilon}(Y_t|\theta^*)|S_t=S\right] = 0.$$

Therefore, all the cross terms are zero,

$$E_{\theta^*}\left[\nabla_\vartheta \ln h_{S_t,\epsilon}(Y_t|\theta^*)\nabla_\vartheta \ln h_{S_{t'}}(Y_{t'}|\theta^*)^\top\right] = 0,$$

and hence, we arrive at

$$E_{\theta^*}\left[\nabla_\vartheta \Phi_n(\theta^*)\nabla_\vartheta \Phi_n(\theta^*)^\top\right] = \frac{1}{n^2}\sum_{t=1}^{n}\sum_{S\subset[K]} P_{\theta^*}(S_t = S)\left[F(\theta^*;S,\epsilon)\right] \tag{C.28}$$

for the second moment of the gradient of $\Phi_n(\theta^*)$. Therefore,

$$\begin{aligned}
E_{\theta^*}\left[\|\nabla_\vartheta \Phi_n(\theta^*)\|^2\right] &= E_{\theta^*}\left[\mathrm{Tr}\left(\nabla_\vartheta \Phi_n(\theta^*)\nabla_\vartheta \Phi_n(\theta^*)^\top\right)\right] \\
&= \mathrm{Tr}\left(E_{\theta^*}\left[\nabla_\vartheta \Phi_n(\theta^*)\nabla_\vartheta \Phi_n(\theta^*)^\top\right]\right) \\
&= \frac{1}{n^2}\sum_{t=1}^{n}\sum_{S\subset[K]} P_{\theta^*}(S_t = S)\mathrm{Tr}(F(\theta^*;S,\epsilon)) \\
&\leq \frac{1}{n}\max_{S\subset[K]} \mathrm{Tr}\left(F(\theta^*;S,\epsilon)\right),
\end{aligned}$$

which concludes the proof. □

C.3.2 Convergence of the posterior distribution

Let $\mu \in (0,1)$ and, for $\theta, \theta' \in \Delta$, define

$$\mathcal{E}_n^\mu(\theta,\theta') := \mu c_u \|\theta - \theta'\|^2 - \frac{1}{n}\sum_{t=1}^{n} V_t(\theta,\theta'), \tag{C.29}$$

where $V_t(\theta,\theta')$ was defined in (C.16) and $c_u > 0$ was defined in the proof of Lemma 11, respectively. The proof of Theorem 8 requires the following lemma concerning $\mathcal{E}_n^\mu(\theta,\theta')$.

Lemma 18. *There exists $\mu \in (0,1)$ such that, for any $\varepsilon > 0$, we have*

$$\lim_{n\to\infty} P_{\theta^*}\left(\int_\Delta e^{nm_0\mathcal{E}_n^\mu(\theta,\theta^*)}\mathrm{d}\theta > e^\varepsilon\right) = 0.$$

Proof. Define the product measure

$$P^\otimes(\mathrm{d}(\theta,\cdot)) := \frac{\mathrm{d}\theta}{|\Delta|} \times dP_{\theta^*}(\cdot) \tag{C.30}$$

for random variables $(\Theta \in \Delta, \{S_t \subset \{1,\ldots,K\}, Y_t \in [K]\}_{t\geq 1})$, where $\mathrm{d}\theta$ is the

Lebesgue measure for ϑ restricted to Δ and $|\Delta| := \int_\Delta \mathrm{d}\theta$. We will show that the parameter μ in (C.29) can be chosen such that the collection of random variables

$$\mathcal{C} := \{f_n := \max\{1, e^{nm_0\mathcal{E}_n^\mu(\Theta,\theta^*)}\} : n \geq 1\}$$

is uniformly integrable with respect to $P^\otimes$. For uniform integrability, we need to show that for any $\varepsilon > 0$, there exists a $K > 0$ such that

$$E^\otimes[|f_n| \cdot \mathbb{I}(f_n > K)] < \varepsilon, \quad \forall n \geq 1.$$

For any $K > 1$ and $n \geq 1$, we have

$$\begin{aligned}
E^\otimes[|f_n| \cdot \mathbb{I}(f_n > K)] &= E^\otimes[e^{nm_0\mathcal{E}_n^\mu(\Theta,\theta^*)}\mathbb{I}(f_n > K)] \\
&\leq \sup_{\theta\in\Delta} e^{nm_0\mathcal{E}_n^\mu(\theta,\theta^*)} P^\otimes(f_n > K) \\
&= \sup_{\theta\in\Delta} e^{nm_0\mathcal{E}_n^\mu(\theta,\theta^*)} \int_\Delta P_{\theta^*}\left(\mathcal{E}_n^\mu(\theta,\theta^*) > \frac{\ln K}{nm_0}\right)\frac{\mathrm{d}\theta}{|\Delta|} \\
&\leq e^{n\mu m_0 c_u} P_{\theta^*}\left(\mathcal{E}_n^\mu(\theta,\theta^*) > 0\right),
\end{aligned}$$

where the last line follows from $\mathcal{E}_n^\mu(\theta,\theta^*) \leq \mu c_u \|\theta - \theta^*\|^2 \leq \mu c_u$. Using (C.24), the last expression can be upper-bounded as

$$\begin{aligned}
e^{nm_0\mu c_u} P_{\theta^*}\left(\mathcal{E}_n^\mu(\theta,\theta^*) > 0\right) &= e^{n\mu m_0 c_u} P_{\theta^*}\left(\frac{1}{n}\sum_{t=1}^{t} V_t(\theta,\theta^*) < \mu c_u \|\theta - \theta'\|^2\right) \\
&\leq e^{n\mu m_0 c_u}(1-c_u)^{\lceil n(1-\mu)\rceil} \prod_{i=1}^{\lceil n(1-\mu)\rceil} \frac{n-i+1}{n(1-\mu c_u)-i+1}. \qquad \text{(C.31)}
\end{aligned}$$

The parameter μ can be arranged such that (C.31) converges to 0. For such μ, we have that for any ε there exists a $N_\varepsilon > 0$ such that for all $n > N_\varepsilon$, $E^\otimes[|f_n| \cdot \mathbb{I}(f_n > K)] < \varepsilon$ for any $K > 0$. Finally, choose $K_\varepsilon = e^{N_\varepsilon m_0 \mu c_u}$ so that $E^\otimes[|f_n| \cdot \mathbb{I}(f_n > K_\varepsilon)] < \varepsilon$ for any $n \geq 1$. Hence, $\mathcal{C}$ is uniformly integrable for a suitable choice of μ.

Next, we show that each f_n in $\mathcal{C}$ converges in probability to 1. The convergence is implied by the fact that for every $\theta \in \Delta$ and $\varepsilon > 0$, we have

$$P_{\theta^*}(\max\{1, e^{nm_0\mathcal{E}_n^\mu(\theta,\theta^*)}\} > e^\varepsilon) = P_{\theta^*}(\mathcal{E}_n^\mu(\theta,\theta^*) > \varepsilon) \to 0.$$

by Lemma 13. Since $\mathcal{C}$ is uniformly integrable, the Vitali convergence theorem ensures that f_n converges in distribution (with respect to $P^\otimes$) to 1, i.e., $\lim_{n\to\infty} E^\otimes(f_n) = 1$. Since $P^\otimes = \frac{\mathrm{d}\theta}{|\Delta|} \times dP_{\theta^*}(\cdot)$ is a product measure as defined in

(C.30), the stated limit implies that

$$E_{\theta^*}\left[\int_\Delta \max\{1, e^{nm_0\mathcal{E}_n^\mu(\theta,\theta^*)}\}\mathrm{d}\theta\right] \to 1,$$

that is, the sequence $\int_\Delta \max\{1, e^{nm_0\mathcal{E}_n^\mu(\theta,\theta^*)}\}\mathrm{d}\theta$ converges to 1 in distribution with respect to P_{θ^*}. Since convergence in distribution to a constant implies convergence in probability, we have

$$\int_\Delta \max\{1, e^{nm_0\mathcal{E}_n^\mu(\theta,\theta^*)}\}\mathrm{d}\theta \xrightarrow{P_{\theta^*}} 1. \tag{C.32}$$

Finally, since we have

$$\int_\Delta e^{nm_0\mathcal{E}_n^\mu(\theta,\theta^*)}\mathrm{d}\theta \leq \int_\Delta \max\{1, e^{nm_0\mathcal{E}_n^\mu(\theta,\theta^*)}\}\mathrm{d}\theta,$$

and the right-hand side converges in probability to 1, we conclude. □

Proof of Theorem 8. Writing down Lemma 9 with θ^* and any $\theta \in \Delta$ separately for $t = 1, \ldots, n$, summing the inequalities and dividing by n, we obtain

$$\begin{aligned}\Phi_n(\theta^*) - \Phi_n(\theta) &\geq \nabla_\vartheta \Phi_n(\theta^*)^\top(\vartheta^* - \vartheta) + m_0 \sum_{t=1}^n V_t(\theta, \theta^*) \\ &= \nabla_\vartheta \Phi_n(\theta^*)^\top(\vartheta^* - \vartheta) + m\|\theta - \theta^*\|^2 - m_0\mathcal{E}_n^\mu(\theta, \theta^*).\end{aligned}$$

where $m := \mu m_0 c_u$. Reversing the sign,

$$\Phi_n(\theta) - \Phi_n(\theta^*) \leq \nabla_\vartheta \Phi_n(\theta^*)^\top(\vartheta - \vartheta^*) - m\|\theta^* - \theta\|^2 + m_0\mathcal{E}_n^\mu(\theta, \theta^*).$$

Using Cauchy-Schwarz inequality for the first term on the right-hand side, we get

$$\Phi_n(\theta) - \Phi_n(\theta^*) \leq \|\nabla_\vartheta \Phi_n(\theta^*)\|\|\theta - \theta^*\| - m\|\theta^* - \theta\|^2 + m_0\mathcal{E}_n^\mu(\theta, \theta^*).$$

Using Young's inequality $uv \leq \frac{u^2}{2\kappa} + \frac{v^2\kappa}{2}$ for the second term with $u = \|\nabla_\vartheta \Phi_n(\theta^*)\|$, $v = \|\theta^* - \theta\|$, and $\kappa = m$, we get

$$\Phi_n(\theta) - \Phi_n(\theta^*) \leq \frac{\|\nabla_\vartheta \Phi_n(\theta^*)\|^2}{2m} - \frac{m}{2}\|\theta^* - \theta\|^2 + m_0\mathcal{E}_n^\mu(\theta, \theta^*). \tag{C.33}$$

Similarly, using Lemma 16 with θ^* and any $\theta' \in \Delta$ for $t = 1, \ldots, n$, summing the inequalities and dividing by n, we obtain

$$\Phi_n(\theta^*) - \Phi_n(\theta') \leq \nabla_\vartheta \Phi_n(\theta^*)^\top(\vartheta^* - \vartheta') + L_0\|\theta^* - \theta'\|^2.$$

Again, using Cauchy-Schwarz inequality and Young's inequality $uv \leq \frac{u^2}{2\kappa} + \frac{v^2\kappa}{2}$ with $u = \|\nabla_\vartheta \Phi_n(\theta^*)\|$, $v = \|\theta^* - \theta\|$, and $\kappa = 2L_0$, we get

$$\begin{aligned}\Phi_n(\theta^*) - \Phi_n(\theta') &\leq \|\nabla_\vartheta \Phi_n(\theta^*)\|\|\theta^* - \theta'\| + L_0\|\theta^* - \theta'\|^2 \\ &\leq \frac{\|\nabla_\vartheta \Phi_n(\theta^*)\|^2}{4L_0} + 2L_0\|\theta^* - \theta'\|^2 \\ &= \frac{\|\nabla_\vartheta \Phi_n(\theta^*)\|^2}{2L} + L\|\theta^* - \theta'\|^2, \end{aligned} \tag{C.34}$$

where we let $L := 2L_0$. Summing the inequalities in (C.33) and (C.34), we obtain

$$\Phi_n(\theta) - \Phi_n(\theta') \leq \left(\frac{1}{2L} + \frac{1}{2m}\right)\|\nabla_\vartheta \Phi_n(\theta^*)\|^2 - \frac{m}{2}\|\theta^* - \theta\|^2 + \frac{L}{2}\|\theta^* - \theta'\|^2 + m_0\mathcal{E}_n^\mu(\theta, \theta^*). \tag{C.35}$$

Let $a \in (0,1)$ be a constant and define the sequences

$$\begin{aligned}\Omega_n &:= \{\theta \in \Delta : \|\theta - \theta^*\|^2 < \max\{4/m, 4/L\}n^{-a}\}, \quad n \geq 1. \\ A_n &:= \{\theta \in \Delta : \|\theta - \theta^*\|^2 > \max\{4/m, 4/L\}n^{-a}\}, \quad n \geq 1. \\ B_n &:= \{\theta \in \Delta : \|\theta - \theta^*\|^2 \leq \min\{2/m, 2/L\}n^{-a}\}, \quad n \geq 1.\end{aligned}$$

For $\theta \in A_n$ and $\theta' \in B_n$, (C.35) can be used to obtain

$$\Phi_n(\theta) - \Phi_n(\theta') \leq \left(\frac{1}{2L} + \frac{1}{2m}\right)\|\nabla_\vartheta \Phi_n(\theta^*)\|^2 - n^{-a}\left(\max\left\{2, \frac{2m}{L}\right\} - \min\left\{\frac{L}{m}, 1\right\}\right) + m_0\mathcal{E}_n^\mu(\theta, \theta^*).$$

Noting that $\max\left\{2, \frac{2m}{L}\right\} - \min\left\{\frac{L}{m}, 1\right\} \geq 1$, we have

$$\Phi_n(\theta) - \Phi_n(\theta') \leq \left(\frac{1}{2L} + \frac{1}{2m}\right)\|\nabla_\vartheta \Phi_n(\theta^*)\|^2 - n^{-a} + m_0\mathcal{E}_n^\mu(\theta, \theta^*), \quad \theta \in A_n; \theta' \in B_n. \tag{C.36}$$

Multiplying (C.36) with n, exponentiating, and multiplying the ratio of the priors, we get

$$\begin{aligned}\frac{\eta(\theta)\exp\{n\Phi_n(\theta)\}}{\eta(\theta')\exp\{n\Phi_n(\theta')\}} &\leq \frac{\eta(\theta)}{\eta(\theta')}\exp\left[\left(\frac{1}{2L} + \frac{1}{2m}\right)n\|\nabla_\vartheta \Phi_n(\theta^*)\|^2 - n^{1-a} + nm_0\mathcal{E}_n^\mu(\theta, \theta^*)\right] \\ &\leq C_{\eta,n}\exp\left[C_1 n\|\nabla_\vartheta \Phi_n(\theta^*)\|^2 - n^{1-a} + nm_0\mathcal{E}_n^\mu(\theta, \theta^*)\right] \end{aligned} \tag{C.37}$$

for all $\theta \in A_n$ and $\theta' \in B_n$, where $C_1 := \frac{1}{2L} + \frac{1}{2m}$ and $C_{\eta,n} := \sup_{\theta \in A_n, \theta' \in B_n} \frac{\eta(\theta)}{\eta(\theta')}$. The bound in (C.37) can be used to bound the ratio between the posterior probabilities

$\Pi(A_n|Y_{1:n}, S_{1:n})$ and $\Pi(B_n|Y_{1:n}, S_{1:n})$, since

$$\begin{aligned}
\frac{\Pi(A_n|Y_{1:n}, S_{1:n})}{\Pi(B_n|Y_{1:n}, S_{1:n})} &= \frac{\int_{A_n} \eta(\theta) \exp\{n\Phi_n(\theta)\} \mathrm{d}\theta}{\int_{B_n} \eta(\theta) \exp\{n\Phi_n(\theta)\} \mathrm{d}\theta} \\
&= \frac{\int_{A_n} \frac{\eta(\theta) \exp\{n\Phi_n(\theta)\}}{\inf_{\theta' \in B_n} \eta(\theta') \exp\{n\Phi_n(\theta')\}} \mathrm{d}\theta}{\int_{B_n} \frac{\eta(\theta) \exp\{n\Phi_n(\theta)\}}{\inf_{\theta' \in B_n} \eta(\theta') \exp\{n\Phi_n(\theta')\}} \mathrm{d}\theta} \\
&\leq \frac{\int_{A_n} \frac{\eta(\theta) \exp\{n\Phi_n(\theta)\}}{\inf_{\theta' \in B_n} \eta(\theta') \exp\{n\Phi_n(\theta')\}} \mathrm{d}\theta}{\int_{B_n} \frac{\eta(\theta) \exp\{n\Phi_n(\theta)\}}{\inf_{\theta' \in B_n} \eta(\theta') \exp\{n\Phi_n(\theta')\}} \mathrm{d}\theta} \\
&\leq \frac{\int_{A_n} C_{\eta,n} \exp\left[C_1 n \|\nabla_\vartheta \Phi_n(\theta^*)\|^2 - n^{1-a} + n m_0 \mathcal{E}_n^\mu(\theta, \theta^*)\right] \mathrm{d}\theta}{\int_{B_n} 1 \mathrm{d}\theta} \\
&= \frac{1}{\mathrm{Vol}(B_n)} C_{\eta,n} \exp\left[C_1 n \|\nabla_\vartheta \Phi_n(\theta^*)\|^2 - n^{1-a}\right] \int_{A_n} \exp\left[n m_0 \mathcal{E}_n^\mu(\theta, \theta^*)\right] \mathrm{d}\theta,
\end{aligned}$$

where $\mathrm{Vol}(B_n) := \int_{B_n \cap \Delta} \mathrm{d}\theta$. Note that B_n shrinks with n, so there exists a N_B such that for $n > N_B$, the volume B_n can be lower-bounded as

$$B_n \geq \frac{1}{2(K-2)!} \frac{(\sqrt{\pi} \min\{2/m, 2/L\} n^{-a})^{(K-1)}}{\Gamma((K-1)/2+1)},$$

where the factor $\frac{1}{2(K-2)!}$ corresponds to the worst-case situation where θ^* is on one of the corners of Δ, such as $\theta^* = (1, 0, \ldots, 0)^\top$, and the rest is the volume of a $K-1$ dimensional sphere with radius $\min\{2/m, 2/L\} n^{-\alpha}$. The lower bound is the volume of the intersection of a simplex with a sphere centered at one of the sharpest corners of the simplex. Therefore, for $n > N_B$, ratio can further be bounded as

$$\frac{\Pi(A_n|Y_{1:n}, S_{1:n})}{\Pi(B_n|Y_{1:n}, S_{1:n})} \leq C_2 C_{\eta,n} \exp\left[C_1 n \|\nabla_\vartheta \Phi_n(\theta^*)\|^2 + (K-1) a \ln n - n^{1-a}\right] \int_{A_n} \exp\left[n m_0 \mathcal{E}_n^\mu(\theta, \theta^*)\right] \mathrm{d}\theta,$$

where $C_2 := \frac{\Gamma((K-1)/2+1)}{\min\{2/m, 2/L\}^{K-1} \pi^{(K-1)/2}}$ does not depend on n.

Next, we prove that the sequence of random variables

$$Z_n := C_{\eta,n} \exp\left[C_1 n \|\nabla_\vartheta \Phi_n(\theta^*)\|^2 + (K-1) a \ln n - n^{1-a}\right] \int_{A_n} \exp\left[n m_0 \mathcal{E}_n^\mu(\theta, \theta^*)\right] \mathrm{d}\theta$$

converges to 0 in probability, which in turn proves the convergence of $\frac{\Pi(A_n|Y_{1:n}, S_{1:n})}{\Pi(B_n|Y_{1:n}, S_{1:n})}$ in probability to 0. To do that, we need to prove that for each $\varepsilon > 0$ and $\delta > 0$, there exists a $N > 0$ such that for all $n > N$ we have $P_{\theta^*}(Z_n \geq 2\varepsilon) < 2\delta$. Fix $\varepsilon > 0$ and $\delta > 0$.

- Firstly, by Assumption 1, because B_n shrinks towards θ^*, there exists $N_\eta > 0$ such that $C_{\eta,n} < B$ for all $n > N_\eta$.

- Next, let $\beta := C_1 \max_{S\subset[K]} \mathrm{Tr}(F(\theta^*;S,\epsilon))/\delta$. Using Markov's inequality for $\|\nabla_\vartheta \Phi_n(\theta^*)\|^2$ with Lemma 17, we have

$$P_{\theta^*}\left(\|\nabla_\vartheta \Phi_n(\theta^*)\|^2 \geq \frac{1}{n}\frac{\beta}{C_1}\right) \leq \frac{1}{n}\max_{S\subset\{1,\dots,K\}} \mathrm{Tr}(F(\theta;S,\epsilon))\frac{C_1 n}{\beta} = \delta.$$

  Also, since the n^{1-a} dominates the term $\ln n$, one can choose an integer $N_\Phi > 0$ such that

$$\beta \leq \ln(\varepsilon/B) + n^{1-a} - (K-1)a\ln n, \quad \forall n \geq N_\Phi.$$

- Now we deal with the integral in Z_n. We have

$$P_{\theta^*}\left(\int_{A_n} \exp\left[nm_0\mathcal{E}_n^\mu(\theta,\theta^*)\right]\mathrm{d}\theta \geq 2\right) \leq P_{\theta^*}\left(\int_{\Delta} \exp\left[nm_0\mathcal{E}_n^\mu(\theta,\theta^*)\right]\mathrm{d}\theta \geq 2\right) \to 1,$$

  where the convergence is due to Lemma 18. Hence, there exists a $N_\mathcal{E}$ such that for all $n > N_\mathcal{E}$,

$$P_{\theta^*}\left(\int_{A_n} \exp\left[nm_0\mathcal{E}_n^\mu(\theta,\theta^*)\right]\mathrm{d}\theta \geq 2\right) \leq \delta.$$

Gathering the results, for $n > \max\{N_\eta, N_\Phi, N_\mathcal{E}\}$, we have

$$\begin{aligned}
P_{\theta^*}(Z_n \geq \varepsilon) &\leq P_{\theta^*}\left(e^{C_1 n\|\nabla_\vartheta \Phi_n(\theta^*)\|^2 + (K-1)a\ln n - n^{1-a}} \geq \varepsilon/B\right) + P_{\theta^*}\left(\int_{\Delta} \exp\left[nm_0\mathcal{E}_n^\mu(\theta,\theta^*)\right]\mathrm{d}\theta \geq 2\right) \\
&\leq P_{\theta^*}\left(C_1 n\|\nabla_\vartheta \Phi_n(\theta^*)\|^2 + (K-1)a\ln n - n^{1-a} \geq \ln(\varepsilon/B)\right) + \delta \\
&= P_{\theta^*}\left(n\|\nabla_\vartheta \Phi_n(\theta^*)\|^2 \geq \frac{\ln(\varepsilon/B) + n^{1-a} - (K-1)a\ln n}{C_1}\right) + \delta \\
&\leq P_{\theta^*}\left(\|\nabla_\vartheta \Phi_n(\theta^*)\|^2 \geq \frac{\beta}{C_1 n}\right) + \delta \\
&\leq 2\delta.
\end{aligned}$$

(In the first line, we have used $\mathbb{P}(XY > pq) = 1 - \mathbb{P}(XY < pq) \leq 1 - \mathbb{P}(X < p \text{ and } Y < q) = \mathbb{P}(X > p \text{ or } Y > q) \leq \mathbb{P}(X > p) + \mathbb{P}(Y > q)$ for non-negative random variables X, Y and positive p, q.) Therefore we have proved that $Z_n \to 0$ in probability. Finally, since $B_n \subset \Omega_n$, we have

$$\frac{\Pi(A_n|Y_{1:n},S_{1:n})}{\Pi(\Omega_n|Y_{1:n},S_{1:n})} \leq \frac{\Pi(A_n|Y_{1:n},S_{1:n})}{\Pi(B_n|Y_{1:n},S_{1:n})} \leq C_2 Z_n$$

for all $n > N_B$. This implies that,

$$\frac{\Pi(A_n|Y_{1:n},S_{1:n})}{\Pi(\Omega_n|Y_{1:n},S_{1:n})} \xrightarrow{P_{\theta^*}} 0.$$

Since $A_n = \Delta/\Omega_n$, as a result we get $\Pi(\Omega_n|Y_{1:n}, S_{1:n}) \overset{P_{\theta^*}}{\to} 1$. This concludes the proof. □

C.3.3 Convergence of the expected frequency

Proof of Theorem 9. Assumption 4 ensures that there exists a $\kappa_0 > 0$ and $k^* \in \{0, \ldots, K-1\}$ such that for all $0 \leq k \neq k^* < K$,

$$U(\theta^*; S^*, \epsilon) - U(\theta^*; \{\sigma_{\theta^*}(1), \ldots, \sigma_{\theta^*}(k)\}, \epsilon) \geq \kappa_0.$$

By Assumption 3, there exists a $\delta_1 > 0$ such that

$$\|\theta - \theta'\| \leq \delta_1 \Rightarrow |U(\theta, S, \epsilon) - U(\theta', S, \epsilon)| < \kappa_0/2.$$

Moreover, since the components of θ^* are strictly ordered,

$$\delta_2 := \min_{k=1,\ldots,K-1} (\theta^*(k) - \theta^*(k+1)) > 0.$$

Choose $\delta = \min\{\delta_1, \delta_2/\sqrt{2}\}$. Define the set

$$\Omega_\delta = \{\theta \in \Delta : \|\theta - \theta^*\|^2 \leq \delta^2\}.$$

Then, for any $\theta \in \Omega_\delta$, $\sigma_\theta = \sigma_{\theta^*}$ and $S^*_\theta = S^*$. This implies that $\{\theta_n \in \Omega_\delta\} \subseteq \{S_{n+1} = S^*\}$. Since perfect sampling is assumed, we have $Q(\mathrm{d}\theta_t|Y_{1:t}, S_{1:t}) = \Pi(\mathrm{d}\theta_t|Y_{1:t}, S_{1:t})$. Hence,

$$P_{\theta^*}(S_{n+1} = S^*) \geq E_{\theta^*}\left[P_{\theta^*}(\theta_n \in \Omega_\delta|S_{1:n}, Y_{1:n})\right] = E_{\theta^*}\left[\Pi(\Omega_\delta|Y_{1:n}, S_{1:n})\right] \tag{C.38}$$

Recall the sequence of sets

$$\Omega_n = \{\theta \in \Delta : \|\theta - \theta^*\|^2 \leq cn^{-a}\}$$

defined in Theorem 8. There exists an $N_1 > 0$ such that $n > N_1$ we have $\Omega_n \subseteq \Omega_\delta$. For such N_1, we have

$$E_{\theta^*}\left[\Pi(\Omega_\delta|Y_{1:n}, S_{1:n})\right] \geq E_{\theta^*}\left[\Pi(\Omega_n|Y_{1:n}, S_{1:n})\right], \quad n > N_1.$$

Combining with (C.38), we can write as

$$P_{\theta^*}(S_{n+1} = S^*) \geq E_{\theta^*}\left[\Pi(\Omega_n|Y_{1:n}, S_{1:n})\right], \quad n > N_1. \tag{C.39}$$

We will show that the right-hand side converges to 1. To do that, fix $\varepsilon > 0$. By Theorem 8, there exists a $N_2 > 0$ such that

$$P_{\theta^*}\left(\Pi(\Omega_n|Y_{1:n}, S_{1:n}) > \sqrt{1-\varepsilon}\right) > \sqrt{1-\varepsilon}.$$

This implies that, for $n > N_2$,

$$E_{\theta^*}(\Pi(\Omega_n|Y_{1:n}, S_{1:n})) > \sqrt{1-\varepsilon}\sqrt{1-\varepsilon} + 0(1-\sqrt{1-\varepsilon}) = 1-\varepsilon.$$

This shows that $E_{\theta^*}\left[\Pi(\Omega_n|Y_{1:n}, S_{1:n})\right] \to 1$ as $n \to \infty$. Since the right-hand side of (C.39) converges to 1, so does the left-hand side. Therefore, we have proven (5.19).

To prove (5.20), we utilize the convergence of Cesaro means and write

$$\lim_{n\to\infty} \frac{1}{n} \sum_{t=1}^{n} P_{\theta^*}(S_t = S^*) = \lim_{t\to\infty} P_{\theta^*}(S_t = S^*) = 1,$$

where the last equality is by (5.19). Finally, we replace $P_{\theta^*}(S_t = S)$ by $E_\theta(\mathbb{I}(S_t = S_t))$ on the left-hand side and conclude the proof. □