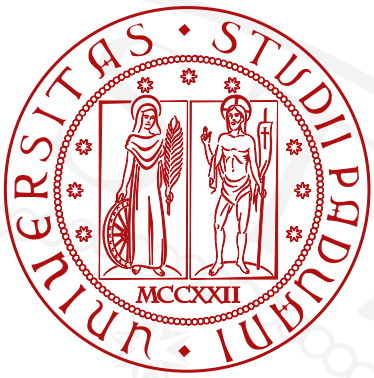


Università degli Studi di Padova

Master Thesis in Computer Science

Local Differential Privacy Mechanisms for Frequency Estimation with Application to Mobility Data

Master Candidate
Ava Louise Finnegan
Student ID 2072113

Supervisor
Prof. Francesco Silvestri
University of Padova

Co-supervisor
Dott. Fabrizio Boninsegna
University of Padova

Academic Year
2023/2024

Abstract

Differential Privacy is a powerful mathematical tool that is applied to data allowing aggregate statistics to be released while protecting each individual's privacy. These aggregate statistics allow for information to be inferred about a population. In this work, we will focus on frequency estimation particularly on origin-destination commuting data. Origin-destination commuting data is sensitive due to its association with individual locations and it is under the protection of the General Data Protection Regulation (GDPR). Despite this sensitive nature, this data plays a crucial role in a number of scenarios such as the planning of public transportation systems. A solution to this problem is to use differential privacy to add noise in a controlled way to the dataset. The aim is to strike a balance between ensuring individuals privacy while maximising utility of the data.

Origin-Destination commuting happens on many hierarchical levels. Commuting occurs country to country, county to county, city to city and so forth. A characteristic of origin-destination commuting data is vast datasets with numerous potential journeys, many of which remain unused, resulting in highly sparse data. Our in-depth analysis will examine a number of differentially private mechanisms. Two categories of differentially private techniques will be studied namely central differential privacy and local differential privacy. The errors of the various mechanisms will be analysed. We will present the benefits of each mechanism and preform experiments on an origin-commuting dataset. We will investigate the the trade-offs between different differentially private mechanisms.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Differential Privacy is a mathematical definition of privacy which quantifies the amount of information an adversary can get about any individuals. Privacy is a very important topic especially in our current society with the abundance of data. Our data is extremely powerful and considered one of the most important commodities. In this data-rich society there is an overwhelming need for privacy to prevent against adversary attacks which can prevail against popular privacy techniques such an pseudonymisation or anonymisation.

In 2006, Netflix held a competition offering a cash prize for the improvement of their recommendation system which is integral to their service as a streaming platform. They publicly released a dataset containing over a 100 million movie ratings from nearly half a million subscribers. Each row consisted of an anonymised user ID, a movie ID, the rating and the date of said rating. This data is deemed as sensitive and thus measures were taken so that individuals could not be identified from this dataset alone. However this was insufficient to protect the users contained in this dataset. Narayanan & Shmatikov demonstrated this fact by cross-referencing the Netflix dataset with publicly available IMDb data where the users could be identified. [NS08]. The dataset was no longer private, Netflix did not release another competition and there was a legal case taken against them. This breach of users data destroys trust, has legal ramifications and is also a deterrent to innovation and growth.

The Netflix attack is known as a linkage attack where anonymised datasets can be 'linked' with auxiliary data from datasets in order to de-anonymise the dataset. This is a difficult attack to predict as the auxiliary data is clearly not known beforehand. Another type of attack known as a differencing attack in which multiple queries to a dataset can expose facts about individuals.

Having mathematical guarantees against individual's data being leaked can lead to the accessible use of data. This can lead to gains and technological advances such as a hospital making available a dataset on a disease such that experts in statistics or machine learning can make advances. Notably Google has utilised differential privacy to learn statistics from their users with their technology known as RAPPOR (Randomised Aggregatable Privacy-Preserving Ordinal Response) [EKP14] In 2020, the American Census used differential privacy as the basis of their disclosure avoidance system (DAS) which is designed to withstand re-identification threats against the US Census. [Abo]

Within the differential privacy framework, there are two main settings: central and local. Both of these are investigated in this thesis but there is an emphasis on the local setting. In central differential privacy (DP) the users send their sensitive data to a central aggregator where the

data undergoes a perturbation. The weakness in this setting is the trust in the central aggregator. Local differential privacy (LDP) overcomes this issue. Unlike central privacy, the data perturbation occurs locally, usually on users devices. The trade-off between privacy and utility are a challenge in all differential privacy frameworks but additional considerations regarding computation time and communication cost are required in the local setting.

The data released from these mechanisms are aggregate statistics about a population. These datasets are usually composed of a large number of individuals and so mechanisms must handle large amounts of data. Our focus is the estimation of elements' frequencies which in this thesis is commutes between an origin and a destination. Origin-destination commuting occurs across various hierarchical levels resulting in extensive datasets with numerous potential journeys. Frequency estimation of origin-destination commuting is highly valuable for urban planning, building transportation infrastructure, and providing services to different areas. Differential Privacy is a solution to protect origin-destination commuting data, one of the most sensitive categories of data. The overarching goal of this thesis is to provide a study of the state-of the art LDP mechanisms and their application on a commuting dataset. The mechanisms must be differentially private, provide good accuracy and demonstrate scalability. The thesis is structured as follows:

Chapter 2: The problem is outlined and the background knowledge, theorems used and the state of the art is reviewed.

Chapter 3: The mechanisms are explained in depth, proved to be differentially private. They are shown to be unbiased estimators for frequency estimation and their error bounds are calculated.

Chapter 4: The commuting dataset 'POWSCAR' is explored and key observations are made.

Chapter 5: A comprehensive analysis is conducted with a main focus on privacy, utility and computation cost. The payoffs of using different mechanisms is discussed.

After comparing 4 LDP mechanisms, we found that the Hadamard Mechanism performed the best on our origin-commuting dataset considering both the accuracy and the computation time. This LDP mechanism successfully identified the most common commutes. However, when the size of the dataset was increased, the utility was greatly reduced.

Chapter 2

Preliminaries

We will now describe the set-up of our problem. We will also provide the background of differential privacy along-with the state of the art papers. We aim to give a complete explanation of the topic before moving on with our in depth analysis of the mechanisms in Chapter 3.

2.1 Problem Set-Up

Let $\mathcal{X}$ be a finite data universe, a dataset $D \in \mathcal{X}^n$ is a collection of elements from the data universe $D = \{x_i\}_{i=1,\dots,n}$. Our data universe is a set of origin-destination commutes ie. $\mathcal{X} =$ {Dublin-Cork, Dublin-Galway, Cork-Galway} and the dataset is a collection of individuals. Let n denote the number of individuals in the dataset and $|\mathcal{X}|$ denote the size of the data universe. Each individual reports exactly one element and we focus on the situation in which the list of possible elements is known in advance. We will apply differentially private mechanisms on data universes for the estimation of counting queries.

A *counting query* $q : \mathcal{X}^n \times \mathcal{X} \to [0, 1]$ takes as input a dataset and a element (or category) in the data universe, and returns the relative frequency of that element in the dataset

$$q(D, y) := \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\{x_i = y\}, \tag{2.1}$$

where the indicator function $\mathbb{1}$ returns one if the predicate inside the bracket is satisfied, otherwise it returns zero. A *histogram query* $\mathbf{h} : \mathcal{X}^n \to [0, 1]^{|\mathcal{X}|}$ is the vector of all the counting queries, so it represents the empirical distribution of the dataset

$$\mathbf{h}(D) := \big(q(D, y_1), \dots, q(D, y_{|\mathcal{X}|})\big).$$

Errors For these queries are important two errors: the *per entry error* and the *maximum error*. In the following $\tilde{q}$ indicates a differentially private estimate of q (the same for $\mathbf{h}$).

- The *per entry error* is the error of the counting query, hence $|\tilde{q}(D, y) - q(D, y)|$ for $y \in \mathcal{X}$.
- The *maximum error* is the max error of a counting query, hence the L_∞ error of the histogram query
$$\max_{y \in \mathcal{X}} |\tilde{q}(D, y) - q(D, y)| = ||\tilde{\mathbf{h}}(D) - \mathbf{h}(D)||_\infty$$

2.2 Differential Privacy

Differential Privacy was first defined by Dwork, McSherry, Nissim and Smith in their 2006 paper [DMNS06] and won the 2017 Gödel Prize.

We have discussed the need for differential privacy and also how it is resistant to external attacks such as linkage attacks. We now proceed to dive deeper into the benefits of differential privacy as a means to protect our data. Differential privacy is a promise to protect an individual from any additional harm in regards to the inclusion of their data in a dataset. There is an important distinction to be made about this additional harm as conclusions of a study may still impact an individual. This can happen regardless if their data was part of the dataset.

An example which outlines this distinction between privacy guarantees and the consequence of differentially private results is taken from [CD14]. There is a study which investigates the affect of smoking on our health. Assume we have an individual Susan who is a smoker. She is debating whether or not to participate in this study as she is worried that her involvement could affect her insurance. Under differential privacy, her involvement in the study will not change the outcome. However, if the outcome of the study is that 'Smoking causes cancer' then Susan's insurance will be raised. This is not a violation of differential privacy but rather an insight into a population. This outlines that even though differential privacy is a strong privacy guarantee, it does not promise unconditional protection from harm. Conclusions drawn from a survey may reflect the situation of individuals. The same outcome would be observed with almost the same probability independent of the involvement of any respondent.

We now analyse the definition of differential privacy at a high level before moving to the formal definition. We are given two datasets which are neighbours meaning they differ in the data of a single individual. Then we run each dataset through a differentially private mechanism $\mathcal{M}$. The output $\mathcal{Y}$ of the mechanism is the privatised dataset or query. For a mechanism to be ϵ-differentially private, the output distributions of the two datasets must have a distance of at most ϵ. This ϵ is a parameter known as the privacy budget. We will now discuss what notion of distance is chosen and why.

Total Variation Distance

Total variation distance also known as the statistical distance is the first choice we consider. It is defined as the maximum distance between the first dataset D and the second dataset D'.

$$\max_{T \subseteq \mathcal{Y}} |\Pr[M(D) \in T] - \Pr[M(D') \in T]| \leq \epsilon$$

This is not sufficient for privacy. To demonstrate this, take $\epsilon = 1/2n$. A mechanism which outputs a random row from the original dataset half of the time satisfies the total variation difference with this ϵ. This is an obvious breach of privacy. The individual whose data was released has been impacted by their involvement in the dataset. This distances measure ensures that the output of a mechanism does not change significantly with the presence or absence of a single individual. However, this can lead to some individuals being exposed especially in cases when their input is rare.

Bayesian Interpretation

The condition we use for differential privacy is that $\forall T \subseteq \mathcal{Y}$

$$\Pr[M(D) \in T] \leq e^{\epsilon}\Pr[M(D') \in T]|$$

This is a stronger condition than the total variation distance. The Bayesian interpretation of differential privacy allows us to understand why this multiplicative measure is a good choice. This was formulated explicitly by Kasiviswanathan & Smith [KS14]. They show that this definition of differential privacy is resistant to attacks by an adversary with auxiliary information. They provide a precise guarantee proven in terms of the inferences drawn by a Bayesian adversary. The posterior belief of the adversary is considered which is their prior belief about the dataset updated by the conditional distribution output by the mechanism. They verify that the posterior belief of the adversary when the mechanism is run on dataset D is close to the posterior belief when the mechanism is run on D'. This shows that differential privacy has meaningful privacy guarantees even in the presence of arbitrary side information.

2.2.1 Definition of Concepts

There are different flavours of differential privacy but we will focus on two main types which we formally define. These are pure differential privacy and approximate differential privacy which have both satisfy the Bayesian interpretation.

Definition 2.2.1 (Pure Differential Privacy [CD14]) *Given an algorithm $\mathcal{M} : \mathcal{X}^n \to \mathcal{Y}$. Consider any two datasets $D, D' \in X^n$ which differ in exactly one entry. We call these neighbouring datasets which we can denote as $D \sim D'$. We say that $\mathcal{M}$ is ϵ-(pure) differentially private (ϵ-(pure) DP) if, for all neighbouring D, D', and all $T \subseteq \mathcal{Y}$, we have*

$$Pr[\mathcal{M}(D) \in T] \leq e^{\epsilon} Pr[\mathcal{M}(D') \in T]$$

where the randomness is over the choices made by $\mathcal{M}$.

Differential privacy protects an individuals privacy as the true answer to a query is perturbed by the addition of random noise generated to a carefully chosen distribution. The response returned is the true answer to the query plus noise. We know define a relaxation of pure differential privacy known as approximate differential privacy.

Definition 2.2.2 (Approximate Differential Privacy [CD14]) *We say that $\mathcal{M}$ is ϵ-(approximately) differentially private ((ϵ, δ)-DP) if, for all neighbouring D, D', and all $T \subseteq \mathcal{Y}$, we have*

$$Pr[\mathcal{M}(D) \in T] \leq e^{\epsilon} Pr[\mathcal{M}(D') \in T] + \delta \tag{2.2}$$

where the randomness is over the choices made by $\mathcal{M}$.

Differential privacy is not binary but rather the level of privacy depends on the privacy budget ϵ. In literature, ϵ usually ranges from 0.5 to 10 and values above this range should be treated wearily. The smaller the value of ϵ, the higher the privacy. One should bear in mind that a high privacy budget lowers the utility. This is due to the higher level of noise added to the data. At the other extreme, an $\epsilon < 1/n$ is not useful as the utility of the data is lost. In approximate

differential privacy, $\delta \ll 1/n$ is chosen so that every user has a constant probability to be (ϵ, δ)-DP protected. This will be shown in Section 3.1.2.

Differential privacy also supports group privacy allowing a group such as a family to have privacy guarantees. The strength of the privacy guarantee decreases linearly with the number of individuals in a group. Thus a smaller privacy budget ϵ should be chosen for the same level of privacy as an individual.

Definition 2.2.3 (Group Differential Privacy [CD14]) *Any ϵ-differentially private mechanism $\mathcal{M}$ is $(k\epsilon)$-differentially private for groups of size k. That is, for all $||D - D'||_1 \leq k$ and all $S \in \text{Range}(\mathcal{M})$*

$$Pr[\mathcal{M}(D) \in S] \leq e^{(k\epsilon)} Pr[\mathcal{M}(D') \in S]$$

where the probability space is over the coin flips of the mechanism $\mathcal{M}$.

Composition allows for the combination of multiple DP mechanisms and is fundamental to differential privacy. This allows developers to innovate their own solutions using mechanisms as building blocks for a differentially private system. The composition theorem again provides mathematically guarantees for the use of multiple mechanisms. It allows for a simple analysis and understanding of the privacy guarantees for individuals which inevitably aids design of novel methods.

Definition 2.2.4 (Composition Theorem [CD14]) *Let $\mathcal{M}_1 : \mathbb{N}^{|\mathcal{X}|} \to \mathcal{R}_1$ be an ϵ_1-DP algorithm, and let $\mathcal{M}_2 : \mathbb{N}^{|\mathcal{X}|} \to \mathcal{R}_2$ be an ϵ_2-DP algorithm. Then their combination, defined to be $\mathcal{M}_{1,2} : \mathbb{N}^{|\mathcal{X}|} \to \mathcal{R}_1 \times \mathcal{R}_2$ by the mapping: $\mathcal{M}_{1,2}(D) = (\mathcal{M}_1(D), \mathcal{M}_2(D))$ is $\epsilon_1 + \epsilon_2$-differentially private.*

Another advantage of differential privacy is its immunity to post-processing. Once the differentially private dataset has been released, any techniques can be applied to the dataset and this will not affect the privacy guarantees. Of course the caveat is that information from the original dataset may not be used. A third party cannot reduce the privacy loss even with access to auxiliary information. This bullet-proof definition allows full use of the differentially private dataset from the release of statistics to its use in training machine learning models.

Definition 2.2.5 (Proposition 2.1 [CD14]) *Differential privacy is immune to post-processing.*

Formally, the composition of a data-independent mapping f with an (ϵ)-differentially private algorithm $\mathcal{M}$ is also (ϵ)-differentially private.

2.2.2 Limitations of Differential Privacy

Differential privacy was a consideration during the COVID 19 pandemic for contact tracing [PT20]. The goal of contact tracing applications is to identify and notify individuals that were located in the same area as a COVID positive individual. Differential privacy naturally was a topic of conversation to protect the locations of users. It can be used as an alerting system where individuals locations are privatised and they receive a notification when they have entered a location in which there was a Covid positive case. There have been recent publications that concentrate on the application of differential privacy for contact tracing [RLAW24]. These privacy guarantees would apply to the population at large. However once an individual has

been identified as COVID positive, individuals must waive their right to privacy in order for the safety and protection of others. This an example of the limitations of differential privacy. It cannot be used for these Covid positive individuals as their exact movements need to be tracked. This is the same reason why differential privacy cannot be used for drug trials as exact data about an individual must be understood. When the identification of an individual is required, differential privacy is not achievable.

To summarise, Differential Privacy provides mathematical guarantees to individuals' privacy even against attacks using auxiliary data. The guarantees can be extended to groups and the use of multiple mechanisms. It provides protection to users over their data and obtains trust. This opens up many avenues of research such as the use of mobility data in transport planning or the improvement of applications using users' locations. There are some limitations for specific use cases such as the identification of individuals. Finally, the released differentially private datasets are immune to post processing allowing improvements on the utility. We will now see two settings in which differential privacy is used.

2.3 Central Differential Privacy

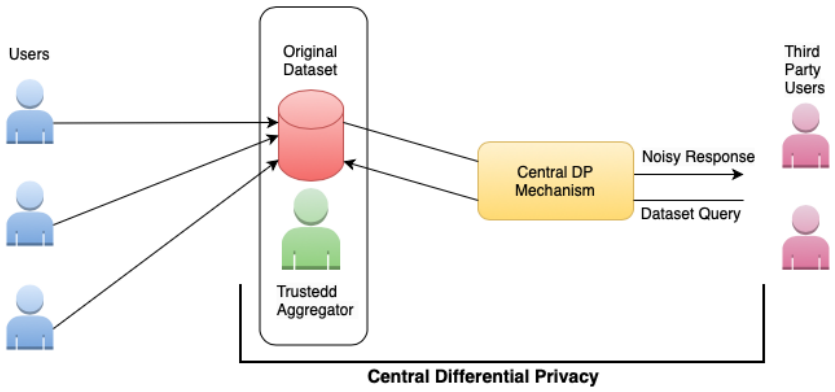


Figure 2.1: Central Differential Privacy

In central differential privacy, the users data is gathered together to create a dataset. There is a trusted aggregator who has access to this original dataset which contains the users true data. When a query is asked on the dataset, the answer to this query is passed through a central DP-Mechanism which carefully applies noise to the query and this noisy answer is released. When multiple queries are asked, additional noise may be required to ensure differential privacy. This is to avoid situations where numerous responses are provided for the same query and the true answer can be deduced. The central DP mechanisms which will be discussed are Laplace Mechanism and Stability Histogram.

2.4 Local Differential Privacy

In the local setting each individual i perturbs their own element x_i by passing it through a locally differentially private (LDP) mechanism $\mathcal{M}$. The output distribution of each individual is ϵ-LDP. This differs from central DP where only the final distribution of the output is required to be ϵ-DP. This additional requirement protects the individual from any central curator having

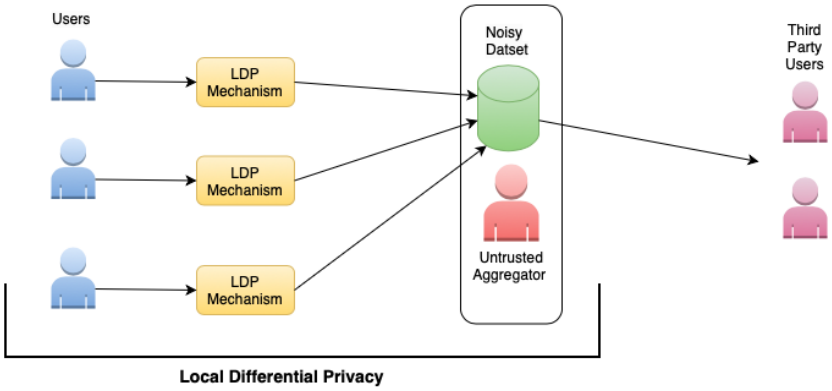


Figure 2.2: Local Differential Privacy

access to the original data. It is a powerful level of privacy as no one will have access to the sensitive data, only the perturbed version of it. The information they can gain about the sensitive data is bounded. This is reassuring for the user especially when the data is sensitive. The drawback is that the accuracy of the estimated dataset $\tilde{D}$ is reduced and answering queries such as frequency estimation is more difficult. However, there are various methods proposed which allow for high utility from the aggregated data and in turn can obtain more accurate frequency estimations. There are a number of LDP mechanisms which will be discussed namely Randomised Response (also known as Direct Encoding), Unary Encoding Optimal Local Hashing and Hadamard Transformation/Encoding.

After the perturbation of an individuals data, the data is randomised and no longer holds much significance alone. It is only after the aggregation across a large number of individuals that LDP can have meaningful utility. The LDP Mechanisms we will discuss follows a 2 step process for each individual namely encoding and perturbation. In the encoding step, the individual encodes the original value into a a format which makes it suitable for the perturbation technique. The encoded value is then perturbed by the randomised algorithm that achieves local differential privacy. The encoding and the perturbation occur locally on individuals devices is known as the 'client side'. Thus it is important to use a suitable LDP mechanisms for the problem. The number of individuals, the size of the data universe and the privacy budget all play important roles for accuracy but of the communication cost must also be a consideration. Reduction of the communication cost is necessary to ensure scalability of mechanisms and the application to real-world problems. Recently, the Hadamard matrices have been applied to create the Hadamard Mechanism, an LDP mechanism with minimal communication cost. This mechanism will be discussed in detail in Chapter 3.

Once the data has been encoded and has undergone perturbation locally, it is sent to a centralised aggregator. This 'server side' process consists of aggregation of the data and also estimation. The aggregator estimates the query results according to the perturbation strategy. In this way, the data can undergo post-processing to ensure the expectation of the query is unbiased. Further post-processing can also be performed to improve estimation accuracy.

2.4.1 Pure Protocol

The notion of a pure protocol for frequency estimation was introduced by Wang et al. [TWJ17]. A pure protocol is defined for LDP mechanisms and counting queries. It introduces the Support function, a function which maps an output to a set of inputs which support it. In short, this set consists of all the inputs that can give a particular output after being run through an LDP mechanism. A protocol is defined as pure if it satisfies two equations for the mechanisms' output with probabilities p^* and q^*. The probability that an input x_i is mapped to an output that supports it is p^*. The second probability q^* is the probability that the input x_j is mapped to an output which supports x_i where $x_i \neq x_j$. The final condition requires that $p^* > q^*$.

Formally this is written as follows where $\mathcal{M}$ is the LDP mechanism applied to x_i, the input for the i-th individual.

$$Pr[\mathcal{M}(x_i) \in \{y | x_i \in \text{Support(y)}\}] = p^*$$

$$\forall_{x_i \neq x_j} Pr[\mathcal{M}(x_j) \in \{y | x_i \in \text{Support(y)}\}] = q^*$$

The state of the art LDP mechanisms that we will study fulfill these requirements and are pure protocols. This will be shown in Chapter 3. This allows for the mechanisms to be generalised and Theorem 1 and 2 below can be applied. In addition to simplifying the analysis, it provides a more coherent comparison between mechanisms. The main benefit of a mechanism being a pure protocol that it allows for optimisation of the parameters. This was outlined by Wang et al. [TWJ17] in which they provided an optimised version of local hashing, OLH. This was found by choosing the optimal probabilities p^* and q^* which minimised the variance of the counting query.

Theorem 1 ([TWJ17]) *For an LDP pure protocol, the expectation of the counting query is unbiased.*

$$\mathbb{E}\left[\frac{q(\tilde{D}, y) - q^*}{p^* - q^*}\right] = q(D, y)$$

Theorem 2 ([TWJ17]) *For an LDP pure protocol, the variance of the estimation of the counting query is:*

$$\text{Var}[q(\tilde{D}, y] = \frac{n \cdot q^* \cdot (1 - q^*)}{(p^* - q^*)^2} + \frac{n \cdot q(D, y) \cdot (1 - p^* - q^*)}{p^* - q^*}$$

2.4.2 Post-Processing

Post-processing on the noisy counting queries after LDP can improve utility. This is an important consideration as LDP mechanisms produce nosier datasets than the traditional centralised DP. This results in the noisy counting queries $\tilde{q}(D, y)$ producing negative frequencies or outputs which do not sum to 1. As differential privacy is immune to port-processing, it is within our best interest to try and improve these results. In their paper 'Locally Differentially Private Frequency Estimation with Consistency' Wang et al. discuss various post-processing techniques for counting queries [WLZL+19]. We will outline the techniques that we will apply.

Base

In this method there is no modification of the estimated frequencies. This results in some of the frequencies taking negative values. We will show in Chapter 3 that the output of the LDP mechanism will produce unbiased estimates. Base preserves this as some of the frequencies will be overestimated and some underestimated.

Base-Pro

A natural step is to round up all negative frequencies to zero. This makes more sense logically for the data and subsequent statistics. This is the technique that Base-Pro will use. However always rounding up introduces a positive bias into the data and the output will no longer be an unbiased estimate. The negative noise is removed but never the positive noise. This solves the issue of negative counts but the frequencies will not sum to 1.

Base-Cut

Base-Cut is a post-processing technique aimed to solve both negative counts and retain the original number of individuals, assumed to be public knowledge. Under this method, the frequency estimations are sorted in decreasing order and their values are kept until their combined value is greater than the number of individuals. At this point, any element that falls under this threshold is set to zero. This is a more time-consuming method in comparison to Base-Pro and may cause issues when the data universe is large.

These post-processing techniques will be applied to each of the LDP methods implemented namely Randomised Response, Unary Encoding, Optimised Local Hashing and the Hadamard Mechanism. We exploit the basic information that the total number of individuals is known and that negative values are not possible. The affect of each of these techniques for our mobility dataset is outlined in Chapter 5.

2.5 State-of-the-Art

There has been plenty of data breaches such as the case of Netflix. In 2018, the European Union passed GDPR to ensure that companies are upheld to a high standard when dealing with personal data of individuals. Companies must be aware and transparent of how they collect, process and store data. They are liable to large fines if they do not adhere to these standards. The Cambridge Analytica scandal among other large data breaches have gained global attention. Multinational companies such as Google, Apple & Meta have invested into differentially private solutions to gather individuals data. Their goal ultimately is to privately obtain statistics and information about their users to improve their product/service while simultaneously maximising utility. Using differential privacy not only protects these companies from legal issues but also garners user's trust with mathematical guarantees that their data is protected.

Building private histograms by applying differential privacy has been studied using various techniques both in the central and the local DP setting. Histograms of individual's data underpins a variety of machine learning and statistical tasks. The focus of this thesis is on mobility data, an extremely important and relevant topic with applications in disease spread modelling, util-

ity management and urban planning [SSND+21]. We first look at recent work and successful deployments of differentially private solutions.

Google was one of the first companies to roll out a differentially private solution to obtain their clients' data. In 2014, they published the paper 'RAPPOR: Randomized Aggregatable Privacy-Preserving Ordinal Response' which detailed their algorithm [EKP14]. RAPPOR is a LDP algorithm that applies direct encoding with Bloom filters. Further details of this algorithm will be discussed in Section 3.2.3. Erlingsson et al. proposed 3 different flavours of RAPPOR; one-time RAPPOR, basic RAPPOR and basic one-time RAPPOR which can be applied in different use cases. They introduced the use of cohorts to facilitate accuracy by the prevention of many collisions in the Bloom filters. Cohorts are assigned different sets of hash functions to use and individuals report their cohort number along with their private data. One of the applications of RAPPOR is to flag malicious software that can infiltrate individuals Chrome browser. To identify these main players, Google collected user's information from their browsers settings. From ~ 14 million reports, they identified the top 31 unexpected Chrome homepage domains.

Apple followed suit in 2016 when they announced the deployment of local differential privacy for part of their user data collection. The various applications include identifying popular emojis, popular health data types, and media playback preferences in Safari. Apple discusses the scalability of their solutions and detail their algorithms on both the client side and the server side in their paper 'Learning with Privacy at Scale' [DPT17]. Their LDP solution uses the Hadamard Mechanism (discussed in Section 3.2.4) along with the count sketch algorithm which finds frequency reported data elements along with accurate counts from a stream. They dub this algorithm the Hadamard Count Mean Sketch where each user only sends one privatised bit. The algorithms' goal is to place the majority of the resource load on the server side. Microsoft is another company which has incorporated differential privacy to gather statistics from users [DKY17].

In regard to the central setting, the US Census traded their data swapping privacy method that had been implemented since 1990 to differential privacy in 2020. This allows the US Census Bureau to publish the noisy data without any increased risk of disclosure. They have praised the transparency of differential privacy to their previous method of data swapping in which geographical identifiers were swapped between households with similar characteristics. By design, the Census Bureau did not release information about its specific details for swapping. This would compromise privacy but this lack of transparency makes it difficult for users to understand the impact of swapping on their data. In 2020, Approximate Differential Privacy was used and the privacy parameters ϵ, δ that were used were published in the US Census Bureau's Disclosure Avoidance Census 2020 Manual [Bur21] which is publicly available.

There is a trade-off between trusting a central aggregator and accuracy. It is a fair statement to assume that individuals at large do not trust private companies such as Apple and Google with their data. In these scenarios, LDP is a good option to broker trust with users while simultaneously obtaining data. However, in the case of a census in which all residents of a country must legally partake, central DP is a better choice. The central aggregator already has access to the data so central DP mechanisms outputting higher accuracy's are preferable.

So far, we have solely discussed central DP and local DP. These settings of differentially private lie at the extremes of trust, either we do not trust the central aggregator or we do. Recently there have been new methodologies that fall between these two levels of trust.

The Shuffle Model is a novel method introduced by Erlingsson et al. [EFM+19] in 2019 to address the problem of monitoring users data over time. Google's RAPPOR among others have addressed this issue but additional noise must be injected to project the user's privacy. Of course, lower accuracy is achieved by mechanisms that defend against such privacy erosion. The

Shuffle Model introduces an intermediate step known as the Shuffler applied in the LDP setting between the client and the aggregator. After the users data has been perturbed using an LDP method, random shuffling is applied before the data is sent to an un-trusted aggregator. This shuffling usually occurs in batches and the users data must be anonymised beforehand so that it is not traceable to a single individual after shuffling. This amplifies the privacy guarantees offered to the users and has inspired a new line of research.

The use of differential privacy to protect users' in federated learning has gained traction in recent years. Under federated learning, batches of individuals locally use their data to collaboratively train a deep learning model. This decentralised approach prevents the storage of all of the training data in a single place but is still susceptible to privacy attacks. IBM has created a framework for federated learning which supports the use of differential privacy [LBT+20]. Recently, Google published a paper detailing how they trained language models with federated learning and differential privacy [ZRX+23]. Their application Gboard has been rolled out on user's devices and provides next word prediction of user's keyboards.

There have been some papers which explore alternatives to the Shuffle Model. Bharadwaj and Comode demonstrate an (ϵ, δ)-DP method for histogram generation in a federated setting via a sampling-based procedure that does not add noise to data [CB22]. They use Poisson sampling where uncertainty is introduced using a small sampling rate. Only counts above a certain threshold which depends on the δ parameter are used. This model sits between the shuffle model and the centralised DP model. The exact results of the sampling are released with no additional noise. This method is unique in that the noise is not added to the data itself but rather the selection of the elements from individuals.

Local Differential Privacy is an important factor in the future development of private and secure federated learning solutions. This thesis applies the state of the art LDP mechanisms to mobility data. The many applications of mobility data along with the proliferation of mobile devices make it an extremely interesting topic. The mobility data studied is acquired from the Irish 2016 census for commuting. There are two hierarchical levels of the dataset and LDP will be applied to both. The first level has many more users than elements resulting in a dense histogram with almost all elements containing counts. The second level shows an extremely sparse histogram with $n \ll |\mathcal{X}|$. The majority of elements have a count of zero. This is a common occurrence when considering all possible perturbations of commutes. The dataset will be explored in detail in Chapter 4.

2.6 OpenDP

The Census 2020, Google's RAPPOR and GBoard have all been successful in utilising differentially private solutions. However, case-by-case solutions are a blocker to companies/institutions that do not have the funds to develop a personalised solution. OpenDP is an open source framework that hopes to bridge the gap and spread differentially private solutions. To quote the developers: "The target use cases for OpenDP are to enable government, industry, and academic institutions to safely and confidently share sensitive data to support scientifically oriented research and exploration in the public interest." [GHV20]

The framework is broken into modular components namely measurements and transformations. It is through these operators that the randomisation is injected into the data. The measurement operator is a mapping from a dataset to an output of an arbitrary type. The measurement operator at a minimum requires as an input the data universe $\mathcal{X}$, the privacy budget ϵ and the function which uses a differentially private mechanism. An example of an appropriate measurement is the addition of Laplace noise to the count. Other measurements that are

supported are the use of Gaussian Mechanism, Stability Histogram and Randomised Response.

The transformation operator is a mapping from a dataset to another dataset. Privacy is not required for a transformation in OpenDP. Examples are selecting a column from the dataset, partitioning the dataset or performing a count on an element in the dataset.

OpenDP includes smart calculations that can determine the scale of the noise that needs to be added for a given privacy budget once the global sensitivity is known. This makes the OpenDP library more inclusive and lowers the entry barrier for the application of differential privacy. The intricacies of differential privacy can be handled by the framework. In this thesis, the Laplace Mechanism and the Stability Histogram, the two central DP mechanisms studied were applied on our dataset using OpenDP.

Chapter 3

Mechanisms & Analysis

There is a focus on both central DP and LDP mechanisms. The goal is to complete a theoretical overview of both the privacy and the utility of each mechanism. Each mechanism is described in full and then a proof confirming that the mechanism is ε-DP/ε-LDP is completed. A private mechanism must also be accurate for it to be usable. Focusing on counting queries, we show that each mechanism is an unbiased estimator for counting queries. In the case of the LDP Mechanisms, the notion of a pure protocol from Wang [TWJ17] is applied. The per-entry error and the maximum error for the counting queries is calculated. In LDP the communication cost to collect each individuals data after the encoding and perturbation is determined. The errors on the counting query along with the communication cost are examined with respect to the number of individuals and the dimension of the histogram.

3.1 Mechanisms to achieve Central Differential Privacy

3.1.1 Laplace Mechanism

According to Dwork and Roth [CD14], the Laplace mechanism works by computing a function f and perturbing each coordinate with noise drawn from the Laplace distribution. In the case of a counting query q(D), the noisy answer $\tilde{q}(D)$is released where the noise added is distributed according to the Laplace distribution, $\tilde{q}(D) = q(D) + \text{noise}$. The scale of the noise added is calibrated according to the sensitivity of the query where the sensitivity used is the ℓ_1-sensitivity. The ℓ_1 sensitivity is the maximum change according to the ℓ_1 norm of the counting query $q : \mathcal{X}^n \times \mathcal{X} \to [0, 1]$ computed in any two neighbouring datasets:

$$\mathrm{GS}_1(\mathbf{h}) = \max_{D \sim D'} ||q(D, y) - q(D', y)||_1$$

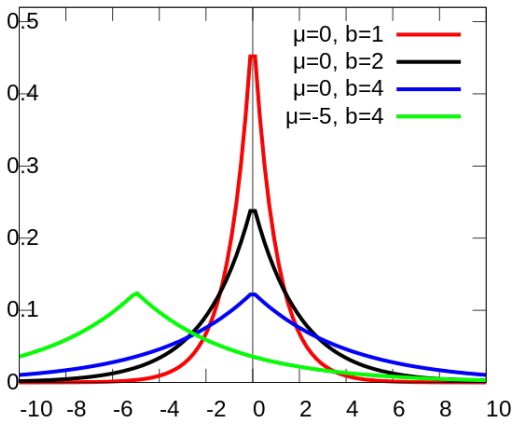
Intuitively this is the uncertainty in the response that is introduced to hide the participation of any individual.

The probability density function of Laplace is

$$p(x) = \frac{1}{2b} \exp\left(-\frac{|x - \sigma|}{b}\right) \tag{3.1}$$

where σ is a mean and b is a scale parameter. This is denoted with $\text{Lap}(\sigma, b)$. In our mechanism, $\sigma = 0$ as the distribution is centered around 0 thus $\text{Lap}(b)$ is used as our notation. The tails in the Laplacian decay exponentially. The Laplace distribution is a symmetric version of the exponential distribution. The variance of the Laplace distribution is $2b^2$.

Figure 3.1: Laplace Probability Distribution



We will consider two neighboring relations for central differential privacy.

- Privacy by Substitution: we say that $D = \{x_i\}_{i=1,\dots,n}$ and $D' = \{x'_i\}_{i=1,\dots,n}$ are neighbor ($D \sim D'$) if exists only one index j such that $x_j \neq x'_j$ (they differ by one row).
- Privacy by Addition/Removal : $D = \{x_i\}_{i=1,\dots,n}$ is neighbor by addition to $D' = \{x_i\}_{i=1,\dots,n} \cup \{q\}$ or by removal to $D' = \{x_i\}_{i=1,\dots,n} \setminus \{q\}$ for some $q \in D$.

The Laplace mechanism adds Laplace noise directly to the histogram query

$$\tilde{\mathbf{h}}(D) = \mathbf{h}(D) + Z \qquad \text{where } Z \sim \texttt{Lap}\left(\frac{\text{GS}_1(\mathbf{h})}{\varepsilon}\right)^{|\mathcal{X}|},$$

where $\text{GS}_1(\mathbf{h})$ is the 1-global sensitivity (under the L_1-norm). In our case the histogram query is the counting query for an element y such that $\mathbf{h}(D) = q(D, y)$ where $y \in \mathcal{X}$ and $q(D, y)$ returns the relative frequency of y in the dataset. To obtain relative frequencies for all elements, $|\mathcal{X}|$-histograms queries must be asked.

Laplace Mechanism is an unbiased estimator of the counting query

The noise added to the histogram query differs depending on which type of 1-global sensitivity that is used. In the analysis, we will consider the 1-global sensitivity by substitution but both are discussed below for completeness.

- **1-global sensitivity by substitution**

  The global sensitivity by substitution of the counting query is $\frac{2}{n}$. This is when there is a substitution of an entry currently belonging to database D with a new entry which may belong to a different element in the data universe. This would cause the counting query $q(D, y_1)$ to differ by $\frac{1}{n}$ and also $q(D, y_2)$ to differ by $\frac{1}{n}$ where $y_1 \neq y_2$ and $y_1, y_2 \in \mathcal{X}$. The histogram query would then clearly differ by a maximum of $\frac{2}{n}$.

- **1-global sensitivity by addition/subtraction**

  Under the assumption that each entry in the database can be mapped to 1 element only, the maximum that an entry can contribute to the counting query q(D,y) is $\frac{1}{n}$. Thus, the global sensitivity by addition/removal of the histogram query $\mathbf{h} : \mathcal{X}^n \to [0,1]^{|\mathcal{X}|}$ is $\frac{1}{n}$.

The noise added to the histogram query using the 1-global sensitivity by substitution is

$$Z \sim \texttt{Lap}\left(\frac{2}{\varepsilon n}\right)^{|\mathcal{X}|}$$

The Laplace Mechanism is an unbiased estimator of the counting query. This can be seen by showing that the expectation of the noisy counting query, $E[\tilde{\mathbf{h}}(D)]$ is equal to the true value of the counting query, $\mathbf{h}$(D). We use the fact that expectations are linear and the expectation of the Laplace probability distribution centered at zero is 0.

$$\begin{aligned} E[\tilde{\mathbf{h}}(D)] &= E[\mathbf{h}(D) + Z] \\ &= E[\mathbf{h}(D)] + E[Z] \\ &= E[\mathbf{h}(D)] + E\left[\texttt{Lap}\left(\frac{2}{\varepsilon n}\right)^{|\mathcal{X}|}\right] \\ &= E[\mathbf{h}(D)] \\ &= \mathbf{h}(D) \end{aligned}$$

Laplace Mechanism is ε-DP mechanism

The Laplace Mechanism is ε-differentially private if the ratio of the probability distributions of the neighbouring datasets differs by e^{ε}. Using privacy by substitution, the scale of noise required is $b = 2/\varepsilon n$.

$$\begin{aligned} p(x) &= \frac{\varepsilon n}{4} \exp\left(-\frac{\varepsilon n|x|}{2}\right) \\ p(x') &= \frac{\varepsilon n}{4} \exp\left(-\frac{\varepsilon n|x'|}{2}\right) \end{aligned}$$

Using the fact that the substitution of random variable x with another random variable x' will affect the counting query by at most 2/n, we can obtain the following,

$$\begin{aligned} \frac{p(x)}{p(x')} &= \exp\left(\frac{\varepsilon n}{2} \cdot (|x'| - |x|)\right) \\ &\leq \exp\left(\frac{\varepsilon n}{2} \cdot \frac{2}{n}\right) \\ &\leq e^{\varepsilon} \end{aligned}$$

Compute the per entry error and max error

If $Z \in \mathbb{R}^{|\mathcal{X}|}$ is a Laplace random variable from Lap(b), then, for every

$$Pr[|Z_i| \geq t] = \exp(-t/b) \tag{3.2}$$

Thus only looking at a singular counting query where noise corresponding to $\texttt{Lap}\left(\frac{2}{\varepsilon n}\right)$ is added and using $t = b \cdot \log(1/\gamma)$ the following can be obtained

$$Pr\left[|Z_i| \geq \frac{2}{\varepsilon n} \cdot \log(1/\gamma)\right] = \gamma$$

The per entry error is

$$|\tilde{q}(D, y) - q(D, y)| = |q(D, y) + Z_i - q(D, y)| = |Z_i| \quad \text{for} \quad y \in \mathcal{X}.$$

Using these two facts

$$|\tilde{q}(D, y) - q(D, y)| \leq \mathcal{O}\left(\frac{1}{\varepsilon n}\right)$$

To obtain the max error, we need to consider that Laplace noise will be added to each of the histogram queries. As there are $|\mathcal{X}|$ queries corresponding to the number of elements in the data universe, the Laplace noise is added $|\mathcal{X}|$ times, once per query.

$$\tilde{\mathbf{h}}(D) = \mathbf{h}(D) + Z = (\big(q(D, y_1) + \text{Lap}(2/\varepsilon n), \ldots, q(D, y_{|\mathcal{X}|}) + \text{Lap}(2/\varepsilon n)\big))$$

The probability that at least one error is greater than t is upper bounded by the sum of all errors using a standard union bound.

$$\Pr\left[\bigcup_{i=1}^{|\mathcal{X}|} |Z_i| \geq t\right] \leq \sum_{i=1}^{|\mathcal{X}|} \Pr\left[|Z_i| \geq t\right]$$

$$\sum_{i=1}^{|\mathcal{X}|} \Pr[|Z_i| \geq t] = |\mathcal{X}| \exp(-t/b)$$

$$\sum_{i=1}^{|\mathcal{X}|} \Pr\left[|Z| \geq \frac{2}{\varepsilon n} \cdot \log\left(\frac{|\mathcal{X}|}{\gamma}\right)\right] = \gamma$$

Thus the maximum error L_∞ of the histogram query is

$$\max_{y \in \mathcal{X}} |\tilde{q}(D, y) - q(D, y)| \leq \mathcal{O}\left(\frac{\log(|\mathcal{X}|)}{\varepsilon n}\right)$$

3.1.2 Stability Histogram

Bun et al. asserts the notion of Stability histograms which is another type of central DP mechanism used for counting queries [BNS15]. This mechanism differs from the Laplace Mechanism as the Stability Histogram satisfies approximate differential privacy rather than pure differential privacy. A new parameter is introduced for this type of differential privacy, δ, which adds an additive factor to the privacy definition.

In Stability Histograms, $\delta \in (0, 1/n)$ [Vad17]. This δ is chosen as δ is a worst case probability to not satisfy DP and we want to protect every individual equally and with high probability. When $\delta \ll 1/n$ is chosen, every user has a constant probability to be (ε, δ)-DP-protected. This can be seen using the union bound.

$$\sum_{i=1}^{n} \delta \ll \frac{n}{n} \ll 1$$

This means that each user is protected with probability $1 - n/\delta \gg 0$.

Stability histograms add noise to counting queries based on the Laplace distribution. However, there are some additional pre and post processing steps designed to improve the error of the noisy counting queries. Firstly, the Stability Histogram adds Laplace noise only to elements that have a positive count. Then there is a threshold of $2 \cdot \frac{\log(2/\delta)}{\varepsilon n} + \frac{1}{n}$ where only counts greater than this are released. Thus the noisy counting queries will only return elements which have a 'high' value of noisy counts (at a minimum, the noisy count will be equal to the threshold value). This is summarised in the points below.

1. If count is 0 - no noise added - output is 0
2. Otherwise add independent noise which is distributed according to Lap $\left(\frac{2}{\varepsilon n}\right)$ to the result of each query
   (a) noisy count $\geq 2 \cdot \frac{\log(2/\delta)}{\varepsilon n} + \frac{1}{n}$ then output noisy count
   (b) noisy count $< 2 \cdot \frac{\log(2/\delta)}{\varepsilon n} + \frac{1}{n}$ then reduce count to 0

In 2020, an optimal Stability Histogram algorithm was developed for the problem of partition selection. [DVGM20]. They showed that a truncated version of Lap$(2/\varepsilon n)$ is optimal for $(1/n)$-sensitive queries. This problem formally known as the differentially private set union focuses on maximising utility from an per-partition aggregation. This optimal Stability Histogram is used to maximise the number of partitions to be released from a dataset for aggregation. Note that privacy by addition/removal is used and each individual is associated with a single partition.

Stability Histogram is (ε, δ)-DP

The Stability Histogram is (ε, δ)-DP if the ratio of the probability distributions of neighbouring datasets differ by e^{ε} and an additive factor δ. Privacy by substitution is used and the Laplace noise added to the histogram queries is Lap$(2/\varepsilon n)$. There are 2 cases to consider when proving the privacy of the stability histogram. The first is that the noisy query outputs zero for D but a count is output for D'. The second case is when a noisy count is output for both $(D \sim D')$. As the datasets differ by one row, the queries will differ for two elements in the histogram, all other elements will be identically distributed.

We consider $D \sim D'$ meaning $D = \{x_1, \ldots, x_i, \ldots, x_n\}$ and $D' = \{x_1, \ldots, x_j, \ldots, x_n\}$ with $x_i \neq x_j$. We are again using privacy by substitution for our analysis. In the first scenario, a noisy count is output for element x_i for a query on D' but outputs a count of zero for D.

$$q(D, x_i) = 0 \quad \& \quad q(D', x_i) \neq 0$$

We can find the probability that this bad event will happen.

$$\Pr[\tilde{q}(D, x_i) = 0] = 0 \quad \& \quad \Pr\left[\tilde{q}(D', x_i) \geq \frac{2\log(2/\delta)}{\varepsilon n}\right]$$

Now lets look at the scenario in which both datasets have non-zero counts for an element but the Stability Histogram outputs 0 for D but not D'. If an element x_i has a non-zero count in D then it contributes to the relative frequency by $1/n$ in D'. However, after the Stability Histogram, in order for this element to contribute to the output, the Laplace noise added to the element plus the original counting query must be greater than the threshold.

$$q(D, x_i) \neq 0 \quad \& \quad q(D', x_i) \neq 0$$

$$\Pr\left[\tilde{q}(D, x_i) = 0\right] = \Pr\left[q(D, x_i) + \text{Lap}\left(\frac{2}{\varepsilon n}\right) \leq \frac{2\log(2/\delta)}{\varepsilon n}\right]$$

To solve for the probability, we use the Laplace bound 3.2

$$tb = 2 \cdot \frac{\ln(2/\delta)}{\varepsilon n}$$

$$t \cdot \frac{2}{\varepsilon n} = 2 \cdot \frac{\ln(2/\delta)}{\varepsilon n}$$

$$t = \log(2/\delta)$$

Substituting this in to obtain

$$\Pr\left[q(D, x_i) + \text{Lap}\left(\frac{2}{\varepsilon n}\right) \leq 2 \cdot \frac{\ln(2/\delta)}{\varepsilon n}\right] = \left(\frac{2}{\delta}\right)^{-1}$$

$$\Pr\left[q(D, x_i) + \text{Lap}\left(\frac{2}{\varepsilon n}\right) \leq 2 \cdot \frac{\ln(2/\delta)}{\varepsilon n}\right] = \left(\frac{\delta}{2}\right)$$

$$\Pr\left[q(D, x_i) + \text{Lap}\left(\frac{2}{\varepsilon n}\right) \leq 2 \cdot \frac{\ln(2/\delta)}{\varepsilon n}\right] = \frac{\delta}{2}$$

The probability that this bad event happens is symmetrical, therefore in total a bad event happens with probability $\frac{\delta}{2} + \frac{\delta}{2} = \delta$. If this happens, the adversary will learn some information present in D' that they would not have learned from D corresponding to the information of an individual. Thus, the Stability Histogram will output a non-differentially private output with probability δ.

Thus the neighbouring datasets can differ by an additive term of δ. The final scenario to consider is when noisy counts are output for both ($D \sim D'$). This scenario is the same as the proof that shows the Laplace Mechanism is ε-DP. The first scenario is $(0, \delta)$-DP, the second is $(\varepsilon, 0)$-DP. Putting these together, the stability histogram is (ε, δ)-DP.

Compute the per entry error and max error

In stability histograms, there are 3 scenarios to consider for the error. The first is when the true count is zero, no noise is added and thus there is no error. In the remaining two scenarios, Laplace noise is added for each element in the histogram. The threshold has an upper bound of

$$2 \cdot \frac{\log(2/\delta)}{\varepsilon n} + \frac{1}{n} = \mathcal{O}\left(\frac{\log(1/\delta)}{\varepsilon n}\right)$$

The per-entry error of the histogram query for released counts is the same as the per-entry error of the Laplace Mechanism.

$$|\tilde{q}(D, y) - q(D, y)| \leq \mathcal{O}\left(\frac{1}{\varepsilon n}\right)$$

The noisy queries which output 0 may have come from an element frequency count that fell below the threshold after Laplace noise was added but had an original value above the threshold. Thus overall, the per entry error of the Stability Histogram is

$$|\tilde{q}(D, y) - q(D, y)| \leq \mathcal{O}\left(\frac{1}{\varepsilon n} + \frac{\log(1/\delta)}{\varepsilon n}\right) \leq \mathcal{O}\left(\frac{\log(1/\delta)}{\varepsilon n}\right)$$

As we have removed all elements with a count of 0, the maximum error is no longer dependent on the size of the data universe, $|\mathcal{X}|$ but instead the number of non empty elements which is at most equal to the number of individuals n. Thus the maximum error L_∞ of the histogram query for released counts is

$$\max_{y \in \mathcal{X}} |\tilde{q}(D, y) - q(D, y)| \leq \mathcal{O}\left(\frac{\log(n)}{\varepsilon n}\right)$$

The maximum error L_∞ of the histogram query for all counts is

$$\max_{y \in \mathcal{X}} |\tilde{q}(D, y) - q(D, y)| \leq \mathcal{O}\left(\frac{\log(n)}{\varepsilon n} + \frac{\log(1/\delta)}{\varepsilon n}\right) \leq \mathcal{O}\left(\frac{\log(1/\delta)}{\varepsilon n}\right)$$

This is because $\delta << \frac{1}{n}$ and so

$$\log\left(\frac{1}{\delta}\right) >> \log(n)$$

This error does not depend on the size of the universe. This is beneficial for very sparse datasets where the number of individuals is much less than the size of the universe. The trade off is that this mechanism is not a ε-DP mechanism but (ε, δ)-DP.

3.2 Mechanisms to achieve Local Differential Privacy

We have seen that the central DP methods have errors that depend on $\mathcal{O}\left(\frac{1}{\varepsilon n}\right)$. We will see that LDP mechanisms have a higher error $\mathcal{O}\left(\frac{1}{\varepsilon\sqrt{n}}\right)$. We now show how each LDP mechanism applied to each individual causes this increase in error. The following Chernoff-Hoeffding bound will be used to analyse the errors.

Theorem 3 (Chernoff-Hoeffding for bounded random variables ([DP09] Exercise 5.3)) *If $X = X_1, \ldots, X_n$ is the sum of n independent random variables with $a_i \leq X_i \leq b_i$ for each $i \in [n]$, then*

$$\Pr[|X - \mathbb{E}[X]| > t] \leq 2e^{-\frac{2t^2}{\sum_{i=1}^{n}(b_i - a_i)^2}}$$

3.2.1 Randomised Response

Randomised Response was a method developed for statistical surveys to allow individuals to answer potentially damaging or revealing questions [War65]. The aggregator does not care about each individuals answer but rather the answer of the population as a whole. These aggregate queries allow for the estimation of the number of individuals which have have a certain property. The addition of random noise in a careful way allows both the individuals to retain their privacy through plausible deniability while simultaneously giving the curator accuracy. Randomised Response is a type of local differential privacy as only the individual knows their true value. There are two probabilities p and q. The user's data remains unchanged in the noisy database with probability p. However the users value can be changed from their original response with probability q.

Randomised Response is traditionally used with 2 possible values for the output. When extended to include many possible elements, it is commonly known as Generalised RR. Randomised Response is not differentially private unless correct probabilities are chosen, thus at the beginning of the analysis the mechanism is denoted as $\texttt{RR}_\varepsilon$ clarifying that we are discussing $\texttt{RR}$ that is ε-LDP.

The randomized response $\texttt{RR}_\varepsilon : \mathcal{X} \to \mathcal{X}$ is defined as follows

$$\Pr[\texttt{RR}_\varepsilon(x) = y] = \begin{cases} p & \text{if } x = y \\ q & \text{otherwise} \end{cases}.$$

The randomized response is applied on each entry of the dataset $\tilde{D} = \{\texttt{RR}_\varepsilon(x_i)\}_{i=1,\ldots,n}$.

Randomized Response is ε-LDP

After the application of RR there are two possible scenarios. The first is that the element in the dataset $\tilde{D}$ was mapped from the same value in the dataset D with probability p.

$$\Pr[\texttt{RR}_\varepsilon(x_i) = y_i | x_i = y_i] = p$$

The second is the probability that the value of an element changes after undergoing RR. With probability q, the value in $\tilde{D}$ changes w.r.t D as it was mapped from another element(category) in the data universe $\mathcal{X}$.

$$\Pr[\texttt{RR}_\varepsilon(x_i) \neq y_i] = (|\mathcal{X}| - 1)[\Pr[\texttt{RR}_\varepsilon(x_i) = y_i | x_i \neq y_i] = (|\mathcal{X}| - 1) \cdot q$$

The probability of these 2 events sum up to 1.

$$\begin{aligned} 1 &= p + (|\mathcal{X}| - 1)q \\ 1 - p &= (|\mathcal{X}| - 1)q \\ q &= \frac{1-p}{|\mathcal{X}| - 1} \end{aligned}$$

In order to satisfy ε-LDP, then $\frac{p}{q} = e^\varepsilon$. Thus the values of p and q can be found.

$$\begin{aligned} &\frac{p}{\frac{1-p}{|\mathcal{X}|-1}} = e^\varepsilon \\ &(|\mathcal{X}| - 1) \cdot p = e^\varepsilon (1 - p) \\ &|\mathcal{X}| \cdot p - p + e^\varepsilon p = e^\varepsilon \\ &p = \frac{e^\varepsilon}{e^\varepsilon + |\mathcal{X}| - 1} \end{aligned}$$

$$\begin{aligned} &(|\mathcal{X}| - 1)q = 1 - \frac{e^\varepsilon}{|\mathcal{X}| - 1 + e^\varepsilon} \\ &q = \frac{1}{e^\varepsilon + |\mathcal{X}| - 1} \end{aligned}$$

The probabilities p and q used in randomized response depend on the value of the privacy parameter ε and also on the size of the data universe.

Find an unbiased estimator of the counting query.

The output of RR is the dataset $\tilde{D} = \{\tilde{x}_i\}_{i,\dots,n}$ where $\tilde{x}_i = \texttt{RR}(x_i)$. There are two probabilities to consider when looking at the counting query $q(\tilde{D}, y)$ with noisy elements. So $\forall y \in \mathcal{X}$,

- The probability that $\tilde{x}_i = y$ and $x_i = y$. This equals p, the probability that RR does not change the input.

- The probability that $\tilde{x}_i = y$ and $x_i \neq y$. This is the probability that RR changes the input to a *particular* output y which is q.

The expectation of the noisy counting query $q(\tilde{D}, y)$ is composed of counting query q(D,y) multiplied by the probability that RR does not change the input plus (1-q(D,y)) multiplied by the probability that RR does change the input.

$$\mathbb{E}[q(\tilde{D}, y)] = q(D, y) \cdot p + (1 - q(D, y)) \cdot q$$

Thus q(D,y) is the expectation of $q(\tilde{D}, y)$ with re-scaling to account for the added noise.

$$\mathbb{E}\left[\frac{q(\tilde{D}, y) - q}{p - q}\right] = q(D, y)$$

$$\mathcal{M} \circ q(\tilde{D}, y) = \frac{q(\tilde{D}, y) - q}{p - q} = \frac{q(\tilde{D}, y) - (\frac{1}{e^{\varepsilon} + |\mathcal{X}| - 1})}{\frac{e^{\varepsilon}}{e^{\varepsilon} + |\mathcal{X}| - 1} - \frac{1}{e^{\varepsilon} + |\mathcal{X}| - 1}}$$

$\mathcal{M} \circ q(\tilde{D}, y)$ will output values in the desired range of [0,1].

Randomised Response is a Pure Protocol

The support function for RR is $\text{Support}_{\texttt{RR}}(y) = y$ as each output value y is supported by an input of y. This is because RR does not encode the data into a different format.

The probability that any value x_i is mapped to x_i after RR is p. This is the same probability that the input value after RR is mapped to an output value that supports it: $p^* = p$.

$$Pr[\texttt{RR}_{\varepsilon}(x_i)\} \in \{y | x_i \in \text{Support}_{\texttt{RR}}(y)\}] = p^*$$

The probability that an input $x_j \neq x_i$ is mapped to x_i's support set is $q^* = q$. This is simply because there is a probability q that the users input value is changed to a different output.

$$\forall_{x_i \neq x_j} Pr[\texttt{RR}_{\varepsilon}(x_j)\} \in \{y | x_i \in \text{Support}_{\texttt{RR}}(y)\}] = q^*$$

Thus RR is a pure LDP protocol with $q^* < p^*$ ie. $\frac{1}{e^{\varepsilon} + |\mathcal{X}| - 1} < \frac{e^{\varepsilon}}{e^{\varepsilon} + |\mathcal{X}| - 1}$. This confirms the analysis above that the mechanism is an unbiased estimator of the counting query by Theorem 1.

Compute the per entry error and the max error

Using the fact that each entry, X_i to the database $\tilde{D}$ is an independent random variable due to the fact that it has undergone RR. Each entry contributes to the counting query with a value of

0 or $\frac{1}{n}$. The following calculation can be made where X is the counting query on the database $\tilde{D}$ and X is the sum of n independent random variables with $0 \leq X_i \leq \frac{1}{n}$.

$$\Pr[|X - \mathbb{E}[X]| > t] \leq 2e^{-\frac{2t^2}{\sum_{i=1}^{n}\left(\frac{1}{n}-0\right)^2}}$$

$$\Pr[|X - \mathbb{E}[X]| > t] \leq 2e^{-\frac{2t^2}{\left(\frac{n}{n^2}\right)}}$$

$$\Pr[|X - \mathbb{E}[X]| > t] \leq 2e^{-2t^2 n}$$

Substituting $X = q(\tilde{D}, y)$ and $\mathbb{E}[q(\tilde{D}, y)] = q(D, y) \cdot (p - q) + q$ to obtain the following

$$\Pr[|q(\tilde{D}, y) - q(D, y) \cdot (p - q) - q| > t] \leq 2e^{-2t^2 n}$$

$$\Pr\left[\left|\frac{q(\tilde{D}, y) - q}{p - q} - q(D, y)\right| > \frac{t}{p - q}\right] \leq 2e^{-2t^2 n}$$

Let $t = \sqrt{\log(2/\gamma)/2n}$

$$\Pr\left[\left|\frac{q(\tilde{D}, y) - q}{p - q} - q(D, y)\right| > \frac{\sqrt{\log(2/\gamma)}}{\sqrt{2n} \cdot (p - q)}\right] \leq \gamma$$

Using the fact that $p - q = \frac{e^{\varepsilon}-1}{e^{\varepsilon}+|\mathcal{X}|-1}$

$$\Pr\left[\left|\frac{q(\tilde{D}, y) - q}{p - q} - q(D, y)\right| > \frac{\sqrt{\log(2/\gamma)} \cdot (e^{\varepsilon} + |\mathcal{X}| - 1)}{\sqrt{2n} \cdot (e^{\varepsilon} - 1)}\right] \leq \gamma$$

$$\Pr\left[\left|\frac{q(\tilde{D}, y) - q}{p - q} - q(D, y)\right| > \left(\frac{\sqrt{\log(2/\gamma)} \cdot |\mathcal{X}|}{\sqrt{2n} \cdot (e^{\varepsilon} - 1)} + \frac{\sqrt{\log(2/\gamma)} \cdot (e^{\varepsilon} - 1)}{\sqrt{2n} \cdot (e^{\varepsilon} - 1)}\right)\right] \leq \gamma$$

$$\Pr\left[\left|\frac{q(\tilde{D}, y) - q}{p - q} - q(D, y)\right| > \left(\frac{\sqrt{\log(2/\gamma)} \cdot |\mathcal{X}|}{\sqrt{2n} \cdot (e^{\varepsilon} - 1)} + \frac{\sqrt{\log(2/\gamma)}}{\sqrt{2n}}\right)\right] \leq \gamma$$

Finally the per entry error is

$$\left|\mathcal{M} \circ q(\tilde{D}, y) - q(D, y)\right| \leq \mathcal{O}\left(\frac{|\mathcal{X}|}{\varepsilon\sqrt{n}}\right)$$

For the max error of the counting query $\max_{y \in \mathcal{X}} |\tilde{q}(D, y) - q(D, y)|$, it is necessary to look at the error of the counting query for each element in the data universe. Taking a union bound over all of the counting queries to obtain the following

$$\Pr\left[\bigcup_{i=1}^{|\mathcal{X}|}\left|\frac{q(\tilde{D},y)-q}{p-q}-q(D,y)\right|>t\right]\leq\sum_{i=1}^{|\mathcal{X}|}\Pr\left[\left|\frac{q(\tilde{D},y)-q}{p-q}-q(D,y)\right|>t\right]$$

$$\sum_{i=1}^{|\mathcal{X}|}\Pr\left[\left|\frac{q(\tilde{D},y)-q}{p-q}-q(D,y)\right|>t\right]\leq 2|\mathcal{X}|e^{-2t^2n}$$

Let $\gamma=\sqrt{\log(2|\mathcal{X}|/\gamma)/2n}$

$$\sum_{i=1}^{|\mathcal{X}|}\Pr\left[\left|\frac{q(\tilde{D},y)-q}{p-q}-q(D,y)\right|>\left(\frac{\sqrt{\log(2|\mathcal{X}|/\gamma)}\cdot|\mathcal{X}|}{\sqrt{2n}\cdot(e^{\varepsilon}-1)}+\frac{\sqrt{\log(2|\mathcal{X}|/\gamma)}}{\sqrt{2n}}\right)\right]\leq\gamma$$

The max error is

$$\max_{y\in\mathcal{X}}\left|\mathcal{M}\circ q(\tilde{D},y)-q(D,y)\right|\leq\mathcal{O}\left(\frac{|\mathcal{X}|\sqrt{\log(|\mathcal{X}|)}}{\varepsilon\sqrt{n}}\right)$$

The error for the randomised response mechanism $\texttt{RR}_\varepsilon:\mathcal{X}\to\mathcal{X}$ is linearly dependent on the size of the data universe. Thus RR is not suitable for histograms in high dimension such as the case of origin-destination commuting where the dimension of the histogram $|\mathcal{X}|=n^2$.

3.2.2 Unary Encoding

The unary encoding mechanism $\texttt{UE}_p(\texttt{UE}_e(x_i)):\mathcal{X}\to\{0,1\}^{\mathcal{X}}$ is composed of 2 steps, the encoding step $\texttt{UE}_e$ and the perturbation step $\texttt{UE}_p$. The full mechanism comprising of both steps $\texttt{UE}_p(\texttt{UE}_e(x_i))$ can be shorthanded to $\texttt{UE}(x_i)$. In the unary encoding step $\texttt{UE}_e$, one-hot encoding is applied to the input of each individual x_i which transforms the input into a $|\mathcal{X}|$-dimensional vector.

$$\texttt{UE}_e(x_i)=\mathbf{B}=[0,\ldots,0,1,0,,\ldots,0]\text{ where }\mathbf{B}(i)\text{ denotes the }i^{th}\text{ location.}$$

In the perturbation step of unary encoding $\texttt{UE}_p$, each bit in the $|\mathcal{X}|$-dimensional vector undergoes a random perturbation as follows

$$\Pr[\texttt{UE}_p(\mathbf{B}(i)=1)]=\begin{cases}p & \text{if }\mathbf{B}(i)=1\\ q & \text{if }\mathbf{B}(i)=0\end{cases}.$$

The unary encoding mechanism is applied on each entry of the dataset $\tilde{D}=\{\texttt{UE}(x_i)\}_{i=1,\ldots,n}$.

Unary Encoding is ε-LDP

We begin by finding the probabilities of p and q such that the mechanism is ε-LDP (Local differential private). The mechanism must be ε- indistinguishable for any different user $x\sim x'$. For this scenario, the vector $\mathbf{B}$ will differ in two locations.

Given $x_i,x_j\in\{0,\ldots,|\mathcal{X}|\}$ and $x_i\neq x_j$ $\quad\mathbf{B}(i)=1,\mathbf{B}(k)=0\ \forall k\neq i\quad\mathbf{B}'(j)=1,\mathbf{B}'(l)=0\ \forall l\neq j$

$$\texttt{UE}_e(x_i) = \mathbf{B} = [0, \ldots, 0, 1, 0, , \ldots, 0]$$
$$\texttt{UE}_e(x_j) = \mathbf{B}' = [0, \ldots, 0, 0, 1, , \ldots, 0]$$

After the application of UE on neighbouring datasets, there are two possible bits in which the output differs as each bit is flipped independently. The probability that neighbouring datasets yield $\mathbf{B}(i) = 1$ and $\mathbf{B}(j) = 0$ or vice versa should differ by a factor of e^{ε} in order to satisfy ε-LDP.

$$\Pr[\texttt{UE}_p(\mathbf{B}(i)) = 1 | \mathbf{B}(i) = 1] \text{ and } \Pr[\texttt{UE}_p(\mathbf{B}(j)) = 0 | \mathbf{B}(j) = 0] = p \cdot (1 - q)$$

$$\Pr[\texttt{UE}_p(\mathbf{B}(i)) = 1 | \mathbf{B}(i) = 0] \text{ and } \Pr[\texttt{UE}_p(\mathbf{B}(j)) = 0 | \mathbf{B}(j) = 1] = (1 - p) \cdot q$$

In order to satisfy ε-LDP, the following relationship must hold

$$\frac{p \cdot (1 - q)}{q \cdot (1 - p)} = e^{\varepsilon}$$

Optimised Unary Encoding from Wang [TWJ17] found the following optimised values of p and q which minimise the variance of the counting query with respect to q.

$$p = \frac{1}{2} \qquad q = \frac{1}{e^{\varepsilon} + 1}$$

Substituting these probabilities into $\frac{p \cdot (1-q)}{q \cdot (1-p)}$, it is shown that the Optimised Unary Encoding Mechanism is ε-LDP.

$$\frac{\left(\frac{1}{2}\right) \cdot \left(1 - \frac{1}{e^{\varepsilon}+1}\right)}{\left(\frac{1}{e^{\varepsilon}+1}\right) \cdot \left(1 - \frac{1}{2}\right)} = \frac{\left(1 - \frac{1}{e^{\varepsilon}+1}\right)}{\left(\frac{1}{e^{\varepsilon}+1}\right)} = \left(\frac{e^{\varepsilon} + 1 - 1}{e^{\varepsilon} + 1}\right) \cdot \left(\frac{e^{\varepsilon} + 1}{1}\right) = e^{\varepsilon}$$

Note: In the case of Googles' Basic RAPPOR, Unary Encoding is used with the probabilities $p = 0.75$ and $q = 0.25$.

Unary Encoding is a Pure Protocol

The output of UE is the vector $\mathbf{B}$, this vector has a length of $|\mathcal{X}|$ and consists of 0's and 1's. It contributes to the counting query for the i^{th} element if the i^{th} location is 1. Thus the vector $\mathbf{B}$ supports each input whose value in the vector is 1: $\text{Support}(y)_{\texttt{UE}} = \text{Support}(\mathbf{B}) = \{i | \mathbf{B}[i] = 1\}$.

The probability that the value $\texttt{UE}(x_i)$ is mapped to is own support set is p ie. the output of the mechanism is $\mathbf{B}[i] = 1$. The probability that the value $\texttt{UE}(x_j)$ is mapped to is the support set of x_i meaning after the perturbation $\texttt{UE}_p(\mathbf{B}[j]) = \mathbf{B}[i]$ is the probability q. Thus $p^* = p$ and $q^* = q$.

$$\Pr[\texttt{UE}_p(\texttt{UE}_e(x_i)) \in \{y | x_i \in \text{Support(y)}_{\texttt{UE}}\}] = p^*$$

$$\forall_{x_i \neq x_j} \Pr[\texttt{UE}_p(\texttt{UE}_e(x_j)) \in \{y | x_i \in \text{Support(y)}_{\texttt{UE}}\}] = q^*$$

Unary Encoding is a pure protocol with $p^* = p$ and $q^* = q$ and $q > p$ as $\frac{1}{e^{\varepsilon}+1} < \frac{1}{2}$.

Find an unbiased estimator of the counting query

For the same reason we discussed in our RR analysis,

$$\mathbb{E}[q(\tilde{D}, y)] = q(D, y) \cdot p + (1 - q(D, y)) \cdot q$$

Thus q(D,y) is the expectation of $q(\tilde{D}, y)$ with re-scaling to account for the added noise. This unbiased estimator is the same for all pure protocols.

$$\mathbb{E}\left[\frac{q(\tilde{D}, y) - q}{p - q}\right] = q(D, y)$$

$$\mathbb{E}[\mathcal{M} \circ q(\tilde{D}, y)] = q(D, y)$$

Compute the per entry error and the max error.

The per entry error is computed using the Chernoff-Hoeffding for bounded random variables. Using the fact that each entry, X_i to the database $\tilde{D}$ is an independent random variable due to the fact that it has undergone UE. Each entry contributes to the counting query with a value of 0 or $\frac{1}{n}$.The following calculation can be made where X is the counting query on the database $\tilde{D}$ and X is the sum of n independent random variables with $0 \leq X_i \leq \frac{1}{n}$.

$$\Pr[|X - \mathbb{E}[X]| > t] \leq 2e^{-\frac{2t^2}{\sum_{i=1}^{n}\left(\frac{1}{n}-0\right)^2}}$$

$$\Pr[|X - \mathbb{E}[X]| > t] \leq 2e^{-\frac{2t^2}{\left(\frac{n}{n^2}\right)}}$$

$$\Pr[|X - \mathbb{E}[X]| > t] \leq 2e^{-2t^2 n}$$

Substituting $X = q(\tilde{D}, y)$ and $\mathbb{E}[q(\tilde{D}, y)] = q(D, y) \cdot p + (1 - q(D, y)) \cdot q$ to obtain the following

$$\Pr[|q(\tilde{D}, y) - q(D, y) \cdot p + (1 - q(D, y)) \cdot q| > t] \leq 2e^{-2t^2 n}$$

Rearranging and substituting $t = \sqrt{\log(2/\gamma)}/\sqrt{2n}$

$$\Pr\left[\left|\mathcal{M} \circ q(\tilde{D}, y) - q(D, y)\right| > \left(\frac{\sqrt{\log(2/\gamma)}}{\sqrt{2n} \cdot (p - q)}\right)\right] \leq \gamma$$

When p =1/2 and q = $1/(e^{\varepsilon} + 1)$ it simplifies to

$$\left|\mathcal{M} \circ q(\tilde{D}, y) - q(D, y)\right| \leq \mathcal{O}\left(\frac{\sqrt{2} \cdot (e^{\varepsilon} + 1)}{(e^{\varepsilon} - 1)\sqrt{n}}\right)$$

Further simplifying the epsilon terms as below

$$\frac{(e^{\varepsilon}+1)}{(e^{\varepsilon}-1)} < \frac{1}{(e^{\varepsilon}-1)} = \frac{1}{1-e^{-\varepsilon}} = \frac{1}{1-1+O(\varepsilon)} = \frac{1}{O(\varepsilon)}$$

Finally per entry error is

$$\left|\mathcal{M} \circ q(\tilde{D}, y) - q(D, y)\right| \leq \mathcal{O}\left(\frac{1}{\varepsilon\sqrt{n}}\right)$$

Using the union bound as we calculated for Randomised Response, the max error is

$$\max_{y \in \mathcal{X}} \left|\mathcal{M} \circ q(\tilde{D}, y) - q(D, y)\right| \leq \mathcal{O}\left(\frac{\sqrt{\log(|\mathcal{X}|)}}{\varepsilon\sqrt{n}}\right)$$

The error for the unary encoding mechanism $\texttt{UE} : \mathcal{X} \to \{0,1\}^{|\mathcal{X}|}$ has no dependence on the size of the data universe. This is very beneficial as the per entry error will not increase with large data universes. However the communication cost of this mechanism is linearly dependent on $|\mathcal{X}|$ so although the error is advantageous for high dimension histograms, the communication cost makes this method less suitable. To improve the communication cost, the input size can be hashed into a smaller domain and then Direct Encoding can be applied to the hashed values.

3.2.3 Optimised Local Hashing

The OLH Mechanism consists of an encoding step $\texttt{OLH}_e$ and a perturbation step $\texttt{OLH}_p$. Let $\mathcal{H}$ be a universal hash function family such that each $H \in \mathcal{H}$ outputs a value in the domain of size $|g|$ where $2 \leq |g| < |\mathcal{X}|$. In the encoding step $\texttt{OLH}_e(x_i) = (H(x_i), h_i)$, each individual randomly chooses a hash function, their input is hashed which reduces the domain from $|\mathcal{X}| \to |g|$. The hashed value is output along with the hashed function used, h_i. In the perturbation step $\texttt{OLH}_p(H(x_i))$, direct encoding is performed on the hashed value and is output along with the hash function used for each individual.

$$\Pr[\texttt{OLH}_p(H(x_i)) = k] = \begin{cases} p & \text{if } H(x_i) = k \\ q & \text{if } H(x_i) \neq k \end{cases}.$$

where $k \in \{0, \ldots, g\}$. The OLH is applied on each entry of the dataset $\tilde{D} = \{\texttt{OLH}((x_i)\}_{i=1,\ldots,n}$. The probabilities p and q have a dependence on the domain size g. OLH optimises g by minimising the variance with respect to g. The optimal domain size was found to be $1 + e^{\varepsilon}$ [TWJ17].

Optimised Local Hashing is a Pure Protocol

The output of OLH is the hashed value along with the hash function used. The support set is $\text{Support}(y)_{\texttt{OLH}} = \text{Support}((y, H)))$. The support set is equal to all possible input values which "support" the output y. In OLH, this is the set of all possible input values in which H(i) = y. Thus $\text{Support}(y)_{\texttt{OLH}} = \{i | H(i) = y\}$.

In OLH, the hash function H outputs a value in the domain of size $|g|$. Thus there are g possible values that the hashed input can take. The probabilities are dependent on g

$$\Pr[\texttt{OLH}_p(H(x_i)) = k] = \begin{cases} p = \frac{e^\varepsilon}{e^\varepsilon + g - 1} & \text{if } H(x_i) = k \\ q = \frac{1}{e^\varepsilon + g - 1} & \text{if } H(x_i) \neq k \end{cases}.$$

where $k \in \{0, \ldots, g\}$.

The probability that any value x_i is mapped to is own support set is straightforward and occurs with probability $p^* = p$.

$$Pr[\texttt{OLH}_p(\texttt{OLH}_e(x_i)) \in \{y | x_i \in \text{Support(y)}_{\texttt{OLH}}\}] = p^*$$

For a family of hash functions $\mathcal{H} = \{H : \mathcal{X} \to [g]\}$, the probability that two elements $x, y \in \mathcal{X}$ are hashed to the same value is $1/g$ due to the uniform hashing assumption.

$$\Pr[H(x) = H(y)] = \frac{1}{g} \quad \text{where} \quad H \in \mathcal{H}$$

For `OLH`, this probability can be calculated from a combination of the two probabilities p and q.

$$\begin{aligned} q^* &= \frac{1}{g} \cdot p + \frac{g-1}{g} \cdot q \\ &= \frac{1}{g} \cdot (p + (g-1) \cdot q) \\ &= \frac{1}{g} \cdot \left(\frac{e^\varepsilon}{e^\varepsilon + g - 1} + (g-1) \cdot \frac{1}{e^\varepsilon + g - 1} \right) \\ &= \frac{1}{g} \cdot \left(\frac{e^\varepsilon + g - 1}{e^\varepsilon + g - 1} \right) \\ &= \frac{1}{g} \end{aligned}$$

Thus the probability that the output of $\texttt{OLH}(x_j)$ where $x_j \neq x_i$ is mapped to the $x_i's$ support set is

$$\forall_{x_i \neq x_j} Pr[\texttt{OLH}_p(\texttt{OLH}_e(x_j)) \in \{y | x_i \in \text{Support(y)}_{\texttt{OLH}}\}] = q^*$$

Using the optimal domain size $g = 1 + e^\varepsilon$, the values for the probabilities p^* and q^* found by [TWJ17] are

$$p^* = \frac{1}{2} \qquad q^* = \frac{1}{e^\varepsilon + 1}$$

Optimised Local Hashing is a pure protocol with $p^* = p$ and $q^* = \frac{1}{1+e^\varepsilon}$ and $q^* < p^*$.

Optimised Local Hashing is ε-LDP

As `OLH` is a pure protocol, the analysis becomes simple as the work has been carried out demonstrating that `OLH` meets the requirements to be a pure protocol. The condition for ε-LDP for a pure protocol is $\frac{p^*}{q^*} \leq e^{\varepsilon}$.

$$\frac{p^*}{q^*} = \frac{1}{2} \cdot \frac{e^{\varepsilon} + 1}{1} = \frac{1 + e^{\varepsilon}}{2} \leq e^{\varepsilon}$$

Find an unbiased estimator of the counting query

As the `OLH` is pure, we can use Theorem 1 that states that the mechanism is an unbiased estimator of the counting query.

$$\mathbb{E}[\mathcal{M} \circ q(\tilde{D}, y)] = q(D, y)$$

Compute the per entry error and the max error

We have shown above that `OLH` is a pure protocol and thus it is an unbiased estimator for the counting queries. The Unary Encoding is also a pure protocol, the same probabilities p^* and q^* were found. Thus, the per entry error and the max error for `OLH` as the same probabilities for p^* and q^* were used. The per entry error and the max error for `OLH` are

$$\left|\mathcal{M} \circ q(\tilde{D}, y) - q(D, y)\right| \leq \mathcal{O}\left(\frac{1}{\varepsilon\sqrt{n}}\right)$$

$$\max_{y \in \mathcal{X}} \left|\mathcal{M} \circ q(\tilde{D}, y) - q(D, y)\right| \leq \mathcal{O}\left(\frac{\sqrt{\log(|\mathcal{X}|)}}{\varepsilon\sqrt{n}}\right)$$

The communication cost does not depend on the size of the universe. The `OLH` mechanism outputs two values, the hashed value is output which is the domain $\lceil 1 + e^{\varepsilon} \rceil$ and the hash function used for each individuals. The amount of data required to be sent is proportional to the number of users, not to the size of the universe as in `UE`. The communication cost is $\log(n)$ due to the space needed to store a hash function for each individual.

3.2.4 Hadamard Mechanism

The Hadamard Mechanism was introduced by Bassily et al. [BNSGT17] in 2017 as part of their state-of the art algorithm for the identification of heavy hitters. The Hadamard Transform has been applied for the estimation of distributions as discussed by Acharya et al [ASZ19]. Apple have also utilised Hadamard Transforms to develop efficient and scalable LDP solutions [DPT17]. These papers are unified in the sense that Hadamard Transforms have been harnessed to reduce computational complexity and communication cost allowing algorithms to be used in real world cases as seen by Apple.

The Hadamard Mechanism (HM) uses Hadamard Transform which is a generalised class of Fourier Transforms. The Hadamard Transform decomposes an arbitrary input vector into a superposition of Walsh Functions. Let $\mathcal{H}$ the universal hash function family such that each $H \in \mathcal{H}$ outputs a value in the domain of size $|k|$. Let $\mathbf{v}$ be vector of length k. For each user, a hash function is chosen at random from $\mathcal{H}$ and used to set $\mathbf{v}[H(x_i)] = 1$ This forms a one-hot encoded vector $\mathbf{v} = [0, \ldots, 0, 1, 0, , \ldots, 0]$. This sparse vector input is encoded with Hadamard Transform which outputs a vector $\mathbf{w} = H_m\mathbf{v}$. The size of the Hadamard Transform Matrix is $k \times k$ where k must be equal to a power of 2, $k = 2^m$. The 1×1 Hadamard Transform is the identity, $H_1 = [1]$. For $m > 1$, the Hadamard Transform is defined and the 2×2 and 4×4 Hadamard Transform is shown. All values in the Hadamard Transform are -1 or +1.

$$H_m = \begin{bmatrix} H_{m/2} & H_{m/2} \\ H_{m/2} & -H_{m/2} \end{bmatrix}$$

$$H_1 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \qquad H_2 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}$$

The encoding $\texttt{HM}_e(x_i)$ can be implemented using the Fast Walsh-Hadamard Transform (FHWT) which computes the Hadamard transform with computational complexity $\mathcal{O}(m \log(m))$. The FWHT algorithm is a divide and conquer algorithm, it recursively breaks down the Hadamard transform of size n in half.

In the perturbation step, the Hadamard Mechanism samples a value from the Hadamard Transform, $\mathbf{w}$ and applies direct encoding to the result. The random perturbation on the $\mathbf{w}_l$, the l-th bit in the vector is as follows

$$\texttt{HM}_p(\mathbf{w}(l)) = \begin{cases} \mathbf{w}(l)(+1) & w.p \quad p \\ \mathbf{w}(l)(-1) & w.p \quad q \end{cases}$$

where $p = \frac{e^\varepsilon}{e^\varepsilon + 1}$. The bit is flipped with probability $q = \frac{1}{e^\varepsilon + 1}$. The final output of the mechanism is the singular bit w_l, the positional argument l and the hash function used h_i. The size of the Hadamard matrix depends on the parameters h and k where h is the number of hash functions in the Hash family and k equals the output domain of these hash functions.

In summary, the Hadamard Mechanism comprises of an encoding step and a perturbation step and sends a single bit for each user $\texttt{HM}_p(\texttt{HM}_e(x_i)) : \mathcal{X} \rightarrow \{0, 1\}$.

The Hadamard Mechanism is a Pure Protocol

The output y from each user is either +1 or -1. The support set $\text{Support}_{\texttt{HM}}(y) = \text{Support}_{\texttt{HM}}(y, H, l)$. The support function takes an output and maps it to a set of inputs which support it. For the Hadamard Mechanism this is any input i which after being hashed by the hash function H, encoded with the Hadamard Transform H_m and the value at the l-th position equals 1. Formally this can be written as $\text{Support}_{\texttt{HM}}(y) = \{i | H_m(H(i))[l] = y\}$ where the parameter m can be found as k is the domain of the hash function and $k = 2^m$.

The probability that any value x_i is mapped to is own support set is the probability $p^* = p$

$$Pr[\texttt{HM}_p(\texttt{HM}_e(x_i)) \in \{y | x_i \in \text{Support(y)}_{\text{HM}}\}] = p^*$$

As there are two possible outputs y which are +1 and -1, the probability that the input x_j was mapped to x_i's support set is $q^* = p/2 + q/2$.

$$\forall_{x_i \neq x_j} Pr[\texttt{HM}_p(\texttt{HM}_e(x_j)) \in \{y | x_i \in \text{Support(y)}_{\text{HM}}\}] = q^*$$

The probabilities are calculated as $p^* = p$ and $q^* = \frac{1}{2} \cdot p + \frac{1}{2} \cdot q$. Substituting these probabilities in we obtain $q^* = 1/2$.

$$\begin{aligned} q^* &= \frac{1}{2} \cdot p + \frac{1}{2} \cdot q \\ &= \frac{1}{2} \cdot \frac{e^\varepsilon}{e^\varepsilon + 1} + \frac{1}{2} \cdot \frac{1}{e^\varepsilon + 1} \\ &= \frac{1}{2} \cdot \left(\frac{e^\varepsilon + 1}{e^\varepsilon + 1} \right) = \frac{1}{2} \end{aligned}$$

Finally, we check the last condition that $p^* > q^*$ is true for all values of ε.

$$\frac{1}{2} < \frac{e^\varepsilon}{e^\varepsilon + 1}$$

$$0 < \frac{e^\varepsilon}{e^\varepsilon + 1} - \frac{1}{2} = \frac{e^\varepsilon - 1}{2 \cdot (e^\varepsilon + 1)} < \frac{e^\varepsilon - 1}{e^\varepsilon + 1}$$

Hadamard Mechanism is ε-LDP

The Hadamard Mechanism is a pure protocol with $p^* = \frac{e^\varepsilon}{e^\varepsilon + 1}$ and $q^* = 1/2$.

The probability that neighbouring datasets output the same value using HM differs by e^ε satisfying ε-LDP.

$$\frac{p^*}{q^*} = \frac{e^\varepsilon}{e^\varepsilon + 1} \cdot \frac{2}{1} = \frac{2}{1 + e^{-\varepsilon}} \leq e^\varepsilon$$

Find an unbiased estimator of the counting query

As the Hadamard mechanism is pure, we can use Theorem 1 that states that the mechanism is an unbiased estimator of the counting query.

$$\mathbb{E}[\mathcal{M} \circ q(\tilde{D}, y)] = q(D, y)$$

Compute the per entry error and the max error

Starting from the following equation which was calculated for UE using the Chernoff-Hoeffding for bounded random variables. It remains the same with the same assumption that each entry X_i to the database $\tilde{D}$ is an independent random variable due to the LDP Mechanism. The output of the HM is a singular bit which can be mapped to an element with the use of the positional parameter l and the hash function used. This output is mapped back to one element, thus contributing to the counting query with a value of 0 or 1.

$$\Pr\left[\left|\mathcal{M} \circ q(\tilde{D}, y) - q(D, y)\right| > \left(\frac{\sqrt{\log(2/\gamma)}}{\sqrt{2n} \cdot (p-q)}\right)\right] \leq \gamma$$

When $p = e^{\varepsilon}/(e^{\varepsilon}+1)$ and q = 1/2 it simplifies to

$$\left|\mathcal{M} \circ q(\tilde{D}, y) - q(D, y)\right| \leq \mathcal{O}\left(\frac{\sqrt{2} \cdot (e^{\varepsilon}+1)}{\sqrt{n} \cdot (e^{\varepsilon}-1)}\right)$$

Further simplifying the epsilon terms as below

$$\frac{(e^{\varepsilon}+1)}{(e^{\varepsilon}-1)} < \frac{1}{(e^{\varepsilon}-1)} = \frac{1}{1-e^{-\varepsilon}} = \frac{1}{1-1+O(\varepsilon)} = \frac{1}{O(\varepsilon)}$$

Finally per entry error is

$$\left|\mathcal{M} \circ q(\tilde{D}, y) - q(D, y)\right| \leq \mathcal{O}\left(\frac{1}{\varepsilon\sqrt{n}}\right)$$

Using the union bound, the max error is

$$\max_{y \in \mathcal{X}} \left|\mathcal{M} \circ q(\tilde{D}, y) - q(D, y)\right| \leq \mathcal{O}\left(\frac{\sqrt{\log(|\mathcal{X}|)}}{\varepsilon\sqrt{n}}\right)$$

The HM has the same per-entry error and the maximum error as both UE and OLH. However the probabilities p^* and q^* are not the same. Unlike UE and OLH where $p^* = \frac{1}{2}$ and $q^* = \frac{1}{e^{\varepsilon}+1}$, in the Hadamard Mechanism $p^* = \frac{e^{\varepsilon}}{e^{\varepsilon}+1}$ and $q^* = \frac{1}{2}$. Substituting these probabilities into Theorem 2 restated below, it can be found that the variance for HM is higher than the variance for UE and OLH.

Theorem 2 *For an LDP pure protocol, the variance of the estimation of the counting query is:*

$$\mathrm{Var}[q(\tilde{D}, y] = \frac{n \cdot q^* \cdot (1-q^*)}{(p^*-q^*)^2} + \frac{n \cdot q(D, y) \cdot (1-p^*-q^*)}{p^*-q^*}$$

The communication cost for HM is the same as OLH due to the use of a hash function for each individual. There are n hash functions needed to encode the element from n individuals. The output of HM is the singular bit w_l, l and the hash function used. Thus the communication cost is $\mathcal{O}(\log(n))$ as the hash function used is the largest piece of information to send.

3.3 Comparison of Different Mechanisms

Table 3.1 summarises the the central DP mechanisms in relation to the per entry error and the maximum error.

	Laplace	**Stability Histogram**
Per Entry Error	$\mathcal{O}\left(\frac{1}{\varepsilon n}\right)$	$\mathcal{O}\left(\frac{\log(1/\delta)}{\varepsilon n}\right)$
Max Error	$\mathcal{O}\left(\frac{\log(\lvert\mathcal{X}\rvert)}{\varepsilon n}\right)$	$\mathcal{O}\left(\frac{\log(1/\delta)}{\varepsilon n}\right)$

Table 3.1: Comparison of Central DP Mechanisms

The main distinction between the Laplace Mechanism and the Stability Histogram is that the maximum error for the stability histogram is no longer dependent on the size of the data universe $|\mathcal{X}|$. This comes at the cost of an approximate differential privacy rather than pure. The thresholds that Stability Histogram applies reduces elements with a low frequency (a frequency under the threshold) to zero. This could be problematic if information about all elements must be known.

	RR	**UE**	**OLH**	**HM**
Communication Cost	$\mathcal{O}(\log(\lvert\mathcal{X}\rvert))$	$\mathcal{O}(\lvert\mathcal{X}\rvert)$	$\mathcal{O}(\log(n))$	$\mathcal{O}(\log(n))$
Per Entry Error	$\mathcal{O}\left(\frac{\lvert\mathcal{X}\rvert}{\varepsilon\sqrt{n}}\right)$	$\mathcal{O}\left(\frac{1}{\varepsilon\sqrt{n}}\right)$	$\mathcal{O}\left(\frac{1}{\varepsilon\sqrt{n}}\right)$	$\mathcal{O}\left(\frac{1}{\varepsilon\sqrt{n}}\right)$
Max Error	$\mathcal{O}\left(\frac{\lvert\mathcal{X}\rvert\sqrt{\log(\lvert\mathcal{X}\rvert)}}{\varepsilon\sqrt{n}}\right)$	$\mathcal{O}\left(\frac{\sqrt{\log(\lvert\mathcal{X}\rvert)}}{\varepsilon\sqrt{n}}\right)$	$\mathcal{O}\left(\frac{\sqrt{\log(\lvert\mathcal{X}\rvert)}}{\varepsilon\sqrt{n}}\right)$	$\mathcal{O}\left(\frac{\sqrt{\log(\lvert\mathcal{X}\rvert)}}{\varepsilon\sqrt{n}}\right)$

Table 3.2: Comparison of LDP Mechanisms

Table 3.2 summarises the LDP mechanisms in relation to the communication cost, per entry error and the maximum error. This communication cost is an additional factor to consider when selecting an appropriate LDP Mechanism. It is the cost of sending the perturbed data of an individuals to the aggregator. The communication cost for Randomised Response (`RR`) and Unary Encoding (`UE`) is dependent on the size of the data universe $|\mathcal{X}|$. The communication cost for Optimised Local Hashing (`OLH`) and the Hadamard Mechanism (`HM`) is dependent on the number of individuals n. The communication cost for the unary encoding is linear for the size of the data universe which makes it unsuitable if the data universe is large. It is not a good choice to use for datasets where $|\mathcal{X}| \gg n$. In practice, this would require every individual to encode their data into a vector of size $|\mathcal{X}|$ which is not feasible with extremely large data universes.

From Table 3.2 there appears to be no theoretical differences between `OLH` and Hadamard encoding except the difference in variation mentioned previously. This is not the case when the decoding time is considered as both mechanism make use of hash functions. The `HM` is faster to decode as the Hadamard Transform allows for efficient reconstruction of the perturbed data. The decoding time is shown in Table 3.3 [CKO20].

	OLH	**HM**
Decoding Time	$\mathcal{O}(n(\lvert\mathcal{X}\rvert))$	$\mathcal{O}(n + \lvert\mathcal{X}\rvert\log(\lvert\mathcal{X}\rvert))$

Table 3.3: Decoding Time

We will see the impact through the running time discussed in Chapter 5. In `OLH`, there are n hash functions which require n calculations to determine the frequency of the elements. This would be the same for `HM` if a naive approach was taken. However the Hadamard Mechanism can be implemented using FHWT which has computational complexity $\mathcal{O}(m \log(m))$.

The maximum error of the counting query, the L_∞ error of the histogram query is in terms of the privacy budget ε, the number of individuals n and the size of the data universe $|\mathcal{X}|$. The Central DP mechanisms are preferable here due to the stronger dependence on n $\left(\mathcal{O}\left(\frac{1}{n}\right)\right.$ vs. $\left.\mathcal{O}\left(\frac{1}{\sqrt{n}}\right)\right)$. Between the LDP Mechanisms, the `RR`is expected to perform the worst due to its additional linear dependence on $|\mathcal{X}|$.

We confirm these observations empirically in Chapter 5, where we compare the accuracy of these mechanisms on the mobility dataset.

Chapter 4

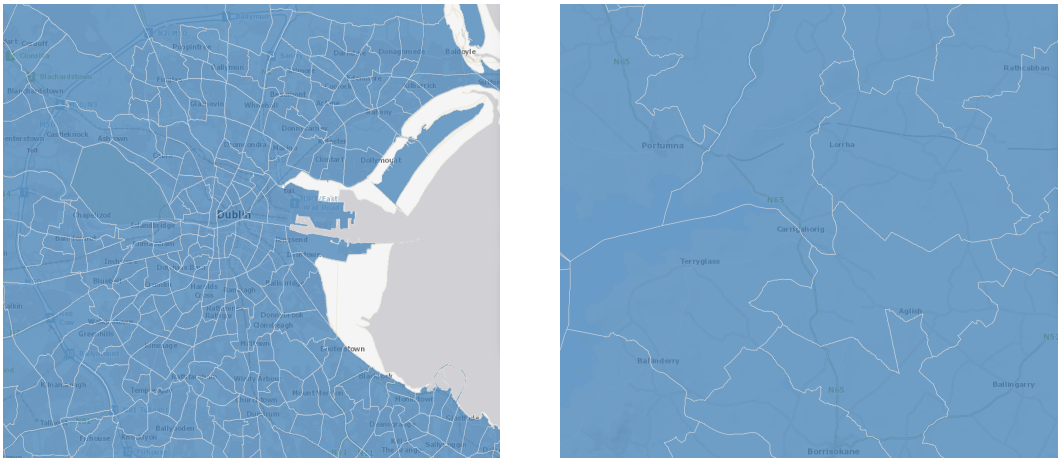
Dataset

The dataset POWSCAR was collected by the Central Statistics Office (CSO) as part of the Irish 2016 Census. [Off16a]. The dataset is known as Place of Work, School or College - Census of Anonymised Records, more commonly known as POWSCAR. A subset of this dataset on commuting is publicly available as a csv file. The following columns will be used for the analysis where the count corresponds to the number of individuals who make a particular journey from one location to another for work, school or college. The data is available for two hierarchical levels; at a County level and at an Electoral Division Level(ED). The start and end location of the journey can be the same. For example, the number of individuals who both reside and work in Maynooth, County Kildare is 4477. Another row from the dataset tells us that the number of individuals that commute from Celbridge, County Kildare to Maynooth, County Kildare is 718. In this case Kildare is the county while Celbridge and Maynooth are the EDs.

Residence CSOED Label	**Residence County Label**	**POWSC CSOED Label**	**POWSC County Label**	**Count**
Maynooth	Kildare	Maynooth	Kildare	4477
Celbridge	Kildare	Maynooth	Kildare	718
Drumcondra South C	Dublin City	Clonskeagh-Belfield	Dún-Laoghaire Rathdown	28

Table 4.1: Commuting Dataset Sample

There are additional columns which will not be used but are present in the dataset namely Residence ED GUID, Residence CSOED, Residence County, POWSC ED GUID, POWSC CSOED and POWSC County. These columns are the unique identification codes and abbreviations for the residence and destination locations.



(a) Dublin City EDs

(b) Irish Midlands EDs

Figure 4.1: Impact of population on size of Electoral Divisions

The public dataset shows the aggregated commuting patterns of individuals at a county level and at an Electoral division (ED) level. There are 3,440 Electoral Divisions (EDs) which are the smallest legally defined administrative areas in the State of Ireland. The CSO implemented additional privacy measures before the public release of the commuting dataset namely,

- Electoral divisions to which fewer than 10 persons commuted have been suppressed.
- Records where no work, school or college was codeable for a worker or student have been removed.

From this point forward, the residence county and residence ED will be known as the origin county and origin ED. An individuals journey between their origin location and their destination location will be referred to as the origin-destination commute.

In the dataset, there are 3,126 EDs present and 31 counties as some counties have been subdivided eg. Dublin has been divided into South Dublin, Dublin City Center, Fingal and Dún-Laoghaire Rathdown. All of the origin locations correspond to an ED and county but this is not the case for the destination locations which have additional values of 'Northern Ireland', 'Overseas', 'No fixed place of work' and 'Work/school from home'. There are 2,750,238 individuals in the dataset. Each individual preforms one commute from one origin to one destination. The first data universe $\mathcal{X}_c$ is the data universe at the county level and is made up of 1085 elements. This was calculated as all the possible permutations of commutes at the county level. In the dataset, there are 1079 commutes that are recorded leaving 6 elements with a count of 0.

Moving onto the ED level which has a drastic increase in the size of the data universe $|\mathcal{X}_e| = 9,796,900$ which corresponds to the total number of possible origin-destination commutes. In the dataset there are 287,116 commutes at the electoral division level which have been carried out by 2,750,238 individuals. This creates a sparse histogram with the majority of the elements having a count of zero.

To visualise the commutes, a flow chart was created using the python library folium. This can be seen in Figure 4.2. To obtain this chart, the latitude and longitude of the most populous part of each county was used ie. Cork city is used to represent the county of Cork. This was more logical than taking the center coordinates of the county itself. The commutes are mapped as

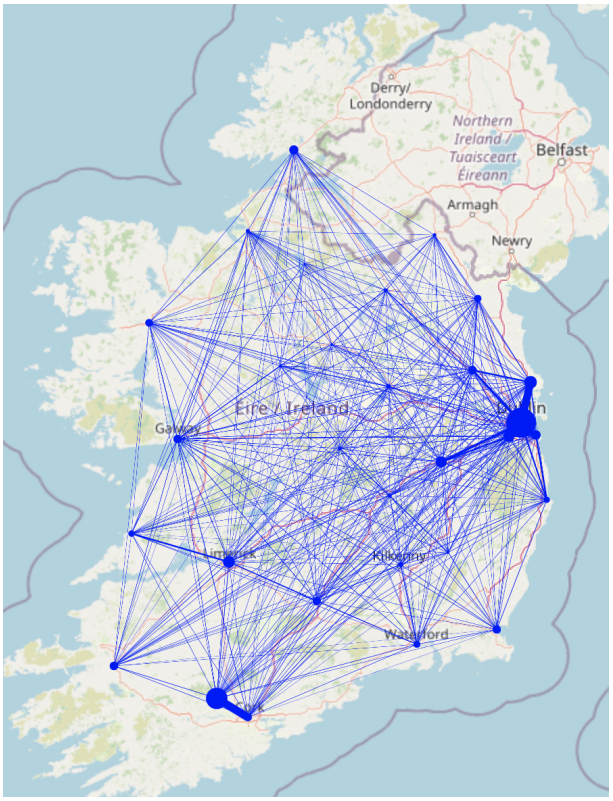
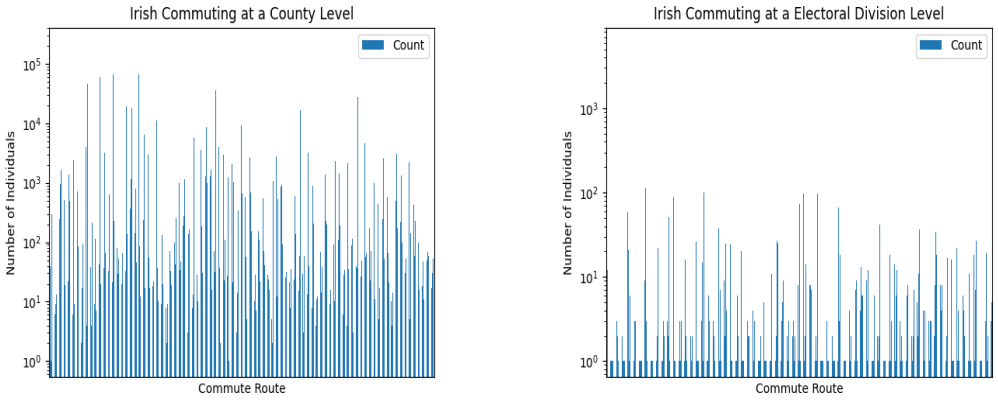


Figure 4.2: Flow Chart of the Commutes at a County Level.

the direct distance between two points. This chart is solely for visualisation purposes as some data is excluded such as a destination 'Overseas' as this could not be represented in the same fashion. This chart provides a deeper understanding of the data. Many of the commutes occur in Dublin, Cork and their surrounding areas.



(a) County Level.

(b) ED Level.

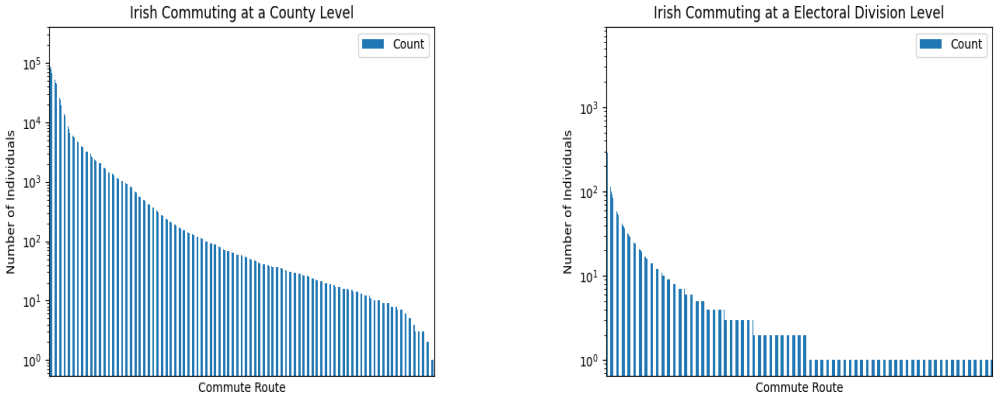
Figure 4.3: Distribution of the Origin-Destination Commutes.

ED Commute	Count
Gortkelly-Ballincollig	1
Fermoy Urban-Clonakilty Urban	1
Granard Urban-Ceannanus Mór(Kells) Rural	1
Portmarnock South-Blackrock-Glenomena	1
Lurgan-Airport	1
Treantaghmucklagh-Na Gleannta	1

Table 4.2: Unique Commutes at an Electoral Division Level

A logarithmic scale was used to view the counts of the origin destination commutes between counties and between EDs. This can be seen in Figure 4.3. Again at the ED level, there are 9,796,900 possible commutes. Of course, due to factors such as distance, time and public transport availability, many of these possible commutes for work, college or school are not feasible. These commutes with a count of 0 are not plotted in the histogram for obvious visualisation reasons. All commutes in the histograms comes from at least one individual. There are 287,116 bins in the ED histogram and 1079 bins in the county histogram.

The dataset was then ordered from the maximum number of counts to the minimum number of counts. This was done in order to clearly view the distribution of the counts which can be seen in Figure 4.4. The distribution of commutes between EDs contains many commutes carried out by a small number of individuals which is to be expected due to the large number of possible commutes.



(a) County Level.

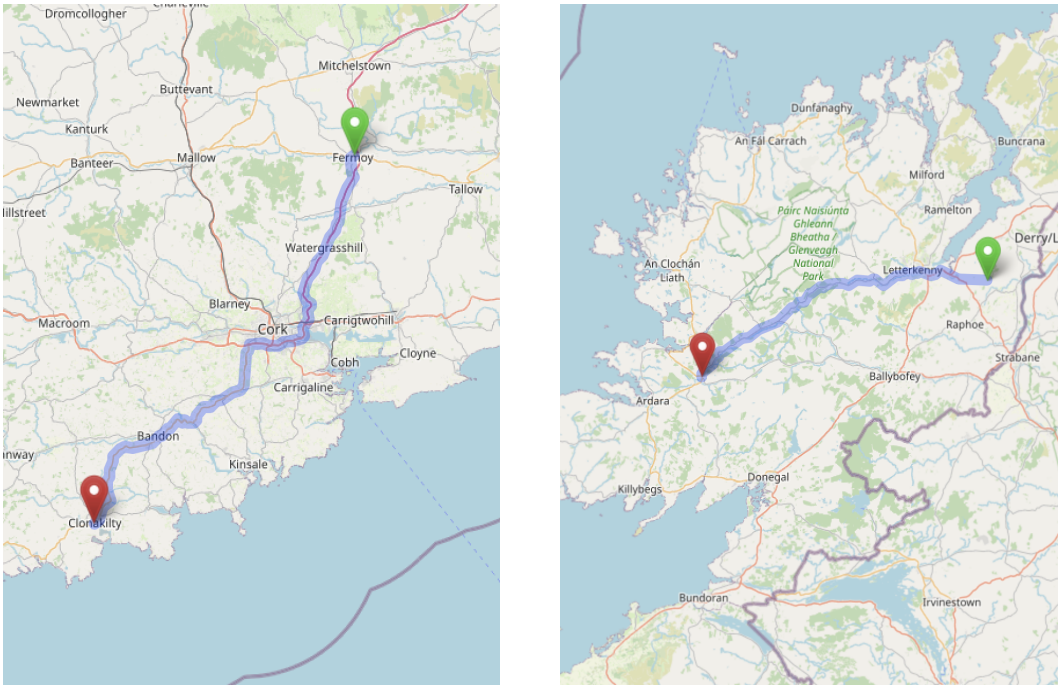
(b) ED Level.

Figure 4.4: Ordered Distribution of Origin-Destination Commutes

Privacy Concerns

Even though some data protection measures were applied to the dataset before its public release, there are some privacy concerns. Currently electoral divisions in which there are fewer than 10 people have been amalgamated with nearby electoral divisions. This ensures that the each origin and destination has a minimum of 10 people. However the same level of data protection was not applied to the commutes. There are 135,771 unique commutes from an origin ED to a

destination ED in the dataset. This leaves 135,771 individuals exposed to the risk of attacks by adversaries. The knowledge of an individuals daily commute for work or college is highly sensitive. A sample of these commutes can be viewed in 4.2. To emphasise the severity of this risk, we show the possible commuting route of 2 individuals in 4.5. Individual A preforms their commute from Fermoy Urban, Co.Cork to Clonakilty Urban, Co. Cork. Individual B commutes from Treantaghmucklagh, Co.Donegal to Na Gleannta,Co.Donegal. This uses the assumption that the individual commuted by car which is reasonable as it is reported that almost 90% of individuals commute by car/van for rural areas. This statistic was released from the same Irish census study [Off16b]. This potential breach of privacy reiterates the need for differential privacy. The differential private mechanisms will now be applied to the dataset and we will investigate how the mechanisms preform on this Commuting dataset.



(a) Fermoy Urban-Clonakilty Urban.

(b) Treantaghmucklagh-Na Gleannta.

Figure 4.5: Examples of Unique Commutes at ED Level.

Chapter 5

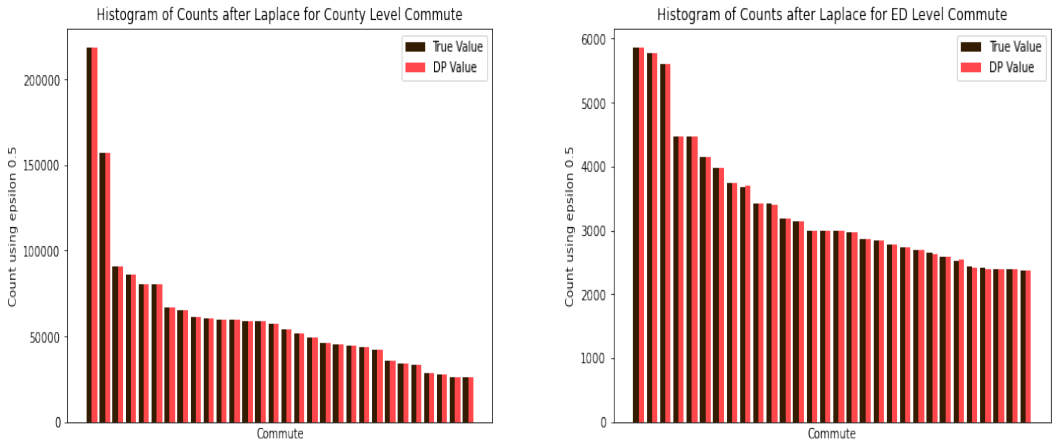
Experiments & Results

An experimental analysis of the discussed mechanisms was carried out using the origin-commuting dataset. The dataset from the CSO contained the commuting journeys at an Electoral Division level and at a County level. Dataset preparation involved moving from an aggregate dataset to a dataset where each row corresponded to an individual. Every possible commute was considered leading to 1085 commutes at the county level and 9,796,900 commutes at the ED level. All experiments were performed in Python 3.9.16 and run on a MacOS Sonoma, with an Intel Core i5 I5-8259U 2.3GHz CPU and 8GB of RAM. Each mechanism was run 10 times to explore the effect that the privacy budget parameter ε has on our dataset. The values of epsilon ranged from 0.5 to 5 in increments of 0.5. A smaller ε adds a higher level of noise leading to stronger privacy results at the risk of nosier counts. The maximum errors and the root mean squared errors were calculated. In order to evaluate the utility, the order that each mechanism placed the commutes was calculated. Lastly, the running time in seconds was compared for different mechanisms. The code is publicly available - https://github.com/avalouisefinnegan/Thesis_DP.

5.1 Central DP Mechanisms

Laplace Mechanism

The OpenDP framework was used to apply the Laplace Mechanism to the dataset. The Laplace noise was added to the histogram queries to produce the noisy histogram: $\tilde{\mathbf{h}}(D)$ which corresponds to adding Laplace noise to 1085 elements at the county level and to ∼9.7 million at the ED level. The Stability Histogram adds $\texttt{Lap}(2/\varepsilon)$ noise to the nonzero elements of the histogram and retains only noisy counting queries that are larger than a threshold value. At the county level, Laplace noise is added to all but 3 elements. However at the ED level, due to the nature of the sparse histogram, the use of the Stability Histogram allows the majority of the elements to be ignored. Laplace noise is added to 287116 elements, 2.5% of the possible commutes taken. Thus the Stability Histogram method is expected to have a much smaller runtime than the Laplace Mechanism at the cost of (ε, δ) DP. After the Stability Histogram is applied, only a subset of counting queries are released as some counts have fallen below the threshold and are treated as zero. To ensure that the output is DP we chose $\delta = \frac{1}{2n}$ which in the desired range, $\delta \in (0, 1/n)$. This was fixed for the entirety of the experiment so that the Stability Histogram could be easily compared against the Laplace Mechanism. The analysis was run for $\varepsilon = (0.5, 1, \ldots, 4.5, 5)$.



(a) County Level

(b) ED Level

Figure 5.1: Histogram of Counts after Laplace Mechanism for $\varepsilon = 0.5$.

In 5.1, a histogram of the top 30 most frequented commutes are shown for $\varepsilon = 0.5$ for both the County Level and the ED Level. It can be seen in the y axis of the graph how the distribution of the levels differ greatly. There are many individuals participating in the same commutes from County-County while the electoral division level is much more granular. There are many elements/categories containing few individuals. The most popular commute from an origin ED to a destination ED is 5871, 0.21% of the total number of individuals. The brown bars correspond to the true count of the original data and the red bars represent the noisy counts which are differentially private. For all released counts, the Stability Histogram has added Laplace noise. Thus the only difference between released counts from the Stability Histogram and the counts from the Laplace Mechanism is the randomness of the noise added. They produce very similar outputs so only the histogram for the Laplace Mechanism is shown. Where these two mechanisms differ greatly is in the elements that have a low count as they will be present in the Laplace but may not be in the Stability Histogram depending on the threshold chosen. Of course, all elements with a count of zero are not present in the Stability Histogram but are in the Laplace Mechanism.

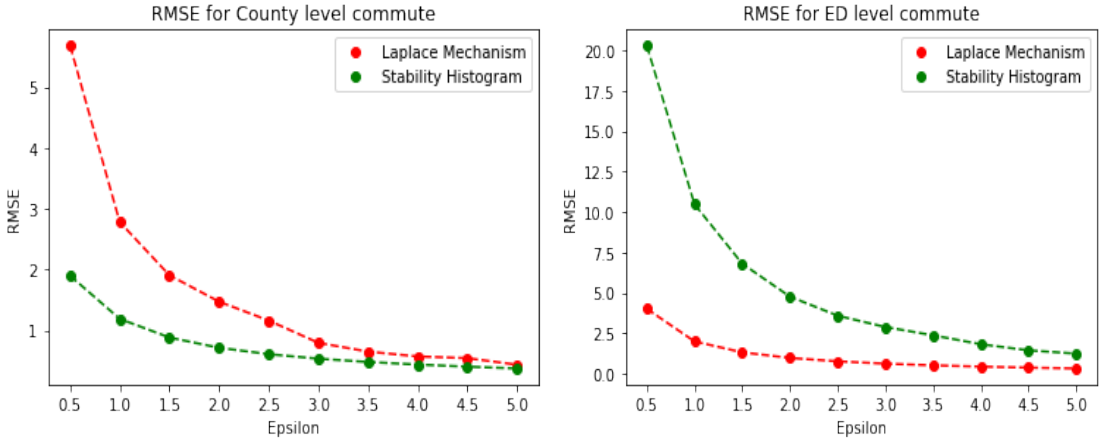
5.1.1 Utility

RMSE

The Root Mean Squared Error(RMSE) was calculated for the total counts. It is plotted for the Laplace Mechanism and the Stability Histogram in Figure 5.2 for both the County Level and the ED level. The Stability Histogram outperforms the Laplace Mechanism at the County Level. The opposite is true at the ED Level with an RMSE for the Laplace Mechanism. There are only 6 elements with a count of 0 at the County level and there is no large benefit of using the Stability Histogram as this introduces a threshold without the computational gains of ignoring many zero count elements.

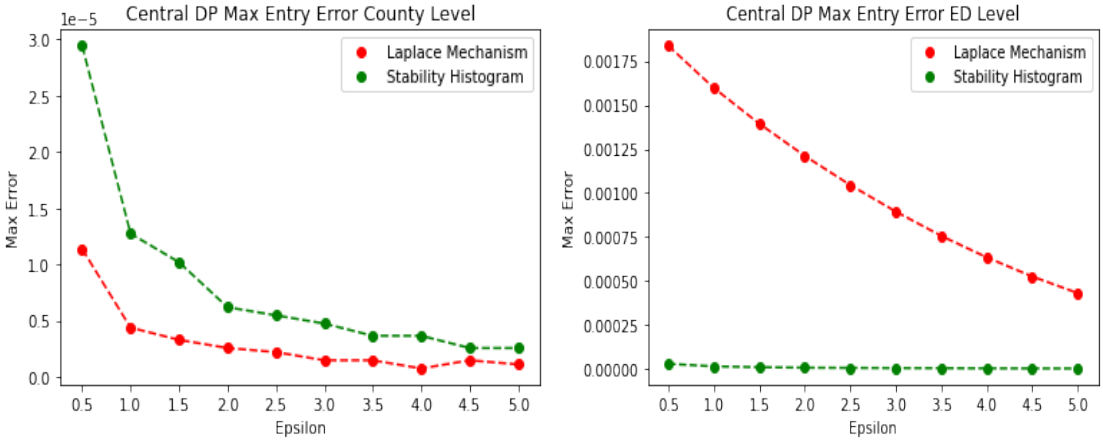
The full potential of the Stability Histogram is used at the ED level as this allows the majority of the elements to be ignored. There are many elements at the ED level with a count of 1 and most of these will be set to 0. We know that these unique commutes pose a threat to privacy and should not have been included in the database in the first place. Many elements with a low count fall under the threshold and this causes the RMSE to be higher. The threshold becomes lower for a higher privacy budget with threshold $= \mathcal{O}\left(\frac{\log(1/\delta)}{\varepsilon n}\right)$. Thus as the value of ε is increased, the RMSE of the Stability Histogram converges towards the RMSE of the Laplace Mechanism.

However the scale of the RMSE is very small and both mechanisms perform well on the dataset and the difference in RMSE is slight. This sentiment will be echoed throughout the analysis, especially when we investigate the LDP mechanisms.



(a) County Level (b) ED Level

Figure 5.2: RMSE for Central DP Mechanisms



(a) County Level (b) ED Level

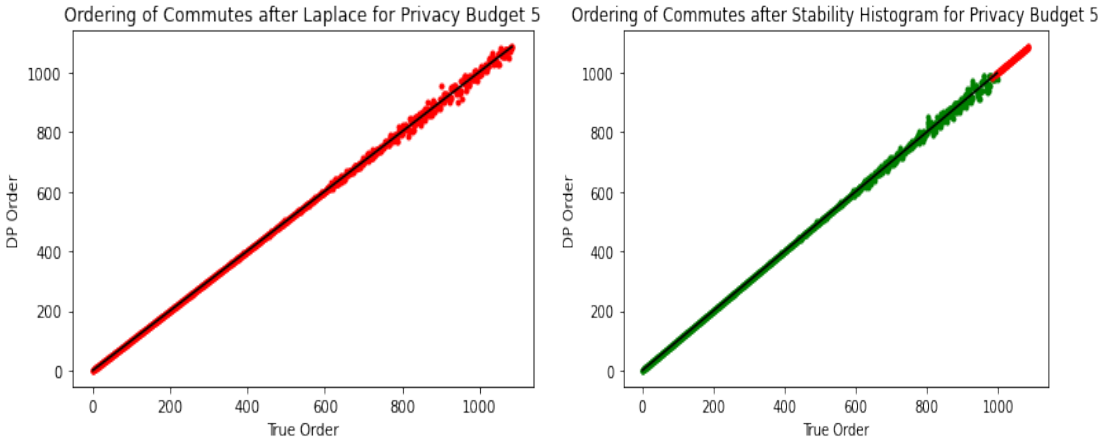
Figure 5.3: Maximum Error for Central DP Mechanisms

Maximum Error

The maximum error is seen in 5.3 which is the maximum difference of a counting query in terms of the relative frequency. The Stability Histogram has a higher max error at the County level and the Laplace Mechanism has a higher max error at the ED level. This confirms the theoretical results above. We saw that max error for the Laplace is dependent on $|\mathcal{X}|$ and the Stability Histogram is dependent on n. As a refresher, the number of individuals in our dataset is $n \approx 2.75$ million, the number of elements at the county level and the ED level are $|\mathcal{X}_c| = 1085$ and $|\mathcal{X}_e| \approx 9.7$ million respectively. The maximum errors become closer in value as the privacy budget is increased. The reason that this is flipped from the RMSE is that the RMSE is an accumulative error over all elements while maximum error is only referring to one element. Thus even though the maximum error is smaller for the Stability Histogram at the ED level, there are 9.7 million elements to consider so millions of small errors can build up more than a few instances of larger errors.

Ordering of the Elements

Another test for the utility is to observe the ordering of the commutes. The ordering of counts is shown for both central DP mechanisms in 5.4 and also can be seen in Tables 5.1 and 5.2. The true order of counts is plotted against the differentially private order, if the mechanism output the true order then the red/green dots would follow black line. The ordering of elements follows the true order almost perfectly behaviour for around the first 600 in both mechanisms before the ordering is slightly altered. Even without the perfect ordering, the mechanism still ranks the commutes in a way that preserves utility. The top commutes retain order while the lowest frequented commutes are still ranked as the least popular. The red dots present in the Stability Histogram graph represent the number of commutes that output a count of 0. This corresponded to 96 commutes. However the utility of the Stability Histogram remains as these 96 elements were the least popular commutes. In the original dataset at the county level, the number of commutes with a true count of 0 was 6. The Laplace Mechanism identified these thus no additional colour highlighting the zero counts was added to the graph.



(a) Laplace Mechanism

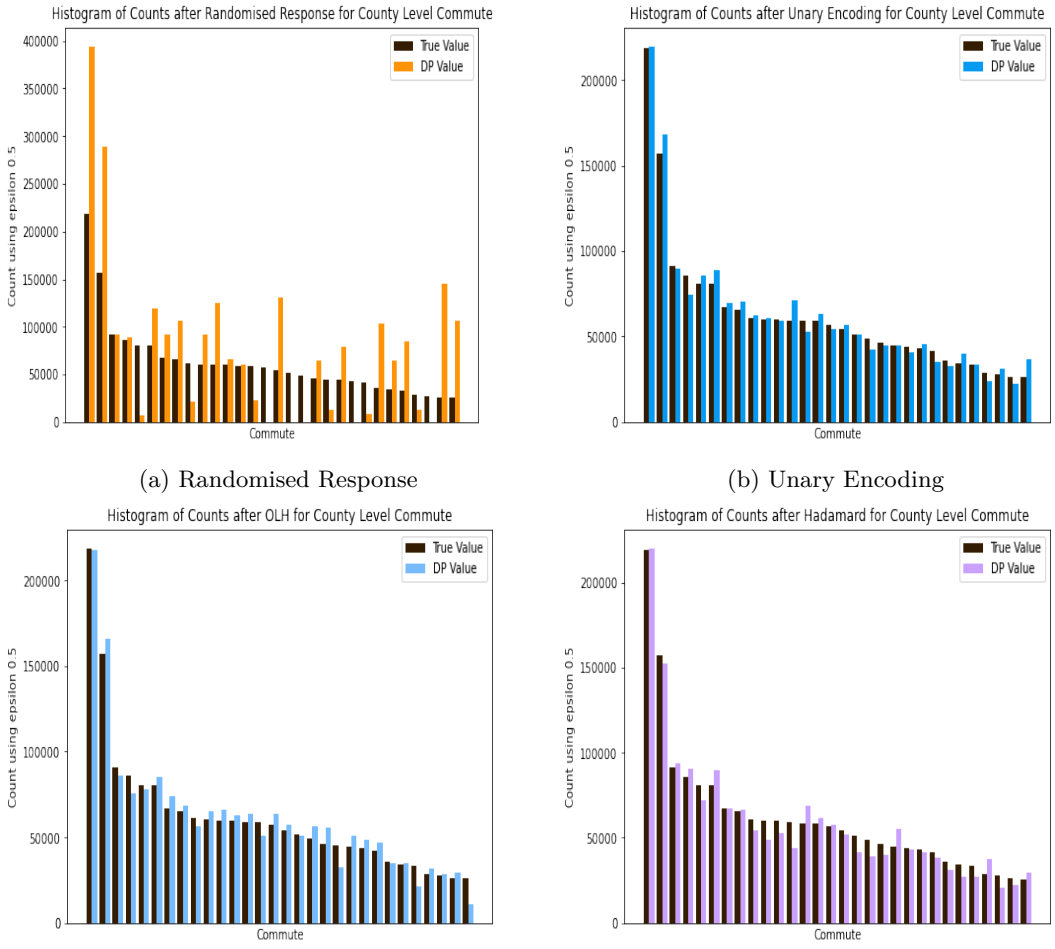
(b) Stability Histogram

Figure 5.4: Ordering for Central DP Mechanisms at County Level with an ε of 5.

5.1.2 Running Times

The running times were recorded in seconds and were the average of 10 iterations. The running time of the mechanisms were the same for the County Level. The Laplace Mechanism ran in 4 seconds and Stability Histogram in 5 seconds. This is expected as both mechanisms added Laplace noise for all of the elements except the 6 elements which had a count of 0. At the ED level, the Stability histogram took 18 seconds, this is not surprising as the number of elements in which Laplace noise must be added has increased. However as most of the elements have 0 counts, the increase in the number of elements to add noise to is minor in comparison to the Laplace Mechanism. The Laplace Mechanism ran in 328.69 seconds. Laplace noise must be added to all 9.7 million elements regardless if the original count is 0 or not leading to this sharp increase in running time.

5.2 LDP Mechanisms - County Level



(a) Randomised Response

(b) Unary Encoding

(c) OLH

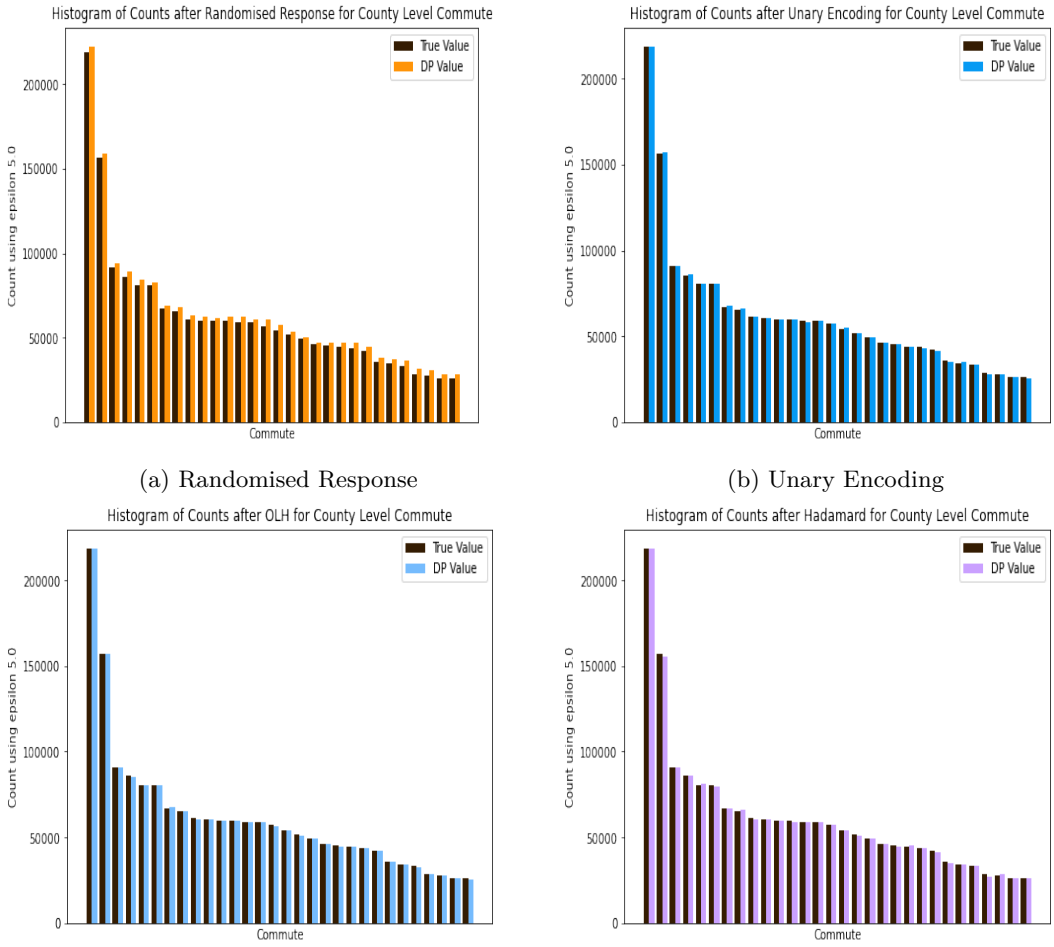
(d) Hadamard Mechanism

Figure 5.5: Histogram of Top 30 Counts with $\varepsilon = 0.5$

As LDP Mechanisms add noise to every individuals' data, the benefits of the additional privacy guarantees are offset by the loss of utility of the data. We analyse these state of the art LDP

mechanisms and investigate their utility on the mobility dataset. The 4 LDP Mechanisms discussed are the Randomised Response(`RR`), Unary Encoding (`UE`), Optimised Local Hashing (`OLH`) and the Hadamard Mechanism (`HM`). Code provided by Cormode et al. written for their paper [CMM21] was applied for `OLH` and the Hadamard Mechanism. We will split our analysis by first applying the mechanisms to the dataset at a county level before moving on to the significantly larger ED level. The Randomised Response Mechanism is expected to perform the worst due to it's linear dependence of the error on $|\mathcal{X}|$. In regards to the communication cost, `UE` is expected to perform the worst which may have a significant impact on the running time of this mechanism.

The first thing to observe is the affect that the privacy budget ε has on the output of the mechanisms. Shown in figures 5.5 and 5.6 are the top 30 commutes for the mechanisms with a privacy budget of 0.5 and 5 respectively. It is clear that an ε of 0.5 produces much nosier counts and the utility is reduced. This is most apparent for `RR` as expected which performs the worst against the other mechanisms. With a privacy budget of 5, there appears to be good utility with the noisy counts roughly following the same order as the senstive data for these 30 most popular commutes.



(a) Randomised Response

(b) Unary Encoding

(c) OLH

(d) Hadamard Mechanism

Figure 5.6: Histogram of Top 30 Counts with $\varepsilon = 5$

5.2.1 Utility

RMSE

The RMSE for all 4 mechanisms is shown in 5.7. Again, `RR` has an RMSE that is much larger than the other mechanisms. The remaining three elements all have similar errors. The RMSE for the LDP mechanisms are significantly larger than the range seen for the central DP levels for all values of ε. Unary Encoding performs the best overall. `OLH` follows the same trajectory until $\varepsilon = 3$, at which point it switches order with the Hadamard Mechanism.

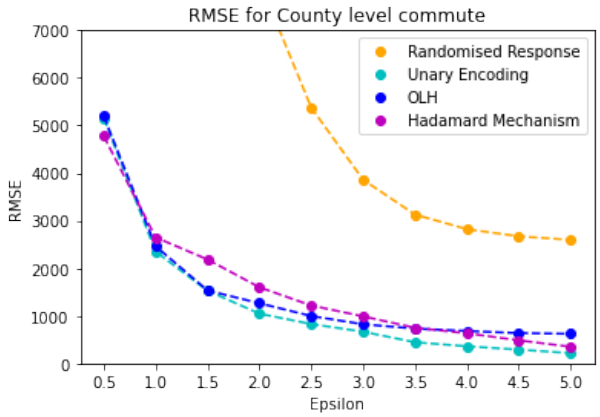


Figure 5.7: RMSE for LDP

Maximum Error

The maximum error calculated in Chapter 3 is the same theoretically for `UE`, `OLH` and `HM` and Figure 5.8 reflects this. The maximum error is in agreement with the RMSE. The maximum error becomes smaller as the privacy budget increases. The decrease in error is steeper for early values of ε and decreases at a slower rate after $\varepsilon = 3$. `OLH` is the only exception to this as it's maximum error does not decrease further.

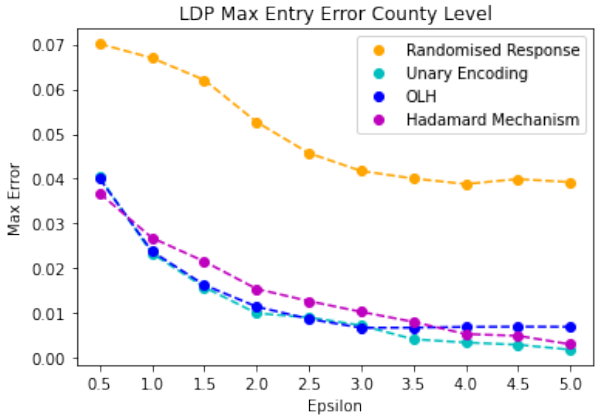


Figure 5.8: Maximum Error for LDP at County Level

County Commutes with a privacy budget of 0.5

Commute	True Count	Lap	Stab H	RR	UE	OLH	Hadamard
Dublin City-Dublin City	1	1	1	1	1	1	1
Cork County-Cork County	2	2	2	2	2	2	2
Fingal-Fingal	3	3	3	172	3	3	3
Limerick City and County x2	4	4	4	188	6	6	4
South Dublin-South Dublin	5	5	5	560	5	5	6
Kildare-Kildare	6	6	6	99	4	4	5
Dún-Laoghaire Rathdown x2	7	7	7	173	9	7	8
Donegal-Donegal	8	8	8	130	8	8	9
Kerry-Kerry	9	9	9	484	11	16	13
Meath-Meath	10	10	10	176	12	10	16

Table 5.1: Utility: Top 10 Commutes at a County Level, $\varepsilon = 0.5$.

County Commutes with a privacy budget of 5

Commute	True Count	Lap	Stab H	RR	UE	OLH	Hadamard
Dublin City-Dublin City	1	1	1	1	1	1	1
Cork County-Cork County	2	2	2	2	2	2	2
Fingal-Fingal	3	3	3	3	3	3	3
Limerick City and County x2	4	4	4	4	4	4	4
South Dublin-South Dublin	5	5	5	5	5	5	5
Kildare-Kildare	6	6	6	6	6	6	6
Dún-Laoghaire Rathdown x2	7	7	7	7	7	7	7
Donegal-Donegal	8	8	8	8	8	8	8
Kerry-Kerry	9	9	9	9	9	9	9
Meath-Meath	10	10	10	11	10	10	10

Table 5.2: Utility: Top 10 Commutes at a County Level, $\varepsilon = 5$.

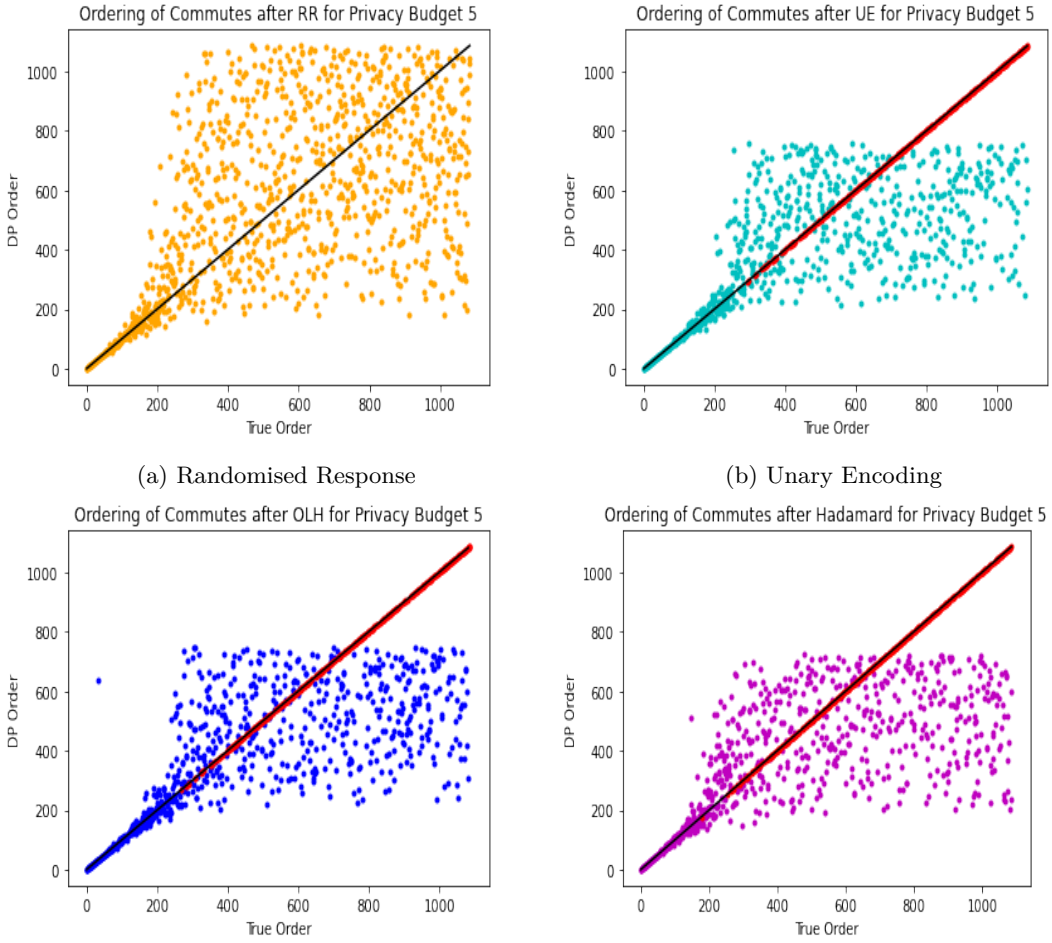
Ordering of the Top 10 Commutes

The ten most frequented commutes are ordered in the Tables 5.1 and 5.2 according to their true count. Each mechanism outputs a differentially private histogram and we are interested if the order of the original elements are preserved. We analyse the accuracy of the DP/LDP mechanisms to maintain the order of the top 10 commutes for a low privacy budget of 0.5 and a higher privacy budget of 5. The number under each mechanism in the table is the order that they placed the $1^{st}, 2^{nd}, 3^{rd}, \ldots, 10^{th}$ element.

The county level corresponds to a dense data universe where the majority of the elements are populated. As expected both of the central DP mechanisms perform well even with a low privacy budget of 0.5. They retain the order of the top 10 commutes. The LDP mechanisms are indeed more unstable than the central DP mechanisms. Randomised Response is particularly sensitive to the change in the privacy budget. It does place the top 10 commutes high up in comparison to the possible 1085 possible commutes but performs poorly with $\varepsilon = 0.5$ when compared to the other LDP mechanisms. `UE`, `OLH` and `HM` all obtain a perfect ordering with $\varepsilon = 5$ while also performing well with $\varepsilon = 0.5$. Randomised Response drastically improves it's utility with the

higher privacy budget. Randomised Response was predicted to have a lower performance than Unary Encoding, `OLH` and the Hadamard Mechanism as it's error does has a linear dependence on the size of the data universe $|\mathcal{X}|$. However it was a surprise that the Randomised Response mechanism was so sensitive to the change in privacy budget. With $\varepsilon = 1.0$, the top ten were ordered in the following way: 1, 2, 15, 27, 6, 28, 73, 22, 399, 64. At $\varepsilon = 1.5$, the order takes another positive leap coming in at: 1, 2, 3, 4, 6, 10, 8, 5, 11, 7. This highlights the larger effect that ε had on Randomised Response. Perhaps the other mechanisms would also have such a change if they had not performed well to begin with.

Ordering of all Commutes



(a) Randomised Response

(b) Unary Encoding

(c) OLH

(d) Hadamard Mechanism

Figure 5.9: LDP: Ordering of Counts with an ε of 5.

The ordering of all elements is looked at in fig 5.9. This demonstrates the fact that the LDP mechanisms perform much worse than central DP due to the perturbation of each individuals data. Even though the County Level has a smaller number of elements at 1085, this is still a high dimensional data universe. We did apply some post-processing to the output of the LDP mechanisms. We have chosen one of the simplest post-processing methods Base-Pro. This means that negative values are rounded up to zero. This was chosen as negative values do not make logical sense when dealing with commuting data of individuals. The true order is plotted

against the differentially private order with the black line represented the perfect ordering. The red dots represent the elements that have a noisy count of zero. In the county level, there were only 6 elements which had a true count of 0.

The RR performs the worst, there is no red shown on the graph as there were no noisy commutes which had a count of 0. After the 250 most popular commutes, the utility of RR is lost. It is very apparent that the random noise injected by the mechanism becomes problematic and the unpopular commutes are impossible to identify with confidence.

There are notably many elements set to 0 using UE, OLH and HM. UE set 329 elements to zero, OLH set 338 elements to zero and HM set the most elements to zero with 363 being set to 0. This means that before post-processing, all of these elements were either already 0 or had a negative value. In UE we can see one was ordered much higher than it should have been. This was the commute 'Carlow-Carlow' which was ordered as 638 but should have been 35.

5.2.2 Running Time

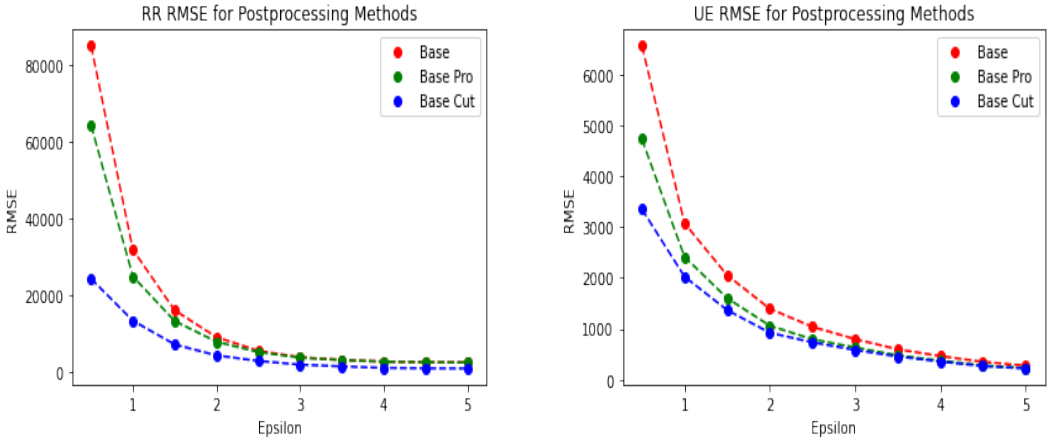
The RR and HM both ran in 9 seconds and 20 seconds respectively. Unary Encoding ran in 274 seconds while the OLH ran in 2296 seconds. In a practical setting, the Hadamard Mechanism is the best choice due it's quick running time and good utility compared to RR. Unary Encoding could also be considered but ultimately this running time makes it difficult to implement. This is due to the communication cost as each individuals data must be encoded into a vector of size 1085. The running time for OLH makes it an inadequate choice. This long run time is due to the use of hash functions. The Hadamard Mechanism also uses hash functions but the use of Hadamard Transforms allows for a short running time in comparison.

5.3 Post-Processing

We have applied three post processing techniques on the LDP mechanisms in an attempt to improve their utility. As mentioned, the above analysis was done with non-negative values. We call this post-processing Base Pro. We wanted to compare this with the original output of the mechanisms where values can be negative. This is known as Base. Finally, we discussed in Chapter 2 that Base Pro creates a positive bias in the data as only negative noise is removed. Thus our last method that we investigated is Base Cut. This ensures that the number of individuals remain the same. The RMSE of the three methods can be seen in 5.10 which were run for each of the 4 LDP mechanisms.

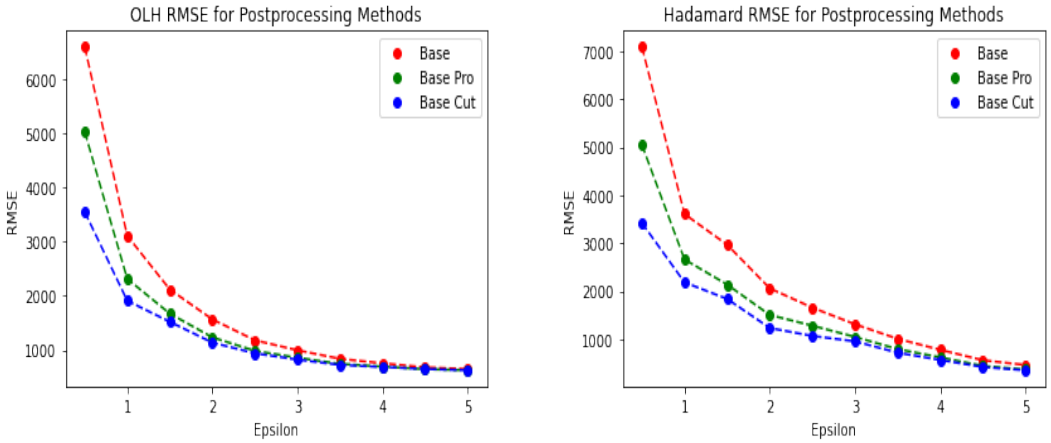
The consensus for all mechanisms is that Base Cut performs the best followed by Base Pro and lastly Base. The RMSE for all mechanisms decreases with a higher privacy budget and with a privacy budget of 5, there is not much difference between the methods.

As Base Cut outperforms Base Pro, we carry out a final test to observe the affect that this post-processing has. This is done by observing the ordering of all of the commutes as the difference will only become apparent for the lower frequented commutes. We perform this for the Hadamard Mechanism only as the other mechanisms will behave in the same manner. The number of elements that were set to 0 went from 363 to 782 elements when moving from Base Pro to Base Cut.



(a) Randomised Response

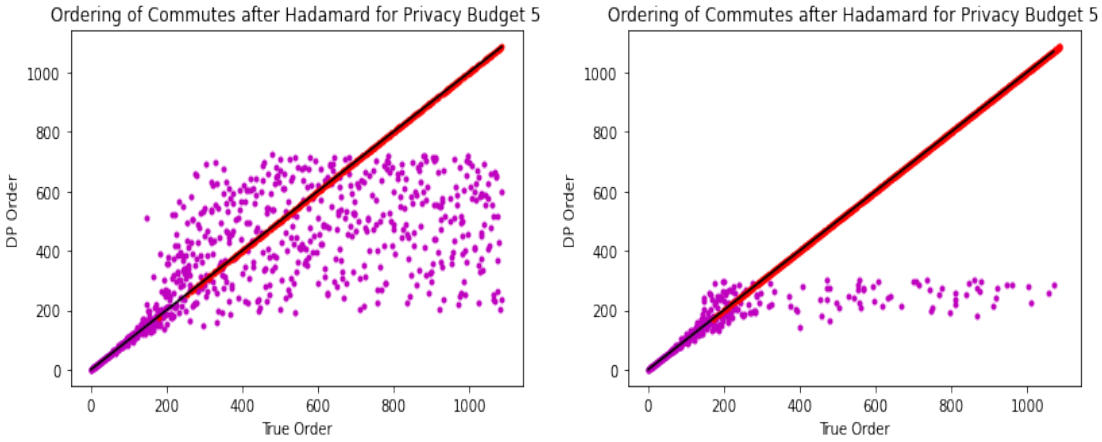
(b) Unary Encoding



(c) OLH

(d) Hadamard Mechanism

Figure 5.10: LDP: The affect of post-processing on the RMSE



(a) Base Pro

(b) Base Cut

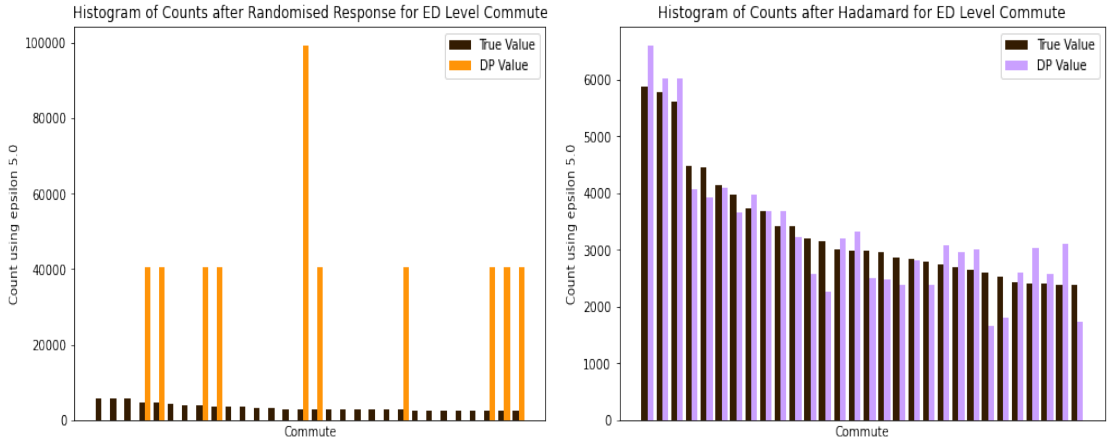
Figure 5.11: Ordering for Hadamard Mechanisms at County Level with an ε of 5 for different Post-Processing Methods.

The threshold in Base Cut is set at the number of individuals in the dataset. As we can see in 5.11, all of the elements after the threshold are set to 0. It is clear that the resulting elements are the most frequented commutes and overall, there is good utility using Base Cut. However some false positives remain in which non-frequent commutes are marked much higher than they should. For many applications using mobility data such as determining most popular commutes, Base Cut gives us better utility. The number of individuals that commute are preserved making it easier to gain insights from the data.

5.4 LDP Mechanisms - ED Level

At the ED level, there are over 9.7 million elements as we have to consider every possible commute, even commutes not present in the dataset. Because of the significant increase of $|\mathcal{X}|$, only 2 of the mechanisms were feasible to run due to to running times. `UE` was ran once for an ε of 5 which took 154272 seconds (42.2 hours). This is unfeasible and so we did not continue with our analysis for `UE`. `OLH` was not run as this took significantly longer than `UE` and would also be unfeasible to use as a possible mechanism for the ED Level. We shift our focus to the remaining two mechanisms, Randomised Response and the Hadamard Mechanism. The privacy budget of 0.5 was not suitable for the LDP mechanisms seen clearly from Table 5.3. Thus the analysis was carried out with the lowest level of privacy we considered thus far, an ε of 5.

We can see the histogram of the top 30 counts for `RR` and `HM` in Figure 5.12. The dependence on the size of the data universe causes `RR` to lose all utility. We saw that it performed the worst at the county level and this is magnified at the ED level.



(a) Randomised Response

(b) Hadamard Mechanism

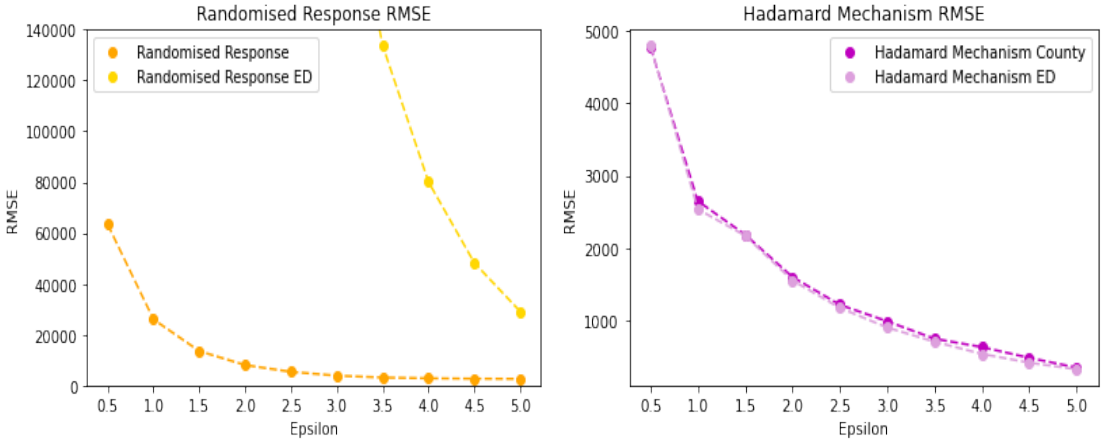
Figure 5.12: Histogram of the Top 30 Counts at ED Level with an ε of 5

5.4.1 Utility

RMSE

The RMSE of `RR` and `HM` for the County Level and the ED level are shown in Figure 5.13. The RMSE of `RR` run on the ED level is significantly worse than at the county level and undoubtedly shows that it is not viable. The RMSE of the Hadamard remains stable for the ED level. This

does not paint the full picture which can be seen when observing the ordering of the commutes. This is intuitively because an RMSE of 1000 has a much larger impact at the ED level.



(a) Randomised Response

(b) Hadamard Mechanism

Figure 5.13: LDP RMSE comparing County Level & ED Level

5.4.2 Maximum Error

In Figure 5.14 we observe that the maximum error for `RR` remains stable for all values of ε. For `HM`, we see that the maximum error does decrease with ε. The values are lower than that of the County level but as we noted with the RMSE, this is because each element contains far less values.

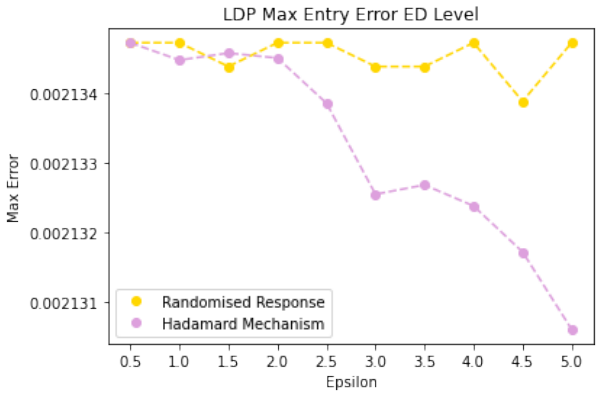


Figure 5.14: Maximum Error for LDP at ED Level

Top 10 frequented commutes

We would now like to observe how the top 10 most frequent commutes are ranked. Table 5.3 shows a privacy budget of 0.5 and Table 5.4 shows a privacy budget of 5. The central mechanisms perform well for both privacy levels. `RR` is not suitable for the ED level as there is no utility output in the noisy counts. The `UE` is able to identify the top 10 counts with $\varepsilon = 5$ however it cannot be used due to the extremely long running times. This leaves only the Hadamard

ED Commutes with a privacy budget of 0.5

Commute	True Count	Lap	Stab H	RR	UE	OLH	Hadamard
Navan Rural-Navan Rural	1	1	1	5031772	-	-	5800035
Blanchardstown-Blakestown x2	2	2	2	967435	-	-	7491580
Naas Urban-Naas Urban	3	3	3	6254410	-	-	56175
Maynooth-Maynooth	4	4	4	6254409	-	-	859197
Clenagh-Clenagh	5	5	5	6254408	-	-	3554047
Ballycummin-Ballycummin	6	6	6	2059273	-	-	1563508
Ballincollig-Ballincollig	7	7	7	6254407	-	-	2788501
Ballysimon-Ballysimon	8	8	8	6254406	-	-	1603968
Tralee Rural-Tralee Urban	9	9	9	6254405	-	-	6766584
Carrigaline-Carrigaline	10	10	10	6254404	-	-	6766583

Table 5.3: Utility: Top 10 Commutes at a Electoral Division Level, $\varepsilon = 0.5$.

ED Commutes with a privacy budget of 5

Commute	True Count	Lap	Stab H	RR	UE	OLH	Hadamard
Navan Rural-Navan Rural	1	1	1	5031638	2	-	1
Blanchardstown-Blakestown x2	2	2	2	6980683	1	-	3
Naas Urban-Naas Urban	3	3	3	6254322	3	-	2
Maynooth-Maynooth	4	4	4	2141293	5	-	5
Clenagh-Clenagh	5	5	5	2141294	4	-	7
Ballycummin-Ballycummin	6	6	6	6254321	7	-	4
Ballincollig-Ballincollig	7	7	7	6254320	8	-	10
Ballysimon-Ballysimon	8	8	8	2141300	6	-	6
Tralee Rural-Tralee Urban	9	9	9	2141301	10	-	8
Carrigaline-Carrigaline	10	10	10	6254319	9	-	9

Table 5.4: Utility: Top 10 Commutes at a Electoral Division Level, $\varepsilon = 5$.

Mechanism. It is not usable with a low ε but raising the privacy budget allows the Hadamard Mechanism to identify the top 10 counts in a reasonable running time.

Ordering of the Elements

As the Hadamard performed well for identifying the top 10 commutes, the analysis is extended to the top 100 commutes. In 5.15 it can be seen that the Hadamard Mechanism performs well for the top 40 counts before the utility decreases. The dark purple dots at the top of the graph show commutes that were ranked lower than 200 using Hadamard. Many of these commutes were ranked much lower than their true value. For example the 24^{th} most popular commute 'Ratoath-Ratoath' is ordered as number 1631 or the commute 'Mullingar Rural-Mullingar North Urban' is placed 686843^{rd} when it should be the 48^{th} most popular ED commute. This poses an issue as the Hadamard Mechanism is not accurate for many of the ED commutes.

(a) County Level

(b) ED Level

Figure 5.15: Hadamard Ordering of Top 100 Counts with an ε of 5.

5.4.3 Running Times

Randomised Response took 7 seconds to run. The Hadamard Mechanism took 32 seconds.

5.5 Electoral Division Level Subsets

An investigation of the electoral divisions for the whole of Ireland is not feasible due to the huge data universe. Thus we have split the dataset into 2 separate subsets of EDs which is illustrated in 5.16 where red shows subset D and the blue shows subset R. The first dataset subset which we will denote as subset D is composed of the EDs of Dublin City. There are 218,860 individuals that commute entirely within Dublin City and we have a data universe size of $\mathcal{X}_D = 26,244$. This correspond to 162 possible origin and destination locations. We then found another subset which had the same size of $\mathcal{X}$. We used the counties Cavan and Leitrim which has $\mathcal{X}_R = \mathcal{X}_D = 26,244$. We denote this subset as R - the rural subset. There are 38,630 individuals in R which is 16% of Dublin. Up to this point, we have ran comparisons on different data universe sizes but kept n fixed. We now reverse this and keep the size of the data universe fixed and vary n. There are two main objectives, the first is to ensure that we can run all LDP mechanisms on a second data universe. The second goal is that by varying the number of individuals, we hope to have a complete and holistic analysis of the state of the art LDP mechanisms. We expect that `UE`, `OLH` and `HM` will all have a similar RMSE and `RR` will not perform well. We saw in Chapter 3 that the error bound is $\mathcal{O}\left(\frac{1}{\varepsilon\sqrt{n}}\right)$. Thus it is predicted that the error will be lower for the Dublin Area.

Figure 5.16: Two Subsets of the Electoral Division.

RMSE

The RMSE was calculated for all 4 LDP mechanisms as shown in 5.17. The per entry error for RR is $\mathcal{O}\left(\frac{|\mathcal{X}|}{\varepsilon n}\right)$ and $\mathcal{O}\left(\frac{1}{\varepsilon n}\right)$ for the other mechanisms. For this final test, we fixed the privacy budget ε to 5 and the size of the data-universe at 26,244 and varied n. Subset D is plotted in shades of purple and subset R is plotted in shades of blue. Every mechanism had a lower RMSE for R which had a lower number of individuals. This confirms our calculations and completes our analysis on the LDP mechanisms.

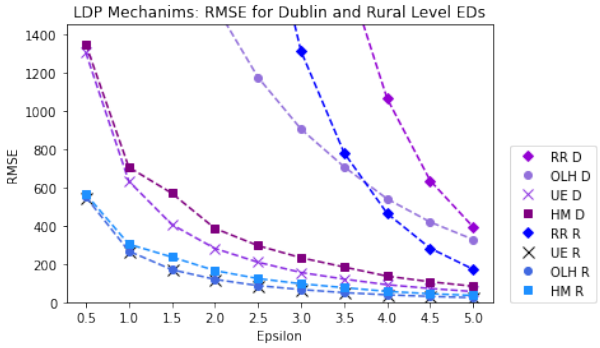


Figure 5.17: LDP RMSE for Dublin and Rural EDs.

Chapter 6

Conclusion

In this final section we will review the topics covered and the results obtained from our research. In total we analysed 6 differentially private mechanisms theoretically and practically. We provided a comprehensive overview of the state of the art LDP mechanisms. The LDP mechanisms studied were Randomised Response, Unary Encoding, Optimised Local Hashing and the Hadamard Mechanism. We compared these LDP mechanisms against two central DP mechanisms namely the Laplace Mechanism and the Stability Histogram. This highlighted the trade-off between privacy and utility in central DP and LDP. The central DP mechanisms performed significantly better on our mobility dataset as less noise was required to perturb the data. The LDP mechanisms provided a higher level of privacy to the individuals but we saw how this negatively impacted the utility.

Providing differentially private solutions to mobility data is a difficult task. The data universe size for mobility data is massive. We applied various tests to our mobility dataset on origin-destination commuting. We aimed to provide a thorough analysis of each mechanism's privacy, utility and running times. We varied the size of the data universe, the privacy budget and the number of individuals to provide a full understanding of the theoretical results.

The Hadamard Mechanism was the best choice for our large domains. It is almost as accurate as Unary Encoding and Optimised Local Hashing but it was orders of magnitudes faster to run. It proved to be the only LDP mechanism suitable for practical settings. We applied 3 different post-processing techniques in an attempt to improve the utility. The techniques we considered were some of the most simplistic. It would have been interesting to consider more complex techniques that use prior knowledge such as the expected distribution of mobility data to further improve on our results.

In summary, the LDP mechanisms provided differentially private results to our counting queries. They proved to be accurate in identifying the most popular commutes of individuals in Ireland. They performed best with smaller domain sizes and their utility was reduced greatly for the larger domains. Randomised Response was inarguably the worst LDP mechanism. Unary Encoding, Optimised Local Hashing and the Hadamard Mechanism all yielded similar utility but ultimately the Hadamard Mechanism was the only practical mechanism for our mobility data. This was due to the infeasible running times of Unary Encoding and Optimised Local Hashing. Further work is needed to improve on the utility and scalability of the LDP mechanisms. Different methodologies such as heavy hitter identification would be better suited for large domain sizes. More exploration is needed for the use of post-processing techniques, domain size reductions and the use of composition.

Acknowledgements

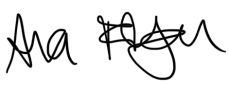
I would like to thank Prof. Francesco Silvestri and Dott. Fabrizio Boninsegna for their guidance throughout my thesis. I have discovered a new area of research that is deeply interesting and this would not have been possible without their counsel. I now have a taste of research and I will take this experience into my future endeavours.

I have had ups and downs in my journey but this had made it even more special and memorable. I would like to thank Alberto, my Unipd mentor who pushed me to think outside of the box and contemplate the road ahead of me.

I would also like to thank all my friends that I have met here in Padova. My experience living in Italy would not be the same without you all and I'm very lucky to be surrounded by such inspiring and genuine people.

I am very thankful to my parents, and to my brothers, for supporting me in all of my decisions. I dedicate this thesis to my Nonno, who always believed in me.

Grazie di cuore,



Bibliography

[Abo] John M. Abowd. Protecting the confidentiality of americas statistic's: Adopting modern disclosure avoidance methods at the census bureau.

[ASZ19] Jayadev Acharya, Ziteng Sun, and Huanyu Zhang. Hadamard response: Estimating distributions privately, efficiently, and with little communication. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1120–1129. PMLR, 2019.

[BNS15] Mark Bun, Kobbi Nissim, and Uri Stemmer. Simultaneous private learning of multiple concepts. *Journal of Machine Learning Research*, 20(94):1–34, 2015.

[BNSGT17] Raef Bassily, Kobbi Nissim, Uri Stemmer, and Abhradeep Guha Thakurta. Practical locally private heavy hitters. *Advances in Neural Information Processing Systems*, 30, 2017.

[Bur21] U.S. Census Bureau. *Disclosure Avoidance for the 2020 Census: An Introduction*. U.S. Government Publishing Office, November 2021.

[CB22] Graham Cormode and Akash Bharadwaj. Sample-and-threshold differential privacy: Histograms and applications. In *International Conference on Artificial Intelligence and Statistics*, pages 1420–1431. PMLR, 2022.

[CD14] Aaron Roth Cynthia Dworkand. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3-4):211–407, 2014.

[CKO20] Wei-Ning Chen, Peter Kairouz, and Ayfer Ozgur. Breaking the communication-privacy-accuracy trilemma. *Advances in Neural Information Processing Systems*, 33:3312–3324, 2020.

[CMM21] Graham Cormode, Samuel Maddock, and Carsten Maple. Frequency estimation under local differential privacy [experiments, analysis and benchmarks]. *arXiv preprint arXiv:2103.16640*, 2021.

[DKY17] Bolin Ding, Janardhan Kulkarni, and Sergey Yekhanin. Collecting telemetry data privately. *Advances in Neural Information Processing Systems*, 30, 2017.

[DMNS06] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography: Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006. Proceedings 3*, pages 265–284. Springer, 2006.

[DP09] Devdatt P Dubhashi and Alessandro Panconesi. *Concentration of measure for the analysis of randomized algorithms*. Cambridge University Press, 2009.

[DPT17] Apple Differential Privacy Team. Learning with privacy at scale. 2017.

[DVGM20] Damien Desfontaines, James Voss, Bryant Gipson, and Chinmoy Mandayam. Differentially private partition selection. *arXiv preprint arXiv:2006.03684*, 2020.

[EFM+19] Úlfar Erlingsson, Vitaly Feldman, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Abhradeep Thakurta. Amplification by shuffling: From local to central differential privacy via anonymity. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2468–2479. SIAM, 2019.

[EKP14] Úlfar Erlingsson, Aleksandra Korolova, and Vasyl Pihur. RAPPOR: randomized aggregatable privacy-preserving ordinal response. *CoRR*, abs/1407.6981, 2014.

[GHV20] Marco Gaboardi, Michael Hay, and Salil Vadhan. A programming framework for opendp. *Manuscript, May*, 2020.

[KS14] Shiva P Kasiviswanathan and Adam Smith. On the'semantics' of differential privacy: A bayesian formulation. *Journal of Privacy and Confidentiality*, 6(1), 2014.

[LBT+20] Heiko Ludwig, Nathalie Baracaldo, Gegi Thomas, Yi Zhou, Ali Anwar, Shashank Rajamoni, Yuya Ong, Jayaram Radhakrishnan, Ashish Verma, Mathieu Sinn, et al. Ibm federated learning: an enterprise framework white paper v0. 1. *arXiv preprint arXiv:2007.10987*, 2020.

[NS08] Arvind Narayanan and Vitaly Shmatikov. Robust de-anonymization of large sparse dataset. *In Proceedings of the 29th IEEE Symposium on Security and Privacy, SP '08*, page 111–125, 2008.

[Off16a] Central Statistics Office. Census 2016 place of work, school or college - census of anonymised records (powscar), 2016.

[Off16b] Central Statistics Office. Census of population 2016 – profile 6 commuting in ireland, 2016.

[PT20] University of Pennsylvania Penn Today. Can contact tracing stop the spread of covid-19?, 2020.

[RLAW24] Rob Romijnders, Christos Louizos, Yuki M Asano, and Max Welling. Protect your score: Contact-tracing with differential privacy guarantees. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 14829–14837, 2024.

[SSND+21] Kamil Smolak, Katarzyna Siła-Nowicka, Jean-Charles Delvenne, Michał Wierzbiński, and Witold Rohm. The impact of human mobility data scales and processing on movement predictability. *Scientific Reports*, 11(1):15177, 2021.

[TWJ17] Ninghui Li Tianhao Wang, Jeremiah Blocki and Somesh Jha. Locally differentially private protocols for frequency estimation. *26th USENIX Security Symposium (USENIX Security 17)*, pages pp. 729–745, 2017.

[Vad17] Salil Vadhan. The complexity of differential privacy. *Tutorials on the Foundations of Cryptography: Dedicated to Oded Goldreich*, pages 347–450, 2017.

[War65] Stanley L Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60(309):63–69, 1965.

[WLZL+19] Tianhao Wang, Milan Lopuhaä-Zwakenberg, Zitao Li, Boris Skoric, and Ninghui Li. Locally differentially private frequency estimation with consistency. *arXiv preprint arXiv:1905.08320*, 2019.

[ZRX+23] Yuanbo Zhang, Daniel Ramage, Zheng Xu, Yanxiang Zhang, Shumin Zhai, and Peter Kairouz. Private federated learning in gboard. *arXiv preprint arXiv:2306.14793*, 2023.