Guaranteeing Differential Privacy using Quantitative Information Flow Analysis

COMP90055 Research Project

Alec Miller
Under the supervision of Dr. Christine Rizkallah

School of Computing and Information Systems
The University of Melbourne

June 2024

Guaranteeing Differential Privacy using Quantitative Information Flow Analysis

COMP90055 Research Project

Alec Miller

Abstract

In today's increasingly data-driven society, huge quantities of sensitive personal data are collected from individuals and analysed, from health records to financial transactions. Insights gained from collected data can be of great benefit to the public. However, these insights must be gained in a manner such that the information of an individual can not be inferred. Differential privacy provides a rigorous framework for guaranteeing privacy in such programs, to ensure a balance is struck between accurate analysis and the protection of sensitive information. In this project, I extend recent work on using Quantitative Information Flow (QIF), a framework for quantifying the severity of information leaks in programs, to analyse differential privacy mechanisms. I demonstrate this analysis with examples of differential privacy programs coded in Kuifje, a programming language built for QIF. The techniques introduced in this report allow for automatic validation and calculation of differential privacy guarantees of mechanisms, providing assurance that data used in these mechanisms supply desired levels of privacy.

Declaration and Acknowledgements

I declare that:

- This project does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any university; and that to the best of my knowledge and belief it does not contain any material previously published or written by another person where due reference is not made in the text.
- This research did not require clearance from the University's Ethics Committee.
- This report is 6000-8000 words in length (excluding text in images, bibliographies and appendices).

This is my original work, under the supervision of Dr. Christine Rizkallah. I want to thank Christine for her guidance and support throughout the project and for providing an insightful introduction to the world of academia. I would also like to thank Vincent Jackson for his assistance with the Kuifje setup and helping me to better understand the language, as well as for his commitment to Kuifje's development.

Contents

1. Research Summary

1.1 Questions

This project aims to answer the following key questions:

1. How can the current literature on the relationship between differential privacy and QIF be extended to fit the more general definition of approximate differential privacy?

2. How can Kuifje, the QIF-aware programming language, be used to model differential privacy mechanisms and automatically validate their privacy guarantees?

1.2 Aims

The following aims are given to address the questions above:

1. Provide formal definitions for approximate differential privacy as a problem in QIF (Section 4).

2. Program differential privacy mechanisms in Kuifje and demonstrate how Kuifje output can be used to validate or calculate differential privacy guarantees of mechanisms (Sections 4-6).

1.3 Significance of the Research

Whilst there are previous works discussing how to use QIF to validate the level of differential privacy of programs, there have been minimal examples. These previous examples have only been theoretical and focused on the most simple definition of differential privacy, that is, pure differential privacy. This paper demonstrates programming differential privacy implementations in Kuifje, a language built for QIF, and how Kuifje's output may be analysed to validate differential privacy guarantees. I demonstrate this analysis for pure differential privacy, as well as the more general notion of approximate differential privacy. The methods presented in this paper further

the discussion on automatically validating differential privacy properties. It is hoped that such techniques may be used to assure privacy in more complex, real-world privacy applications.

2. Introduction

With frequent reports of significant data leaks, the present technological landscape is marked by a critical need to ensure that personal data is secure. The benefits of collecting and using statistics from data are evident, from enhancing and optimising business decision-making, to improving healthcare services, to detecting fraud in financial systems. However, serious consideration must be given to how these statistics can obtained whilst safeguarding the privacy of individuals.

Differential privacy, first introduced in [1], is one prominent method of guaranteeing privacy in programs that use sensitive data. Mechanisms used to supply differential privacy add random noise to databases, providing bounds on how much an individual data entry can affect the likelihood of results when querying the data. This ensures that any individual's data used in such a mechanism can not be deduced to an acceptable degree, whilst insights that can be gained from the database as a whole remain reliable.

Quantitative Information Flow (QIF) is a paradigm for quantifying leakage of secret information used in computer programs. QIF aims to measure leakage severity, given the presence of an adversary who can make observations from running programs. QIF has intuitive links to differential privacy, and it has long been discussed how to analyse differential privacy mechanisms using ideas from QIF [2, 3]. More detailed background information on QIF, differential privacy and the connection between them is given in Section 3.

In Section 4, I extend recent work [4] by Morgan and McIver on how QIF can be used to analyse differential privacy mechanisms. Morgan and McIver theorised that such analysis could be conducted using Kuifje [5], a programming language developed for QIF. Using Kuifje will allow for differential privacy properties of mechanisms to be automatically calculated or validated, ultimately assuring that the use of these mechanisms provides acceptable guarantees about the privacy levels of data.

I also extend the ideas presented in Morgan and McIver's paper to fit a less stringent definition of differential privacy, namely approximate differential

privacy [6]. This extension is required in popular differential privacy implementations such as the Gaussian Mechanism. In Section 5, I demonstrate how one may use the presented methods for analysing the differential privacy properties on the discretised version of the Gaussian Mechanism, as well as other prevalent differentially private mechanisms, namely the Randomised Response Mechanism [7] and the discrete Laplace Mechanism [8].

3. Background

3.1 Differential Privacy

3.1.1 Motivation and Foundations

Many techniques for data privacy revolve around keeping data anonymous, with the idea that if a malicious actor were to gain access to personal data, no identifying information such as names, addresses or phone numbers would be available for linking data to an individual. However, as simple as that may sound, anonymised data is still surprisingly vulnerable. In what is known as a linkage attack, an attacker deidentifies anonymised data in one source by matching it to public data in an auxiliary source. Studies have shown that very few characteristics are needed to uniquely identify a person, making linkage attacks much easier than one might expect. For example, 87% of Americans are uniquely identifiable by their ZIP code, date of birth and gender alone [9]. One study [10] linked anonymised Netflix records to IMDb reviews to re-identify users and reveal sensitive information, demonstrating how even non-demographical can be used to de-anonymise data.

Differential privacy, first formally introduced by Dwork [1], presents an alternative solution to the problem of keeping data secure. Instead of relying on anonymisation, differential privacy involves using algorithms to alter data by adding random noise. This ensures that each individual's data is privatised even without anonymity, whilst queries on aggregated data will maintain a desirable level of accuracy.

Figure 3.1 demonstrates the algorithm for the Randomised Response Mechanism [7], which is considered the first differentially private algorithm to be proposed. This scenario involves adding responses to a database to a potentially sensitive yes/no question, such as 'Have you used illegal drugs in the past year?'. A respondent to the question tosses a coin for a 50-50 chance firstly to determine whether they answer truthfully or not. If not, they toss another coin which determines whether they answer 'yes' or 'no'. This process results in a 25% chance of the respondent's true answer to the question being different to what is entered into the database. With a large enough

number of responses to the question, someone could query the average or sum of 'yes' responses with an acceptable level of accuracy, whilst being uncertain about the response of any individual. This process ultimately means any individual who participates in the survey can have confidence that their true response will not be inferred and is therefore private.

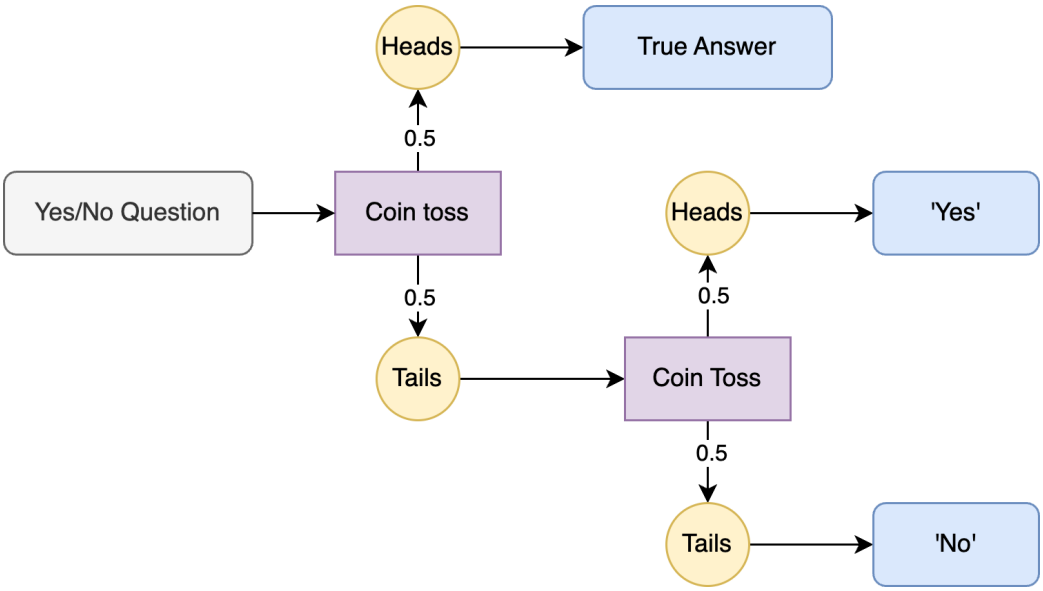

Figure 3.1: Algorithm for adding a response to a yes/no question to a database, using the Randomised Response Mechanism. Probabilities of branches of computation are indicated on the arrows.

There are a number of variations of the mechanism, including using randomisation devices other than a coin, with different probabilities of flipping an answer, as well as for data with more than two categories [11]. The premise remains the same in each of these variations: that ultimately, any individual who participates in the survey can have confidence that their true response can not be inferred, whilst the collected data remains useful.

3.1.2 Formal Definitions

Figure 3.2 shows the intuition surrounding the formal definitions of differential privacy. Say we have a person X, who is unconfident about the security

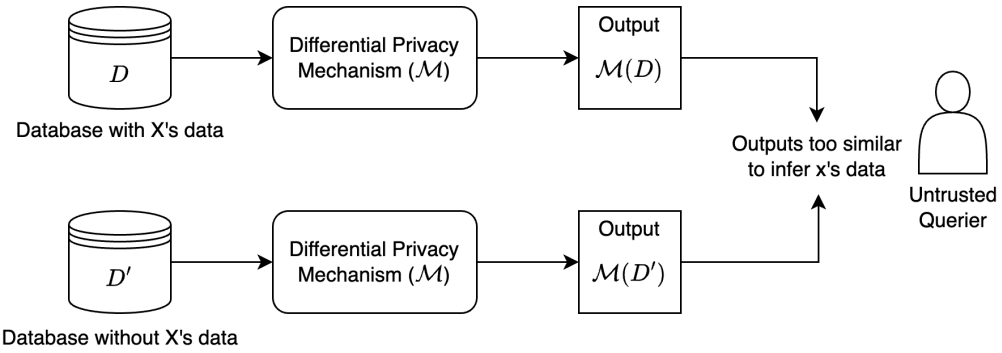

Figure 3.2: Diagram showing how differential privacy keeps person X's data secure from an untrusted querier.

of their data. Differential privacy ensures that when a mechanism $\mathcal{M}$ is run on a database, whether or not X's data is used (i.e. whether or not the database is D or D'), the respective outputs of the mechanism $\mathcal{M}(D)$ and $\mathcal{M}(D')$ will be similar enough that the querier will be unable to infer X's data.

The original formal definition of differential privacy [1] relates to comparing the likelihood of outputs given by algorithms on databases differing by no more than one element.

Definition 1 (ϵ-Differential Privacy) *A randomised algorithm $\mathcal{M}$ is ϵ-differentially private if for all databases D and D' that differ by at most one element, and all $S \subseteq \text{Range}(\mathcal{M})$,*

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

In other words, if an entry is added to a dataset, no outputs of the algorithm are much more likely to be produced whatever the value of the entry ('yes' or 'no' in the Randomised Response Mechanism). Also, no outputs of the algorithm would be much more likely to be produced if the data was not added in the first place. The privacy parameter ϵ indicates the strength of the guarantee, with smaller ϵ values providing stronger privacy.

Note that, when we discuss discrete output sets, instead of the definition needing to hold for all subsets $S \subseteq \text{Range}(\mathcal{M})$, the definition is equivalent to

checking the probability of individual outputs, i.e. comparing $\Pr[\mathcal{M}(D) = y]$ and $\Pr[\mathcal{M}(D') = y]$ for $y \in \text{Range}(\mathcal{M})$ [2].

If a mechanism is ϵ-differentially private, it is also ϵ'-differentially private for all $\epsilon' > \epsilon$. Throughout this report, I use both the terms 'validating' and 'calculating' in relation to obtaining differential privacy guarantees. 'Validating' refers specifically to checking that a mechanism satisfies a certain level of differential privacy. That is by setting ϵ to a specific value and showing that the inequalities in the differential privacy definition hold, we validate the mechanism satisfies ϵ-differential privacy. 'Calculating' refers to finding the strongest guarantee satisfied by a mechanism, that is, we calculate the smallest ϵ (or later, δ) for which the inequalities hold.

If we consider the Randomised Response Mechanism, when the mechanism outputs the answer 'yes' for a respondent, we can compare the probabilities of the respondent's true answer:

$$\frac{\Pr[\text{Response} = \text{Yes}|\text{Truth} = \text{Yes}]}{\Pr[\text{Response} = \text{Yes}|\text{Truth} = \text{No}]} = \frac{3/4}{1/4} = 3 = \exp(\log(3))$$

Note that we have the same ratio of probabilities when the true answer is 'no'. Since the ratio of the probabilities is $\exp(\log(3))$, then the Randomised Response Mechanism is $\log(3)$-differentially private. Also, the Randomised Response Mechanism is ϵ-differentially private for all $\epsilon \geq \log(3)$.

If, in the Randomised Response Mechanism, we set a 50% chance of flipping a user's answer instead of 25%, any input can produce any output with equal probability. This modified algorithm is therefore 0-differentially private. However, the average or sum of all responses would give no meaningful information about the data that created it. For the general design of differential privacy mechanisms, ϵ should be small, around 1 or smaller depending on the dataset and what we wish to gain from it [12], but not too close to 0.

It may be the case that an algorithm provides ϵ-differential privacy most of the time, but with some small probability will give output such that the inequality in Definition 1 will not hold. As I soon demonstrate, for certain mechanisms, this will be the case no matter how large ϵ is set to. Therefore, to accurately provide bounds on the extent to which an algorithm fails to

provide ϵ-differential privacy, it's necessary to introduce the idea of (ϵ, δ)-differential privacy, also known as approximate differential privacy.

Definition 2 ((ϵ, δ)-differential privacy) *A randomised algorithm $\mathcal{M}$ is (ϵ, δ)-differentially private if for all databases D and D' that differ by at most one element, and all $S \subseteq \text{Range}(\mathcal{M})$,*

$$\Pr[\mathcal{M}(D) \in S] \leq \exp(\epsilon)\Pr[\mathcal{M}(D') \in S] + \delta$$

The value of δ gives an allowance for the extent to which $\mathcal{M}$ fails to provide ϵ-differential privacy. Note that while there is a unique minimum value of ϵ for a mechanism providing differential privacy, for (ϵ, δ)-differential privacy, there is a curve of possible minimum pairs. So, in general, we fix ϵ to a desired value and calculate the minimum value of δ for which the inequalities hold.

Definition 2 doesn't provide much information about the behaviour of a mechanism when it fails. For instance, the following algorithm, is $(\log(3), 0.001)$-differentially private:

With probability 0.999: Run the Randomised Response Mechanism.
Otherwise: Leak the whole database.

Even though the probability of leaking the whole database is small, this would be a terrible algorithm to use in practice. We will soon see an example in the Gaussian mechanism, which must have $\delta > 0$, but fails to provide ϵ-differential privacy in a manner much less catastrophic than the Catastrophe mechanism, and is applied in many practical situations for its ability to preserve the usefulness of aggregated data over ϵ-differential private alternatives.

3.1.3 The Laplace and Gaussian Mechanisms

In the Randomised Response Mechanism, each data entry is altered individually before being given to an untrusted aggregator. This setting is known as local differential privacy. Here we introduce two mechanisms that alter a query made on a database containing raw data, controlled by a trusted curator. This setting is known as global differential privacy. The differences are illustrated in Figure 3.3.

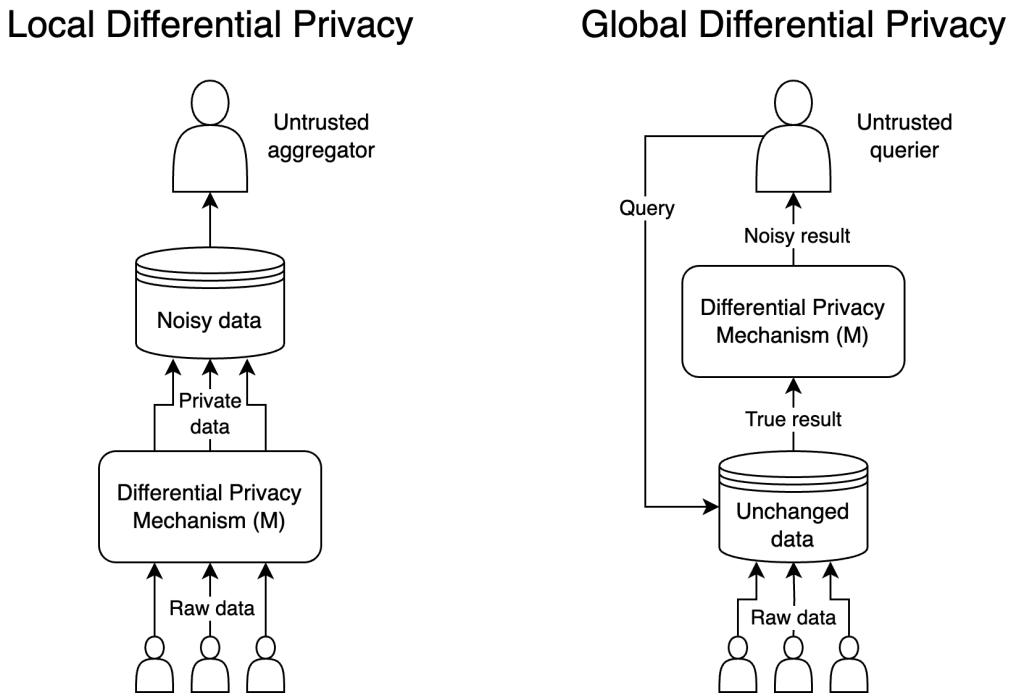

Figure 3.3: Comparison between local and global differential privacy.

The global privacy setting is intuitive for counting queries, which return the number of elements in a database that satisfy a certain property [13], with no other access to the data given to the querier.

The Laplace Mechanism [8] and Gaussian Mechanism [6] are very similar. Figure 3.4 shows the flow of the Laplace Mechanism. Differential privacy is guaranteed by adding random noise to the true answer of the counting query, sampled from the Laplace distribution. The noisy answer to the counting query is returned to the querier, which, as seen in the example, is generally very close to the true answer.

Figure 3.5 shows the probability density function of example noisy count values, that is, the true answer to the counting query + the Laplace PDF in 3.6 on neighbouring databases, i.e. databases that differ by one entry. In this example, these true counts are 100 and 101. For any x, one PDF is never

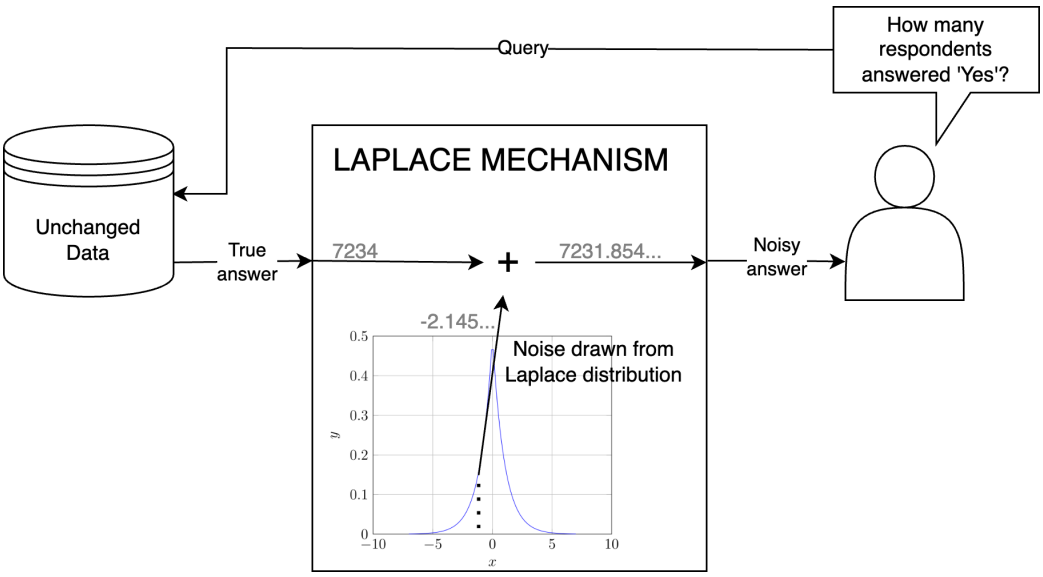

Figure 3.4: How the Laplace Mechanism works on an example counting query.

greater than the other by more than a factor of e^{ϵ}. Hence, the probability of observing any output (or subset of outputs, since this is a continuous distribution,) never changes by more than a factor of e^{ϵ}. That is, the Laplace mechanism is ϵ-differentially private.

As the name indicates, the key difference in the Gaussian Mechanism is that noise is drawn from the Gaussian distribution instead of the Laplace distribution.

Formally, where M is the mechanism on database X and f is a query to the data e.g. a counting query:

$$M(X) = f(X) + Y \ , \ Y \sim \text{Lap}(1/\epsilon) \text{ or } \text{N}(0, \sigma^2)$$

The Laplace distribution $\text{Lap}(1/\epsilon)$ with parameter $1/\epsilon$ provides ϵ-differential privacy as part of the mechanism. The Gaussian distribution $\text{N}(0, \sigma^2)$ with variance σ^2 as part of the mechanism provides (ϵ, δ)-differential privacy with the relation $\sigma^2 = \frac{\log(1.25/\delta)}{\epsilon^2}$. Probability density functions of the distributions are provided in Appendix A.1.

Figure 3.7 demonstrates the intuition behind why the Gaussian mechanism

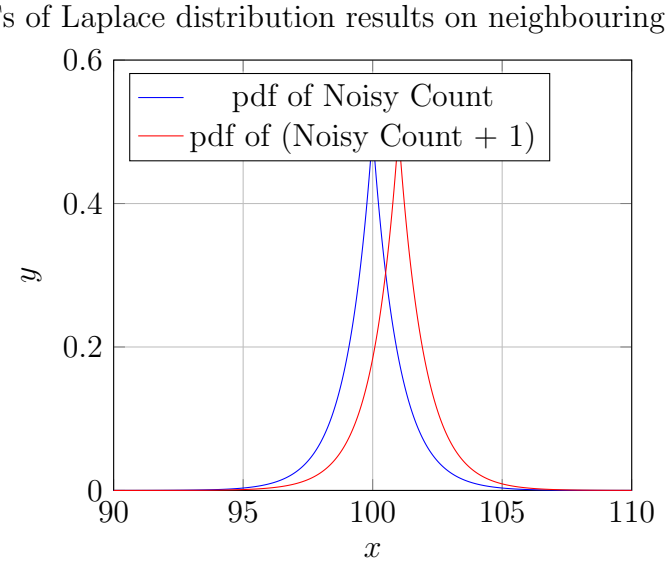

Figure 3.5: Comparison of PDFs of distribution results on neighbouring databases.

only provides (ϵ, δ)-differential privacy and not ϵ-differential privacy. Comparing the probability density functions for the noisy counts for two neighbouring databases, the probabilities of observing any output mostly never differ by a large amount (more than a factor of $e^{\epsilon} = e$). However, for increasingly unlikely outputs in the tails of the distributions, the ratio between the probabilities of outputs gradually increases beyond the factor of e^{ϵ}, which is where the δ value comes from.

We can observe in Figure 3.6 that the Gaussian distribution has a wider spread than the Laplace distribution with the same ϵ, meaning the output is expected to be less accurate for a weaker privacy guarantee. Considering these facts, one may ask why one would consider the Gaussian mechanism over the Laplace mechanism. The advantage of using the Gaussian mechanism comes from being able to use a different measure of sensitivity when determining how much noise needs to be added. Generally speaking, sensitivity is the amount a function will change when its input changes. The different sensitivity measure means that much less noise may need to be added to more complex multi-valued queries for a similar privacy guarantee, compared to the Laplace distribution. This is beyond the discussion of this

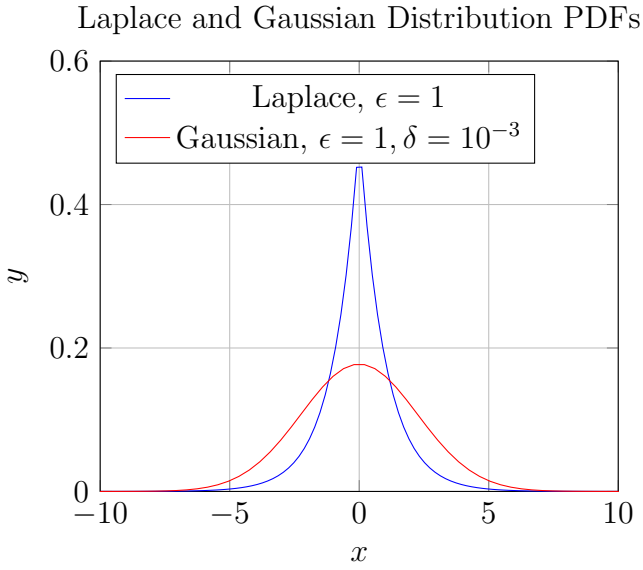

Figure 3.6: Comparison of PDFs of the Laplace and Gaussian distributions.

report, as all examples are for single-valued queries with sensitivity=1. However, it is worth noting that both of these mechanisms find use in practice.

In practice, it is ideal to use the discrete versions of these distributions, which give integer output, as displayed in Figure 3.8. The probability mass functions are provided in Appendix A.1. These are the Discrete Laplace Mechanism [14] and the Discrete Gaussian Mechanism [15]. These discretisations provide essentially the same privacy and accuracy guarantees as their continuous counterparts. The discrete mechanisms are preferred due to security issues related to using floating point arithmetic to approximate continuous functions, as demonstrated by Mironov in [16]. Another advantage is interpretability; for instance, it makes little sense to say 7231.854... respondents answered 'yes' to a survey question. In Section 4, I use modified versions of the Discrete Laplace and Discrete Gaussian Mechanisms to demonstrate the calculation of differential privacy guarantees using QIF.

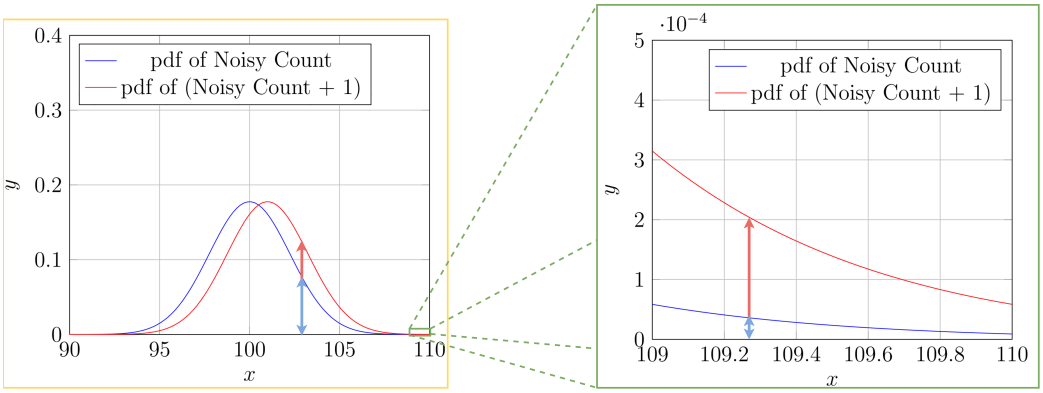

Figure 3.7: Plots demonstrating how the tail behaviour of the Gaussian distribution can only supply approximate differential privacy.

3.2 Quantitative Information Flow

3.2.1 Motivation

A long-standing goal of computer security is to control the leakage of information used in computer systems. Information such as private communications, passwords and personal data used by programs are required to be kept secure from outside view. However, the goals of these programs, such as publishing statistics from data, necessarily involve leaking some information about the data that created that output. Other aspects of programs, such as timing information and power consumption can also unintentionally leak information. For this reason, the theory of Quantitative Information Flow (QIF) was developed to provide a rigorous framework for quantifying information leakage in programs. This provides notions of security more descriptive than simply identifying whether or not programs leak information and labelling them 'secure' or 'insecure' accordingly. QIF provides numerical guarantees for the amount of information flow from a program, the extent to which it can be tolerated, and how much advantage an attacker gains from the flow of information [17].

3.2.2 Formal Definitions

The following formal definitions of the different aspects of Quantitative Information Flow are detailed in work by Alvim et. al. [17], alongside which I

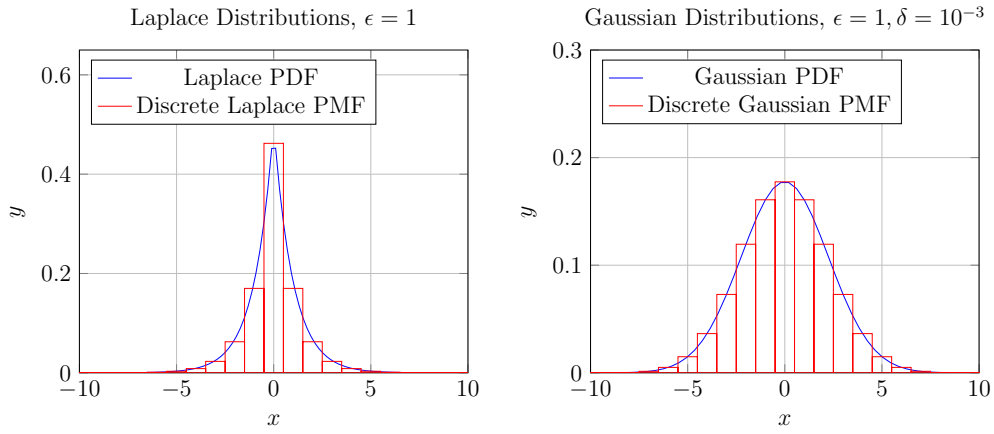

Figure 3.8: Probability density functions of the Laplace and Gaussian distributions, overlayed with the probability mass functions of their respective discrete distributions.

will explain ideas in relation to the Randomised Response Mechanism on a single response. Note, however, that many programs unrelated to differential privacy can be modelled in the QIF paradigm as well.

Secrets represent values that are of interest to an adversary, such as a person's data, or a password. In general, they are values over which there is some uncertainty. A secret π is of type $\mathbb{D}\mathcal{X}$, i.e. a probability distribution ($\mathbb{D}$) of values of type $\mathcal{X}$. $\pi_x : [0, 1]$ is the probability assigned to the secret's exact value being $x : \mathcal{X}$. In the Randomised Response Mechanism, the secret values are the possible values of the respondent's true answer, that is, 'yes' and 'no', which we'll denote as $x = 0$ and $x = 1$. As for the prior probability distribution of the secret, we assume a uniform distribution, meaning $\pi_0 = \pi_1 = 1/2$.

Mechanisms represent randomised algorithms that return observables of type $\mathcal{Y}$, which depend on secrets. A mechanism can be represented by a stochastic channel matrix (i.e. a matrix where the sum of all entries in a row is equal to 1). $C : \mathcal{X} \times \mathcal{Y} \to [0, 1]$. C_{xy} is the probability that y is observed given x is the secret. The following is the channel matrix for the Randomised Response

Mechanism:

$$RRM := \begin{array}{c} \\ \texttt{resp=0} \\ \texttt{resp=1} \end{array} \begin{array}{c} \begin{array}{cc} \texttt{result=0} & \texttt{result=1} \end{array} \\ \begin{pmatrix} 3/4 & 1/4 \\ 1/4 & 3/4 \end{pmatrix} \end{array}$$

The rows of the matrix correspond to the values of a user's response (labelled `resp`), which is the secret, and the columns correspond to the output of the mechanism (labelled `result`), which is what the adversary can observe.

Given prior distributions of secrets, we can also calculate joint distributions of secrets and observables, marginal distributions of observables and posterior distributions of secrets. Let $\pi_x|^y$ be the probability of the secret being x given y is observed/leaked by the mechanism. In the RRM example, if `result=0` is observed, the posterior distribution of the secret is $\pi_0|^0 = 3/4$ and $\pi_1|^0 = 1/4$. If `result=1` is observed, the posterior distribution of the secret becomes $\pi_0|^1 = 1/4$ and $\pi_1|^1 = 3/4$. Denoting p_y as the marginal probability of observing y, for RRM we have $p_0 = p_1 = 1/2$.

In a program, we have both observed probabilistic processes (that is, marginal distributions of observations) and unobserved probabilistic processes (that is, posterior distributions of a secret). Combining these brings us to the notion of hyper-distributions, that is, probability distributions of probability distributions, of type $\mathbb{D}^2\mathcal{X}$. When distributions are finite and mechanisms can be defined using finite matrices, hyper-distributions can be written as the weighted $\oplus$-sum of posterior distributions, where the weights are the marginal probabilities. For RRM, this would look like $1/2(\pi|^0) \oplus 1/2(\pi|^1)$.

Loss functions (often more intuitively formalised as gain functions [18]) give the cost to the adversary of taking a certain action when the secret is a certain value. The choice of loss function can depend on the exact goal and capabilities of the adversary. Measures of the uncertainty of a secret under a mechanism can be defined with respect to loss functions. One such measure of uncertainty that is commonly used is Bayes' risk [19]. Technical definitions for concepts related to loss functions are provided in Appendix A.2.1.

3.3 Differential Privacy in QIF

Previous work [2] has been conducted in relating the theory of Quantitative Information Flow to Differential Privacy. Given that differential privacy is concerned with using algorithms to control the amount of information leaked about underlying data, differential privacy can very naturally be modelled using the QIF paradigm. As demonstrated earlier with the RRM, mechanisms can be modelled as channels, where data entries are secrets and observables are the outputs of the differential privacy mechanisms.

The following definition is provided by Morgan and McIver for ϵ-differential privacy of channel matrices:

Definition 3 *A channel M is ϵ-differentially private for x, x' if for all $y \in \mathcal{Y}$,*

$$M_{xy} \leq e^{\epsilon} M_{x'y} \text{ and } M_{x'y} \leq e^{\epsilon} M_{xy}$$

If these inequalities hold for all possible pairs x, x' that correspond to neighbouring databases, then the mechanism M as a whole satisfies ϵ-differential privacy.

Channel differential privacy is also defined by Morgan and McIver in terms of loss functions, providing another way to validate the differential privacy guarantees of a mechanism. More detail is provided in Appendix A.2.2.

Using Definition 3, for RRM, letting $RRM_{x,y} = RRM_{\texttt{resp=}x,\texttt{result=}y}$, we have:

$$RRM_{0,0} = 3 \times RRM_{1,0} \text{ and } RRM_{1,1} = 3 \times RRM_{1,1}$$

Since the entries of a column differ by a maximum factor of $3 = e^{\log(3)}$, we can observe that the smallest ϵ for which RRM is ϵ-differentially private is $\epsilon = \log(3)$.

3.4 Kuifje

Kuifje is a programming language designed for Quantitative Information Flow, deeply embedded in Haskell and based on previous work by Gibbons

et. al. [5]. When a program is run in Kuifje, whenever a probabilistic step is reached, such as tossing a coin, Kuifje tracks all possible branches of computation and the probabilities of these branches. For each of the variables in the program, Kuifje also tracks the probabilities of the values the variables could take, given the observations the attacker makes.

Below is the Kuifje code for the Randomised Response Mechanism on a single user response, as in Figure 3.1:

```
1 resp <- uniform [0,1];
2 coin <- uniform [0,1];
3 result <- uniform [coin,resp];
4 leak(result);
```

Line 1 of the code uses the `uniform` function to give the user's response `resp` an even chance of being either 0 or 1. Kuifje's attacker model means that the attacker has full access to the program text, so simply setting `resp` to 0 or 1 would give the attacker full information about the variable.
Line 2 does the same as line 1, except it sets the outcome of a coin flip (`coin`) with an even chance of being 0 or 1, that is, tails or heads.
Line 3 sets the value of `result`, which will be the output of the mechanism. Using a `uniform` variable over `coin` and `resp` is equivalent to doing the first coin flip in Figure 3.1. This line gives an even chance of setting `result` to the user's response (`resp`), or the second coin flip (`coin`).
In Line 4, the `leak` function is used to reveal the `result` variable, giving the adversary complete certainty over the value of `result`. The adversary can use this result to gain information on the probabilities of the other variables, `coin` and `resp`.

When we run the code, Kuifje gives hyper-distribution output in the following format:

```
> Variable resp hyper
0.500000   0.250000   R 0.0
           0.750000   R 1.0
0.500000   0.750000   R 0.0
           0.250000   R 1.0
> condEntropy bayesVuln resp hyper
0.750000
```

These hyper-distributions are given for all variables in the program, but from the perspective of an adversary trying to gain information about the response `resp`, we only focus on the hyper-distribution for that.

The left column gives the probabilities of the sequence of observations by the attacker, which for this program are Pr[`result=1`] or p_1 and Pr[`result=0`] or p_0, respectively. Both these probabilities are `0.500000`, as discussed in 3.2.2.

The middle column gives the probabilities of the secret `resp` being equal to the values in the third column, given what the attacker observes. From the first two rows of the hyper-distribution, we see that Pr[`resp=0.0`|`result=1`] = `0.250000` and Pr[`resp=1.0`|`result=1`] = `0.750000`. This matches up with the statement in section 3.2.2 that $\pi_0|^1 = 1/4$ and $\pi_1|^1 = 3/4$.

Along with each hyper-distribution, Kuifje gives a value `condEntropy bayesVuln x hyper` for each variable. This is the Bayes' Vulnerability of the variable, which is simply 1 − Bayes' Risk.

4. Methods

4.1 Proving Approximate Differential Privacy of Programs

Whilst McIver and Morgan provide some discussion on (ϵ, δ)-differential privacy in QIF, a full definition for approximate differential privacy of a channel is introduced below, extending Definition 3 to the following:

Theorem 1 *A channel M is (ϵ, δ)-differentially private for secret values x, x' if and only if*

$$\sum_{y \in \mathcal{S}} M_{xy} \leq e^{\epsilon} \sum_{y \in \mathcal{S}} M_{x'y} + \delta \text{ and } \sum_{y \in \mathcal{S}} M_{x'y} \leq e^{\epsilon} \sum_{y \in \mathcal{S}} M_{xy} + \delta$$

for all $\mathcal{S} \subseteq \mathcal{Y}$

Proof: The theorem follows from Definition 2 for (ϵ, δ)-differential privacy, and Definition 3 for ϵ-differential privacy of a channel. Note that because we can no longer only compare individual observations (y's) of the mechanism as we do with ϵ-differential privacy, the inequalities must hold for all subsets of $\mathcal{Y}$. Hence for all $S \subseteq \mathcal{Y}$ we use the summation $\sum_{y \in \mathcal{S}} M_{xy}$ which is equivalent to $Pr[\mathcal{M}(D) \in S]$.

Using this definition, we can compare the rows of a channel matrix to obtain its differential privacy properties, as has been demonstrated with the Randomised Response Mechanism. The mechanism as a whole is (ϵ, δ)-differentially private if 1 holds for all x, x' pairs that correspond to neighbouring databases. As previously stated, while each ϵ-differentially private mechanism tends to have a unique minimum ϵ, there are many possible minimum (ϵ, δ) combinations. Therefore, for fixed ϵ we calculate the minimum δ for the mechanism, or vice-versa.

Aside from using channel matrices for determining differential privacy guarantees, McIver and Morgan also detail how to use the QIF concept of loss functions to do similarly, formulating what is referred to as the dp_ϵ loss function. Like the function for Bayes' Risk, the dp_ϵ loss function can be used

in many useful calculations for determining the unpredictability of secrets, but specifically in the context of differential privacy. The paper also suggests an extension for (ϵ, δ)-differential privacy. Technical details are provided in Appendix A.2.2. However, the following result from McIver and Morgan's paper follows from Theorem 1:

Lemma 1 *Where $v \in \mathbb{D}\mathcal{X}$ is the uniform prior distribution over secret values, a mechanism M is (ϵ, δ)-differentially private for secrets x, x' if and only if, under the mechanism,*

$$\sum_{y \in \mathcal{Y}} p_y \times \min(v_{x'}|^y \times e^\epsilon - v_x|^y, 0) \geq -\delta/2$$
$$\text{and } \sum_{y \in \mathcal{Y}} p_y \times \min(v_x|^y \times e^\epsilon - v_{x'}|^y, 0) \geq -\delta/2$$

where p_y is the marginal probability $\Pr[\text{Obervation} = y]$ *and $v_x|^y$ is the conditional probability* $\Pr[\text{Secret} = x|\text{Obervation} = y]$

Proof:

$$\sum_{y \in \mathcal{S}} M_{xy} \leq e^\epsilon \sum_{y \in \mathcal{S}} M_{x'y} + \delta \qquad \text{from Thm 1, for all } \mathcal{S} \subseteq \mathcal{Y}$$
$$\Leftrightarrow e^\epsilon \sum_{y \in \mathcal{S}} M_{x'y} - \sum_{y \in \mathcal{S}} M_{xy} \geq -\delta$$
$$\Leftrightarrow \sum_{y \in \mathcal{S}} (M_{x'y} \times e^\epsilon - M_{xy}) \geq -\delta$$
$$\Leftrightarrow \sum_{y \in \mathcal{S}} (v_{x'}|^y \times p_y/v_{x'} \times e^\epsilon - v_x|^y \times p_y/v_x) \geq -\delta \qquad \text{Bayes' Theorem}$$
$$\Leftrightarrow \sum_{y \in \mathcal{S}} (v_{x'}|^y \times p_y/0.5 \times e^\epsilon - v_x|^y \times p_y/0.5) \geq -\delta \qquad \text{since } v \text{ is uniform}$$
$$\Leftrightarrow \sum_{y \in \mathcal{S}} p_y \times (v_{x'}|^y \times e^\epsilon - v_x|^y) \geq -\delta/2$$
$$\Leftrightarrow \sum_{y \in \mathcal{Y}} p_y \times \min(v_{x'}|^y \times e^\epsilon - v_x|^y, 0) \geq -\delta/2$$

In the last line of the proof, we take $\mathcal{S} \subseteq \mathcal{Y}$ such that for all $y \in \mathcal{S}$, $(v_{x'}|^y \times e^\epsilon - v_{x'}|^y) \leq 0$, that is, the $\mathcal{S}$ that minimises the left side of the

inequality in the second last line.

This formula only requires knowledge of the marginal probabilities p_y and conditional probabilities $v_x|^y$ of the mechanism, which is exactly what is given in Kuifje output. Hence, rather than needing to convert the Kuifje output to a channel matrix, one may use Lemma 1 to calculate or validate the differential privacy properties of a mechanism directly. The formula assumes a uniform prior distribution over the secret values, which may not always be the case. For a non-uniform prior distribution, the final inequalities turn out to be

$$\sum_{y \in \mathcal{Y}} p_y \times \min(\frac{\pi_{x'}|^y}{\pi_{x'}} \times e^\epsilon - \frac{\pi_x|^y}{\pi_x}, 0) \geq -\delta$$

$$\text{and } \sum_{y \in \mathcal{Y}} p_y \times \min(\frac{\pi_x|^y}{\pi_x} \times e^\epsilon - \frac{\pi_{x'}|^y}{\pi_{x'}}, 0) \geq -\delta$$

In Section 5, I demonstrate differential privacy calculations using the formulae of both Theorem 1 and Lemma 1.

4.2 Programming Differential Privacy Mechanisms in Kuifje

4.2.1 Mechanism demonstration

As shown in Section 3.4, we can program a differential privacy mechanism in the Kuifje language. The general procedure I use to demonstrate the workings of a mechanism for a counting query is as follows, and is illustrated in 4.1:

1. Take a fixed toy database containing 1's and 0's in the form of a list. These could represent answers to 'yes'/'no' questions.

2. Add an unknown response to the database by using a uniform random variable, i.e. with 50% probability of being 0, and 50% probability of being 1. This gives two possible neighbouring databases.

3. Take the count of 1's in the database and add noise as prescribed by the mechanism. (In the case of the Randomised Response Mechanism, noise is added before taking the count).

4. Leak the noisy count value. That is, reveal this result to the adversary.

I use this procedure for the Randomised Response Mechanism and the discrete Laplace and Gaussian Mechanisms, with results discussed in Section 5. The hyper-distribution output of the unknown response is used for calculating channel matrices and analysing the differential privacy guarantees of the mechanisms. Note that, whilst the toy database is set as one fixed list, the results of the unknown response will remain the same regardless of what sequence of 1's and 0's are contained in that list. That is, the results for the mechanism generalise to any neighbouring databases as required by Definition 1 for differential privacy.

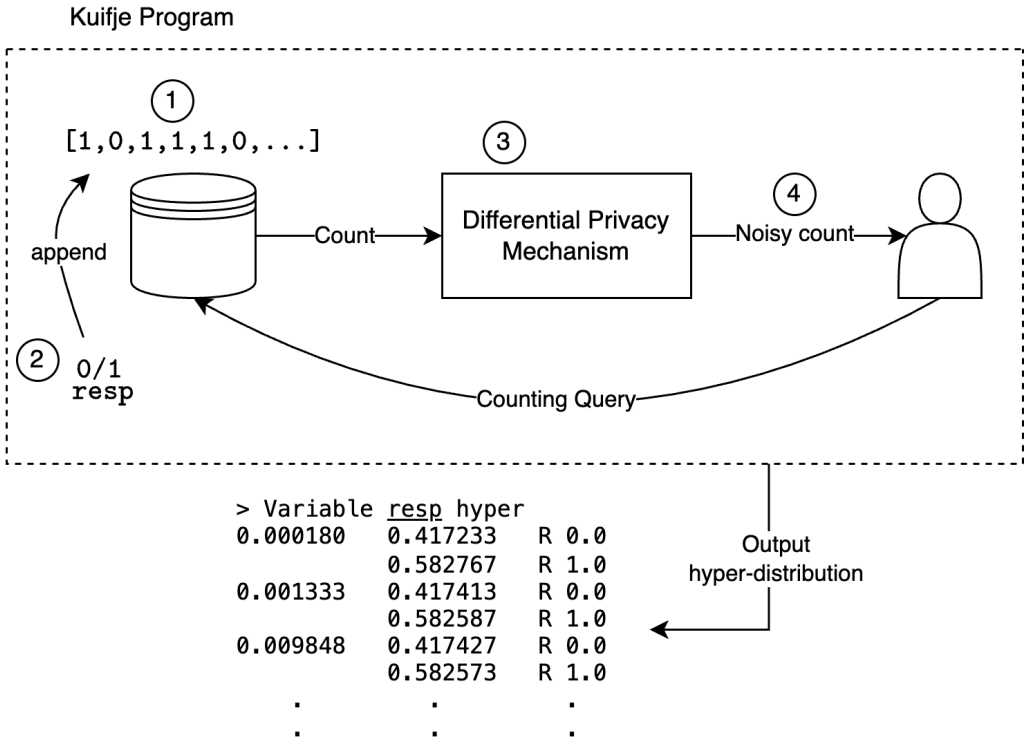

Figure 4.1: Illustrated procedure used in this report of programming differential privacy mechanisms in Kuifje.

4.2.2 Discrete Laplace and Gaussian sampling

In demonstrating the Discrete Laplace and Discrete Gaussian mechanisms, I use an adaptation of an algorithm given by Canonne et. al. [15] for exact sampling from the discrete Laplace and Gaussian distributions. Some steps were taken to approximate the algorithm, some being necessary due to the range of the distributions being infinite. An infinite distribution creates an infinite number of possible computation branches, meaning the program would fail to terminate in Kuifje. As a result, the range of the distributions was limited, resulting in the tails of the distributions being squished as the example in 4.2 shows. This truncation will cause slightly worsened differential privacy results. This is the case even in the example used in Section 5, when the range of the discrete Laplace distribution is set to $[-33, 33]$ and the distribution is narrowed with $\epsilon = 1/3$, meaning the squishing only takes effect in extremely unlikely outcomes.

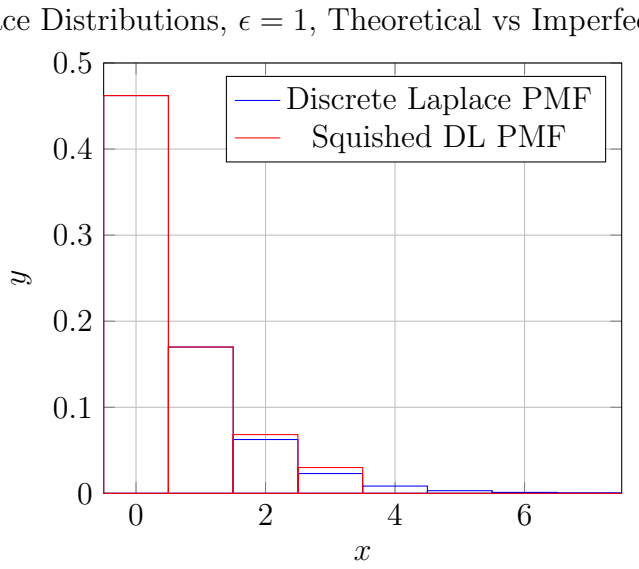

Figure 4.2: Plot showing how limiting the range of the Discrete Laplace distribution can affect its probability mass function. Mass beyond the maximum value (3 in this example) is redistributed to other values before the maximum. The same happens for the minimum value of the distribution.

Limiting the range of the distributions is particularly an issue with the Dis-

crete Laplace Mechanism, which is supposed to be ϵ-differentially private. As demonstrated in Figure 4.3, the limit means that the maximum value of the noisy count is [maximum value of distribution] + [count of 1's in database if unknown response is 1]. This noisy count value can not occur when the unknown response is 0, meaning this output value, albeit highly unlikely, gives complete certainty over which database was used as input. The same goes for the minimum value of the noisy count. This necessarily means that this version of the mechanism can not satisfy ϵ-differential privacy.

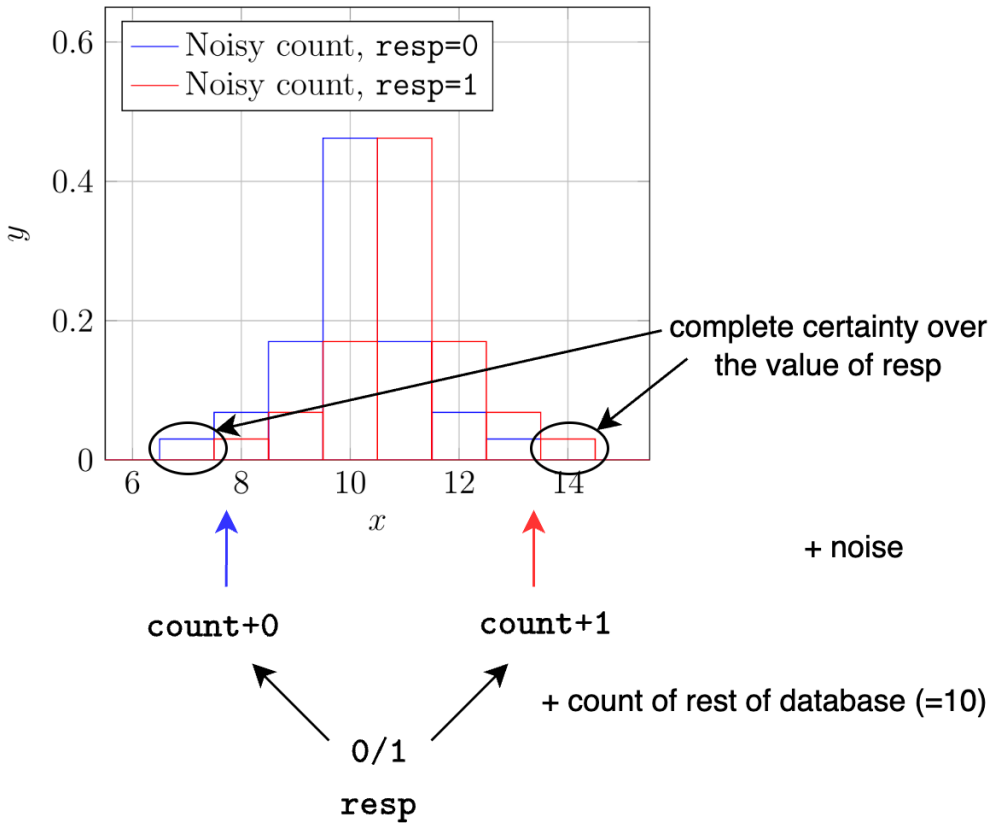

Figure 4.3: Illustration of how limiting the range of the Discrete Laplace distribution introduces a δ.

It would be useful as future work to investigate how to better program the discrete Laplace and Gaussian distributions to reduce the effects of limiting

their range, as well as to compute bounds on the difference in differential privacy results. It is at least possible to set a much wider range to the distributions, such that the effects of truncating become even more negligible, though this greatly increases computation time. Due to the infinite nature of these distributions making them impossible to truly analyse in Kuifje, the implementations of the discrete Laplace and Gaussian mechanisms can only be approximate. However, the imperfections of the distributions at least provide examples for calculating differential privacy results that one may be unsure about beforehand.

4.2.3 Calculating Differential Privacy Results

As explained in 3.4, the output Kuifje provides is in the form of hyper-distributions, containing the probabilities $\Pr[\text{observation} = y]$ (or p_y) and $\Pr[\text{secret} = x | \text{observation} = y]$ (or $\pi_x|^y$).

In the case of our unknown response, which we call `resp`, with `noisyCount` being what is leaked to the adversary, the Kuifje output provides values for $\Pr[\texttt{noisyCount} = y]$ and $\Pr[\texttt{resp} = x | \texttt{noisyCount} = y]$. Using Bayes' theorem, we can calculate the corresponding channel matrix value for x and y, with each entry calculated as the following:

$$M_{xy} = \Pr[\texttt{noisyCount} = y | \texttt{resp} = x] = \frac{\Pr[\texttt{resp}{=}x | \texttt{noisyCount}{=}y] \Pr[\texttt{noisyCount}{=}y]}{\Pr[\texttt{resp}{=}x]}$$

Using this equation, we can calculate all values of the channel matrix M. Note that we do also require the prior probability of the secret, $\Pr[\texttt{resp} = x]$. In our case, since `resp` is drawn uniformly from 0 and 1, we always have $\Pr[\texttt{resp} = x] = 0.5$. In the general case, if these calculations are to be automated, Kuifje would need to know this prior distribution, which can be obtained from when the secret is instantiated or given as input to the program.

Rearranging the inequalities of Definition 3 or Theorem 1 as necessary, one may validate that a mechanism for which we have the channel matrix is differentially private for given ϵ and δ values. For example, using Definition 3, we can find the maximum ratio between M_{xy} and $M_{x'y}$ for $y \in \mathcal{Y}$ to find the minimum ϵ for which M satisfies ϵ-differential privacy (in the case that no ratios are infinite).

One goal of interest is finding the minimum δ for which a channel matrix satisfies (ϵ, δ)-differential privacy, for a fixed ϵ. Using the inequalities in Theorem 1, one may examine all columns (y's) of the matrix that do not satisfy ϵ differential privacy. We can separate the cases where $x > x'$ and $x' > x$, and use each of these subsets ($\mathcal{S}$) to calculate $\delta \geq \sum_{y \in \mathcal{S}} M_{xy} - e^{\epsilon} \sum_{y \in \mathcal{S}} M_{x'y}$ and $\delta \geq \sum_{y \in \mathcal{S}} M_{x'y} - e^{\epsilon} \sum_{y \in \mathcal{S}} M_{xy}$, to give the minimum δ.

Alternatively, given in our examples `resp` is drawn from a uniform distribution, we can also use Lemma 1 to calculate the same properties, without needing to convert to channel matrices. The result also means no work needs to be done to find for which observations the mechanism violates ϵ-differential privacy for a fixed ϵ since that is taken care of by the min() function.

In the following section, I demonstrate such analysis on the Randomised Response Mechanism and the Discrete Laplace and Gaussian Mechanisms.

5. Results and Discussion

5.1 Randomised Response Mechanism

I used the methods described in Section 4.2.1 for programming the Randomised Response mechanism in Kuifje and in Section 4.2.3 for calculating differential privacy results using Theorem and Lemma 1. The channel matrix extracted from the output for the Randomised Response Mechanism, as displayed in Appendix A.3.2, has the same values as RRM in Section 3.3. $\epsilon = \log(3)$ is the smallest ϵ for which this channel is ϵ-differentially private, which matches the theoretical result. Hence, the proposed method works as expected for simple ϵ-differential privacy calculations on discrete finite mechanisms.

5.2 Discrete Laplace and Gaussian

I programmed both the Discrete Laplace and Gaussian Mechanisms in Kuifje using the method in Section 4.2.1, with sampling from the distributions as described in Section 4.2.2. Full code is given in Appendix A.4.1.

The resulting calculations in Appendix A.4.2 show that for the Discrete Laplace Mechanism with parameters that would theoretically provide $\frac{1}{3}$-differential privacy, we instead get a weaker $(\frac{1}{3}, 0.000007)$-differential privacy. As for the Discrete Gaussian mechanism with parameters that would provide $(1, 0.007)$-differential privacy, we instead get $(1, 0.007247)$-differential privacy. These results are expected and can be attributed to the squeezing of the distributions in our sampling method, showing that the methods correctly evaluate the (ϵ, δ)-differential privacy guarantees for the programmed mechanisms. The change in δ between the theoretical guarantee and the Kuifje result is very small, especially in the Discrete Laplace example. Increasing the support of the distribution used could decrease this difference much further. However, these results demonstrate that the differential privacy guarantees of a discrete infinite mechanism can not be calculated exactly with Kuifje, but can only be closely approximated.

6. Conclusion

6.1 Summary

In this project, I have addressed the aims in Section 1.2 as follows: For Aim 1, I have provided formal definitions for approximate differential privacy as a problem in QIF. This includes the definition for (ϵ, δ)-differential privacy of a channel matrix, as in Theorem 1, which allows for the rows of a channel matrix to be compared to validate differential privacy guarantees. I have also introduced Lemma 1, which can be used to calculate (ϵ, δ)-differential privacy guarantees directly using hyper-distributions of secrets in QIF.

Addressing Aim 2, I have programmed differential privacy mechanisms in Kuifje and demonstrated how results from Kuifje can be used to calculate or validate differential privacy guarantees of mechanisms. The mechanisms that were demonstrated in this paper were specifically the Randomised Response, Discrete Laplace and Discrete Gaussian Mechanisms. The differential privacy results of the coded mechanisms were successfully calculated using the methods made when addressing Aim 1. However, a major limitation of Kuifje is the inability to handle distributions with infinite support, meaning that the discrete Laplace and Gaussian distributions used for their respective mechanisms can only be approximated. These approximations may only affect results in the tails of distributions, where probabilities of outcomes are extremely unlikely. However, this approximating can cause a mechanism that satisfies pure differential privacy to only satisfy approximate differential privacy, as seen with the discrete Laplace mechanism. Therefore, the methods introduced can be used to precisely validate or calculate the differential privacy properties of discrete finite mechanisms, and approximately validate or calculate the properties of discrete infinite mechanisms. It would be impractical to attempt these methods on continuous mechanisms, such as the continuous Gaussian or Laplace mechanisms.

The links between QIF and differential privacy, including for approximate differential privacy and the work demonstrating the ability of a QIF-aware language to guarantee differential privacy lay out a foundation for providing these guarantees in more complex differential privacy implementations. This

work can ultimately contribute to ensuring that differential privacy is faithfully implemented in programming, providing confidence that data used in these implementations is kept secure.

6.2 Future work

As mentioned, using the methods described in this paper to determine the differential privacy guarantees of programs can be automated, and for further work, such methods could be added as an extension to Kuifje. This implementation could allow for fast validation of any differential privacy mechanism programmed in the language.

To address the issue of discrete infinite distributions having their range limited in Kuifje, it would be ideal to investigate how to provide bounds on the differential privacy guarantees when these approximations are used. It would also be beneficial to develop better approximations of the discrete Laplace and Gaussian distributions, possibly by including them as primitives that use the probability mass functions of the distributions.

Several other useful concepts in the study of differential privacy would be beneficial to encode in Kuifje. These include: multi-valued queries, which can greatly affect the sensitivity of the query, that is, the amount the query's output can change due to changes in the input; composition schemes, which involve releasing multiple results of differential privacy mechanisms on the same data [8, 20], which can affect differential privacy properties; and the exponential mechanism [21], which is useful for queries involving reporting an element with a maximum (noisy) score, which can be used as a foundation for many ϵ-differentially private mechanisms.

Another area of interest is using relaxations of differential privacy beyond (ϵ, δ)-differential privacy since there are a couple of drawbacks to the definition. These include the fact that some mechanisms, such as the Gaussian Mechanism, satisfy a curve of minimum (ϵ, δ)-differential privacy pairs rather than a single (ϵ, δ) pair, as well as the lack of descriptiveness on the behaviour of the mechanism that causes the delta. For example, in practice, δ is never the probability that the whole database is leaked, as it is in the Catastrophe Mechanism, yet the definition allows for that. Other differential privacy re-

laxations that alleviate these shortfalls have been proposed and studied, such as Rényi differential privacy [22] and concentrated differential privacy [23]. It would therefore be helpful to look into how differential privacy guarantees could be validated under these definitions in Kuifje, as we have demonstrated with ϵ- and (ϵ, δ)-differential privacy. Another relaxation is metric differential privacy [24], which uses a distance metric between secrets, allowing differential privacy to be defined not only for databases differing by one entry.

Differential privacy is proving to have many useful applications, including in the ever-growing area of machine learning. There is much investigation into using differential privacy to protect the data used for training machine learning models [25]. Other interesting work has been done linking differential privacy to robustness to adversarial inputs in machine learning models [26]. Here, differential privacy formalisms are used to be robust against changes in image classification predictions, where human imperceptible noise is added. For these uses, it would be of interest to demonstrate the workings of these differential privacy properties in QIF with small examples of machine learning models.

A. Appendix

A.1 Laplace and Gaussian Distribution Definitions

Continuous Laplace distribution Lap(ϵ) probability density function:

$$f(x) = \frac{\epsilon}{2} e^{-\epsilon|x|}, \quad x \in \mathbb{R}$$

Continuous Gaussian distribution N(μ, σ) probability density function:

$$f(x) = (1/\sqrt{2\pi\sigma^2}) e^{-(x-\mu)^2/2\sigma^2}, \quad x \in \mathbb{R}$$

Discrete Laplace distribution $(Lap)_{\mathbb{Z}}(\epsilon)$ probability mass function:

$$f(x) = \frac{e^{\epsilon} - 1}{e^{\epsilon} + 1} \cdot e^{|x|}, \quad x \in \mathbb{Z}$$

Discrete Gaussian $(N)_{\mathbb{Z}}(\mu, \sigma)$ distribution probability mass function:

$$f(x) = \frac{e^{-(x-\mu)^2/2\sigma^2}}{\sum_{y \in \mathbb{Z}} e^{-(y-\mu)^2/2\sigma^2}}, \quad x \in \mathbb{Z}$$

For $(N)_{\mathbb{Z}}(\mu, \sigma)$, when Sensitivity=1, the Discrete Gaussian Mechanism satisfies (ϵ, δ)-differential privacy for

$$\delta = \mathop{\mathbb{P}}_{Y \leftarrow \mathrm{N}_{\mathbb{Z}}(0,\sigma^2)}[Y > \epsilon\sigma^2 - \frac{1}{2}] + e^{\epsilon} \cdot \mathop{\mathbb{P}}_{Y \leftarrow \mathrm{N}_{\mathbb{Z}}(0,\sigma^2)}[Y > \epsilon\sigma^2 + \frac{1}{2}]$$

A.2 Loss Functions and Uncertainty

A.2.1 Loss Function and Uncertainty Definitions

In general, where $\mathcal{W}$ is a set of actions an adversary could take (usually guessing that the secret is a particular value in $\mathcal{X}$ or that secret satisfies a certain property):

Definition 4 *A* loss function $\ell : \mathcal{W} \times \mathcal{X} \longrightarrow \mathbb{R}$ *is a real-valued function such that* $l(w, x)$ *determines the loss to an adversary if he chooses* w *and the secret is* x*.*

Loss functions provide a measure for the uncertainty of a secret under a mechanism.

Definition 5 *Let* $\ell : \mathcal{W} \times \mathcal{X} \longrightarrow \mathbb{R}$ *be a loss function and* $\pi : \mathbb{D}\mathcal{X}$ *be a secret. The* uncertainty $U_\ell[\pi]$ *of the* π *with respect to* ℓ *is:*

$$U_\ell[\pi] \quad := \quad \min_{w \in \mathcal{W}} \sum_{s \in \mathcal{X}} \ell(w, x) \times \pi_x$$

The average posterior uncertainty of a secret can be defined as follows:

Definition 6 *Where* C *is a mechanism, and* $\pi\rangle C$ *denotes the joint distribution over* $\mathcal{X} \times \mathcal{Y}$*, the* average posterior uncertainty *of the secret is*

$$U_\ell[\pi\rangle C] \quad := \quad \sum_{y \in \mathcal{Y}} p_y \times U_\ell[\pi|^y]$$

Bayes' Risk can be defined using the following loss function:

$$\mathsf{br}(x, x') \quad := \quad 0 \text{ if } x = x' \quad else 1.$$

Definition 7 (Bayes' Risk) *For a prior distribution,*

$$\text{Bayes' Risk } = U_{\mathit{br}}[\pi] = 1 - \max_{x \in \mathcal{X}} \pi_x$$

Under a mechanism,

$$\text{Bayes' Risk } = U_{\mathit{br}}[\pi\rangle C] = \sum_{y \in \mathcal{Y}} p_y U_{\mathit{br}}[\pi|^y]$$

In the *RRM* example, assuming a uniform prior distribution, $U_{\mathsf{br}}[\pi] = 1/2$ and $U_{\mathsf{br}}[\pi\rangle RRM] = (1/2)\times(3/4)+(1/2)\times(3/4) = 1/4$. $U_{\mathsf{br}}[\pi\rangle RRM] < U_{\mathsf{br}}[\pi]$ tells us that the mechanism decreases the uncertainty of the secret's value.

A.2.2 Loss Functions and Uncertainty for Guaranteeing Differential Privacy

The following loss function is suggested by Morgan and McIver for proving (ϵ, δ)-differential privacy of programs:

Definition 8 *Given are $\epsilon > 0$, $\delta \geq 0$ and $\mathcal{W} := (x, x'), \star$. dp_{ϵ^*}, an ϵ-differentially private loss function relative to $\mathcal{W}$ is defined:*

$$\begin{aligned} dp_{\epsilon^*}(w, x) &= -1, && \text{if } w \neq \star \wedge \overleftarrow{w} = x \\ dp_{\epsilon^*}(w, x) &= e^{\epsilon}, && \text{if } w \neq \star \wedge \overrightarrow{w} = x \\ dp_{\epsilon^*}(w, x) &= 0, && \text{if } w = \star \wedge \overleftarrow{w} \neq x \neq \overrightarrow{w} \\ dp_{\epsilon^*}(\star, x) &= 0. \end{aligned}$$

The average posterior uncertainty of the loss function gives the same result as in Lemma 1:

Theorem 2 *Assuming a uniform prior distribution v over secret values x, x', a mechanism M is (ϵ, δ)-differentially private for x, x' if and only if*

$$U_{dp_{\epsilon^*}}[v\rangle M] = \sum_{y \in \mathcal{Y}} p_y \times \min(v_x|^y \times e^{\epsilon} - v_{x'}|^y, 0) \geq -\delta/2$$

M is (ϵ, δ)-differentially private if $U_{dp_{\epsilon^*}} \geq -\delta/2$ for all such pairs x, x'

A.3 Randomised Response Mechanism in Kuifje

A.3.1 Code and Output

```
database = [0,1,0,1,1,0,1,0,0]; // Database of 0's and 1's
resp <- uniform [0, 1]; // New response to add to database
coin <- uniform [0, 1]; // Coin toss

// Add data to database depending on coin and resp
new_data <- uniform [resp, coin];
database.append(new_data);

// Query the count of 1's in the database
```

```
count = 0;
for r in database:
  count = count + r;
leak(count);
```

And the corresponding output hyper-distribution for `resp` (which is identical to when we use a single respondent in 3.4):

```
> Variable resp hyper
0.500000   0.250000   R 0.0
           0.750000   R 1.0
0.500000   0.750000   R 0.0
           0.250000   R 1.0
```

A.3.2 Differential Privacy calculations

Using Bayes' formula calculations as described in 4.2.3, we get the following channel matrix for the Randomised Response Mechanism:

$$RRM := \begin{array}{c} \\ \texttt{resp=0} \\ \texttt{resp=1} \end{array} \begin{array}{c} \begin{array}{cc} \texttt{count=4} & \texttt{count=5} \end{array} \\ \begin{pmatrix} 0.75 & 0.25 \\ 0.25 & 0.75 \end{pmatrix} \end{array}$$

Which, as explained in Section 3.3 can be shown to be log(3)-differentially private.

Alternatively, using Lemma 1, with $x = 0$ and $x' = 1$ and setting $\epsilon = \log(3)$.

$$\begin{aligned} \sum_{y \in \mathcal{Y}} p_y \times \min(v_{x'}|^y \times e^\epsilon - v_x|^y, 0) &= 0.5 \times \min(0.75 \times 3 - 0.25, 0) \\ &\quad + 0.5 \times \min(0.25 \times 3 - 0.75, 0) \\ &= 0 + 0 \\ &= 0 \end{aligned}$$

and

$$\begin{aligned} \sum_{y \in \mathcal{Y}} p_y \times \min(v'_x|^y \times e^\epsilon - v_{x'}|^y, 0) &= 0.5 \times \min(0.25 \times 3 - 0.75, 0) \\ &\quad + 0.5 \times \min(0.75 \times 3 - 0.25, 0) \\ &= 0 + 0 \\ &= 0 \end{aligned}$$

Hence, the mechanism is log(3)-differentially private.

A.4 Discrete Laplace and Gaussian in Kuifje

A.4.1 Code

The following is the code for the Discrete Gaussian Mechanism on a toy database containing 33 1's, 30 0's and 1 unknown response `resp`). The Discrete Laplace Mechanism code is the same, except the outer loop is removed, and we take the first calculation of Z as the Discrete Laplace sample. In the Discrete Laplace Mechanism, $\mathtt{t} = 1/\epsilon$. The range of the distribution being sampled from is reduced to [-33,33].

```
// This program demonstrates the use of the Discrete Gaussian
// Mechanism for providing (epsilon,delta)-dp on a counting query
// on a database with N entries

// Input: parameter variance sigma^2
// Output: Z = one sample from N_Z(0,sigma^2)
e = 2.71828182845904523536028747135266249775 72;

sigma = 2; t = (sigma div 1) + 1;

C = 0; l = 0; Z = 0; B = 1; U = 0; V = 0; i = 0;

while (l < 10):

  B = 1 if (C == 0) else B;
  U = 0 if (C == 0) else U;
  V = 0 if (C == 0) else V;
  i = 0 if (C == 0) else i;

  while (i < 10):
    t1 = 0 if (B == 1) && (U == 0) && (V == 0) else 1;

    // Loop to calculate the probability distribution for U in
    // GenerateU.kf
    // U <- (0 [0.622459] 1) if t1 == 0 else U; // t = 2
    U <- ([ 0.0 @ 0.448441
        , 1.0 @ 0.321322
        , 2.0 @ 0.230237 ]) if t1 == 0 else U; // t = 3

    // Loop to calculate the probability distribution for V in
```

```
    // GenerateV.kf
    V <- ([ 0.0 @ 0.632121
        , 1.0 @ 0.232544
        , 2.0 @ 0.085548
        , 3.0 @ 0.031471
        , 4.0 @ 0.011578
        , 5.0 @ 0.004259
        , 6.0 @ 0.001567
        , 7.0 @ 0.000576
        , 8.0 @ 0.000212
        , 9.0 @ 0.000078
        , 10.0@ 0.000046 ]) if t1 == 0 else V;

    B <- (0 [0.5] 1) if t1 == 0 else B;
    i++;

  Z = (1-(2*B))*(U+(t*V)); // Lap_Z(t) sample

  magZ = (-1 * Z) if Z < 0 else Z;

  C <- 1 [e^(-((magZ-((sigma^2)/t))^2)/(2*(sigma^2)))] 0 if C==0
                                                         else 1;
  l++;


// Create database (63 entries, 33 1's, 1 random resp)
count = 0;
Resps = [1,0,1,1,1,0,0,0,0,0,1,0,1,0,0,0,1,0,0,0,0,1,1,1,0,1,1,1,
         0,1,0,0,1,0,1,1,0,1,1,1,0,0,1,1,1,1,0,1,1,0,1,0,1,1,1,0,
         1,0,1,0,1,0,0];
resp <- 0 [0.5] 1;
Resps.append(resp);

for r in Resps:
  count = count + r;

noisyCount = count + Z;
leak(noisyCount);
```

A.4.2 Differential Privacy calculations

Discrete Laplace Mechanism

If we are to find the minimum δ for which we get $(\frac{1}{3}, \delta)$-differential privacy, we need know for what y

$$\Pr[\text{observation} = y | \texttt{resp} = 1] > e^{\frac{1}{3}} \times \Pr[\text{observation} = y | \texttt{resp} = 0]$$

For any given y, $\Pr[\text{observation} = y]$ and $\Pr[\text{secret} = x]$ are constant (since the secret is uniform), so to calculate δ we only need to consider entries in the hyper-distribution for which $\Pr[\texttt{resp} = 1 | \text{observation} = y] > e^{\frac{1}{3}} \times \Pr[\texttt{resp} = 0 | \text{observation} = y]$.

We look at where ratio of probabilities of `resp` values given an observation is greater than $\exp(\frac{1}{3})$. This is where the greater of the two probabilities of the `resp` value is greater than $\frac{e^{\frac{1}{3}}}{1+e^{\frac{1}{3}}} \approx 0.582570$. Using the terms in Lemma 1, this is where $\min(v_{x'}|^y \times e^\epsilon - v_x|^y, 0) \neq 0$ or $\min(v_x|^y \times e^\epsilon - v_{x'}|^y, 0) \neq 0$. Note that for pure $\frac{1}{3}$-differential privacy, we would have no probabilities for a `resp` value given an observation to be greater than 0.582570. Here we look at the entries in the output where we have the conditional probabilities for `resp` $= 1$ greater than 0.582570:

```
0.000180   0.417233   R 0.0
           0.582767   R 1.0
0.001333   0.417413   R 0.0
           0.582587   R 1.0
0.009848   0.417427   R 0.0
           0.582573   R 1.0
    .          .          .
    .          .          .
-- entries that satisfy 1/3-dp --
    .          .          .
    .          .          .
0.000003   1.000000   R 1.0
```

Using Lemma 1, with $x = 1$ and $x' = 0$ and setting $\epsilon = \frac{1}{3}$,

$$\begin{aligned}\sum_{y\in\mathcal{Y}} p_y \times \min(v_{x'}|^y \times e^\epsilon - v_x|^y, 0) &= 0.000180 \times (0.417233 \times e^{\frac{1}{3}} - 0.582767)\\ &\quad + 0.001333 \times (0.417413 \times e^{\frac{1}{3}} - 0.582587)\\ &\quad + 0.009848 \times (0.417413 \times e^{\frac{1}{3}} - 0.582573)\\ &\quad + 0.000003 \times (0 \times e^{\frac{1}{3}} - 1.000000)\\ &\quad \approx -0.0000036\end{aligned}$$

Note that, due to symmetry, $\sum_{y\in\mathcal{Y}} p_y \times \min(v_{x'}|^y \times e^\epsilon - v_x|^y, 0) = \sum_{y\in\mathcal{Y}} p_y \times \min(v_x|^y \times e^\epsilon - v_{x'}|^y, 0)$. Hence, the smallest δ for which the mechanism satisfies $(\frac{1}{3}, \delta)$-differential privacy is $\delta \approx 0.0000036 \times 2 \approx 0.000007$

Alternatively, using Theorem 1, with Bayes' theorem as described above, we get the following values of the mechanism's channel matrix for the cases we have conditional probabilities greater than 0.582570:

$$M_{dl} = \begin{matrix}\texttt{resp=0}\\ \texttt{resp=1}\end{matrix}\begin{pmatrix}\cdots & 0.000150 & 0.001113 & 0.008222 & 0.000000\\ \cdots & 0.000210 & 0.001553 & 0.011474 & 0.000006\end{pmatrix}$$

With $\mathcal{S}$ being the set of these observations, we have $\sum_{y\in\mathcal{S}} M_{1y} - e^{\frac{1}{3}} \sum_{y\in\mathcal{S}} M_{0y} \approx 0.000007$, meaning the smallest δ for which we have $(\frac{1}{3}, \delta)$-differential privacy is $\delta \approx 0.000007$.

Again, that we could look at hyper-distribution entries for which $\Pr[\texttt{resp} = 0|\text{observation} = y] > e^{\frac{1}{3}} \times \Pr[\texttt{resp} = 1|\text{observation} = y]$, and will get the same results (except flipped) due to symmetry.

Discrete Gaussian Mechanism

Using the same method in A.4.2, we get the following Kuifje distribution for `resp`, for all the entries that fail to satisfy 1-differential privacy (instead of $\frac{1}{3}$-differential privacy). We also ignore all observations with 0.000000 probability (i.e. $< 5 \times 10^{-7}$.):

```
    .          .          .
-- entries that satisfy 1-dp --
```

```
    .          .          .
0.000004   0.085099   R 0.0
           0.914901   R 1.0
0.000037   0.106690   R 0.0
           0.893310   R 1.0
0.000252   0.132964   R 0.0
           0.867036   R 1.0
0.001326   0.164516   R 0.0
           0.835484   R 1.0
0.005490   0.201813   R 0.0
           0.798187   R 1.0
0.017879   0.245085   R 0.0
           0.754915   R 1.0
    .          .          .
-- entries that satisfy 1-dp --
    .          .          .
```

We get the following calculations using Lemma 1 with $x = 1$ and $x' = 0$ and setting $\epsilon = 1$,

$$\begin{aligned}\sum_{y\in\mathcal{Y}} p_y \times \min(v_{x'}|^y \times e^\epsilon - v_x|^y, 0) &= 0.000004 \times (0.085099 \times e^1 - 0.914901)\\ &+ 0.000037 \times (0.106690 \times e^1 - 0.893310)\\ &+ 0.000252 \times (0.132964 \times e^1 - 0.867036)\\ &+ 0.001326 \times (0.164516 \times e^1 - 0.835484)\\ &+ 0.005490 \times (0.201813 \times e^1 - 0.798187)\\ &+ 0.017879 \times (0.245085 \times e^1 - 0.754915)\\ &\approx -0.0036236\end{aligned}$$

Hence, the smallest δ for which the mechanism satisfies $(1, \delta)$-differential privacy is $\delta \approx 0.0036236 \times 2 \approx 0.007247$

We get the following corresponding values of the mechanism's channel matrix:

$$M_{dg} = \begin{matrix}\texttt{resp=0}\\ \texttt{resp=1}\end{matrix}\begin{pmatrix}\cdots & 0.000001 & 0.000008 & 0.000067 & 0.000436 & 0.002216 & 0.008763\\ \cdots & 0.000007 & 0.000066 & 0.000437 & 0.002216 & 0.008764 & 0.026993\end{pmatrix}$$

And as we did with the discrete Laplace Mechanism, we get $\delta \geq \sum_{y\in\mathcal{S}} M_{1y} - e\sum_{y\in\mathcal{S}} M_{0y} \approx 0.038483 - 0.031238 = 0.007247$, meaning 0.007247 is the

smallest value of δ for which the mechanism satisfies $(\frac{1}{3}, \delta)$-differential privacy, which is slightly worse than the theoretical 0.07.

Bibliography

[1] Cynthia Dwork. "Differential privacy". In: *International colloquium on automata, languages, and programming*. Springer. 2006, pp. 1–12.

[2] Mário S Alvim, Konstantinos Chatzikokolakis, Pierpaolo Degano, and Catuscia Palamidessi. "Differential privacy versus quantitative information flow". In: *arXiv preprint arXiv:1012.4250* (2010).

[3] Mário S Alvim, Miguel E Andrés, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. "On the relation between differential privacy and quantitative information flow". In: *Automata, Languages and Programming: 38th International Colloquium, ICALP 2011, Zurich, Switzerland, July 4-8, 2011, Proceedings, Part II 38*. Springer. 2011, pp. 60–76.

[4] Annabelle McIver and Carroll Morgan. "Proving that programs are differentially private". In: *Asian Symposium on Programming Languages and Systems*. Springer. 2019, pp. 3–18.

[5] Jeremy Gibbons, Annabelle McIver, Carroll Morgan, and Tom Schrijvers. "Quantitative information flow with monads in haskell". In: *Foundations of Probabilistic Programming* (2019).

[6] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. "Our data, ourselves: Privacy via distributed noise generation". In: *Advances in Cryptology-EUROCRYPT 2006: 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28-June 1, 2006. Proceedings 25*. Springer. 2006, pp. 486–503.

[7] Stanley L Warner. "Randomized response: A survey technique for eliminating evasive answer bias". In: *Journal of the American Statistical Association* 60.309 (1965), pp. 63–69.

[8] 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*. Springer. 2006, pp. 265–284.

[9] Latanya Sweeney. "Simple demographics often identify people uniquely". In: *Health (San Francisco)* 671.2000 (2000), pp. 1–34.

[10] Arvind Narayanan and Vitaly Shmatikov. "Robust de-anonymization of large sparse datasets". In: *2008 IEEE Symposium on Security and Privacy (sp 2008)*. IEEE. 2008, pp. 111–125.

[11] Graeme Blair, Kosuke Imai, and Yang-Yang Zhou. "Design and analysis of the randomized response technique". In: *Journal of the American Statistical Association* 110.511 (2015), pp. 1304–1319.

[12] Jaewoo Lee and Chris Clifton. "How much is enough? choosing ε for differential privacy". In: *Information Security: 14th International Conference, ISC 2011, Xi'an, China, October 26-29, 2011. Proceedings 14*. Springer. 2011, pp. 325–340.

[13] Cynthia Dwork, Aaron Roth, et al. "The algorithmic foundations of differential privacy". In: *Foundations and Trends in Theoretical Computer Science* 9.3–4 (2014), pp. 211–407.

[14] Arpita Ghosh, Tim Roughgarden, and Mukund Sundararajan. "Universally utility-maximizing privacy mechanisms". In: *Proceedings of the forty-first annual ACM symposium on Theory of computing*. 2009, pp. 351–360.

[15] Clément L Canonne, Gautam Kamath, and Thomas Steinke. "The discrete gaussian for differential privacy". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 15676–15688.

[16] Ilya Mironov. "On significance of the least significant bits for differential privacy". In: *Proceedings of the 2012 ACM conference on Computer and communications security*. 2012, pp. 650–661.

[17] Mário S Alvim, Konstantinos Chatzikokolakis, Annabelle McIver, Carroll Morgan, Catuscia Palamidessi, and Geoffrey Smith. *The Science of Quantitative Information Flow*. Springer, 2020.

[18] Mário S Alvim, Kostas Chatzikokolakis, Catuscia Palamidessi, and Geoffrey Smith. "Measuring information leakage using generalized gain functions". In: *2012 IEEE 25th Computer Security Foundations Symposium*. IEEE. 2012, pp. 265–279.

[19] Geoffrey Smith. "On the foundations of quantitative information flow". In: *International Conference on Foundations of Software Science and Computational Structures*. Springer. 2009, pp. 288–302.

[20] Frank D McSherry. "Privacy integrated queries: an extensible platform for privacy-preserving data analysis". In: *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*. 2009, pp. 19–30.

[21] Frank McSherry and Kunal Talwar. "Mechanism design via differential privacy". In: *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*. IEEE. 2007, pp. 94–103.

[22] Ilya Mironov. "Rényi differential privacy". In: *2017 IEEE 30th computer security foundations symposium (CSF)*. IEEE. 2017, pp. 263–275.

[23] Cynthia Dwork and Guy N Rothblum. "Concentrated differential privacy". In: *arXiv preprint arXiv:1603.01887* (2016).

[24] Konstantinos Chatzikokolakis, Miguel E Andrés, Nicolás Emilio Bordenabe, and Catuscia Palamidessi. "Broadening the scope of differential privacy using metrics". In: *Privacy Enhancing Technologies: 13th International Symposium, PETS 2013, Bloomington, IN, USA, July 10-12, 2013. Proceedings 13*. Springer. 2013, pp. 82–102.

[25] Bargav Jayaraman and David Evans. "Evaluating differentially private machine learning in practice". In: *28th USENIX Security Symposium (USENIX Security 19)*. 2019, pp. 1895–1912.

[26] Mathias Lecuyer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, and Suman Jana. "Certified robustness to adversarial examples with differential privacy". In: *2019 IEEE symposium on security and privacy (SP)*. IEEE. 2019, pp. 656–672.