Guaranteeing Differential Privacy using Quantitative Information Flow Analysis

COMP90055 Research Project

Alec Miller

Submitted in partial fulfilment of the
requirements of the degree of
Master of Information Technology

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, it is critical for these insights to be gained in a manner such that the information of each 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 protecting 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 coded examples of differential privacy programs in Kuifje, a programming language built for QIF. The techniques introduced in this report allow for automatic calculation and validation of differential privacy guarantees of mechanisms, providing assurance that data used in these mechanisms supply desirable 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 project is x words in length (excluding text in images, bibliographies and appendices).

This is my own original work, under the supervision of Dr Christine Rizkallah. I would like to thank Christine for... I would also like to thank Vincent Jackson...

Contents

1. Research Summary

– Yet to write –

2. Introduction

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, such as by enhancing and optimising business decision making, improving healthcare services, and detecting fraud in financial systems. However, it is critical for these insights to be gained in a manner such that the information of each individual can not be inferred. With regular news of significant data leakages, it is a pressing need to develop methods to provide assurance that personal data is secure.

Differential privacy provides a rigorous framework for guaranteeing privacy in programs that use sensitive data, ensuring that a balance can be struck between accurate analysis and protecting personal information. 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. 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 the programs. QIF has intuitive links to differential privacy, and it has long been discussed how to analyse differential privacy mechanisms using ideas from QIF [1].

In Section 4, I extend recent work [2] on how QIF can be used to analyse differential privacy mechanisms. The authors of this paper theorised that such analysis could be conducted using Kuifje [3], a programming language recently developed for QIF. Using Kuifje will allow for differential privacy properties of mechanisms to be automatically calculated or validated, ultimately providing assurance that the use of these mechanisms provide acceptable guarantees about the privacy levels of data. I also extend the ideas presented in [2] to fit a less stringent definition of differential privacy, namely approximate differential privacy. 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 on the Gaussian Mechanism, as well as other prevalent differentially private mechanisms,

namely the Randomised Response Mechanism and the Laplace Mechanism.

3. Background

3.1 Differential Privacy Background

3.1.1 Motivation and Foundations

In today's society, ever-increasing amounts of sensitive personal data is collected from individuals and analysed, from health records to financial transactions to survey responses. Insights gained from collections of data can be of great benefit to the public. However, it is critical for these insights to be gained in such a manner that an individual can trust that their personal data doesn't fall into the wrong hands and cause them to become a victim of cybercrime.

A lot of techniques for data privacy revolve around keeping data anonymous, with the idea that if a malicious actor was to gain access to personal data, no key identifying information would be available for linking data to an individual. However, as easy as it may sound, anonymised data is surprisingly vulnerable. In what is known as a linkage attack, an attacker uses data from an auxiliary source to match people's data public data in one source to their anoymised data in the other. Studies have shown that very few characteristics are needed to uniquely identify a person. For example, 87% of Americans are uniquely identifiable by their ZIP code, date of birth and gender alone [4]. One study [5] linked anoymised Netflix records to IMDb reviews to re-identify users and reveal sensitive information.

Differential privacy, first formally introduced in [6] 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 relating to the aggregated data of a database will still preserve 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 poten-

tially 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 enter their true answer or not. If not, they do another coin toss to determine whether they answer 'yes' or 'no' to the question. This results in a 25% chance that the respondent's true answer to the question is 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 having uncertainty about any particular individual's response.

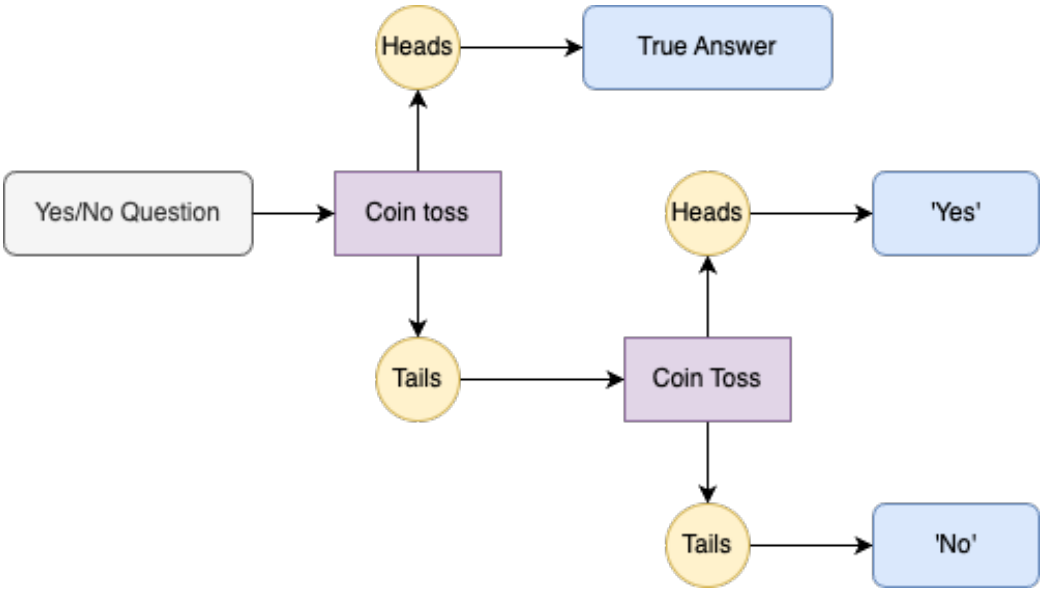

Figure 3.1: Algorithm for adding an answer to a database using the Randomised Response Mechanism

There are a number of variants of the mechanism, such as using randomisation devices other than a coin, with different probabilities of flipping an answer, or with data for more than two categories [8].

3.1.2 Formal Definitions

The original definition of differential privacy [6] relates to the comparing the likelihood of algorithms on neighbouring databases (i.e. differing by no more than one element) giving particular outputs.

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 someone has their data added to a dataset, any outputs of the algorithm are not too much more likely to be produced by one response or the other, or by if the data was not added in the first place. (diagram to explain?) The privacy parameter ϵ represents the strength of the guarantee. The Randomised Response Mechanism is log(3)-differentially private. Assuming the mechanism gives the answer 'yes' for a respondent, comparing the probabilities of the respondent's true answer:

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

Hence, the smallest ϵ for which the Randomised Response Mechanism is ϵ-differentially private is log(3).

The closer epsilon is to zero, the more likely the algorithm on similar databases will generate similar output. However, we are still interested in getting meaningful information from the data, so an algorithm with an ϵ of very close to zero will say almost nothing at all about what input caused it. For example, if in the Random Response Mechanism we instead gave a 50% chance of flipping a user's answer, all inputs would produce any output with the same probability, meaning the algorithm would be 0-differentially private. However, the average or sum of all responses would give no meaningful information about the data that created it. Hence, ϵ should be small (around 1 or smaller depending on the dataset and what we wish to gain from it [9]), but not too close to 0.

Note that when we talk about discrete output sets, instead of the definition needing to hold for all subsets $S \subseteq$ Range(M), we can change the definition so to refer to individual outputs, i.e. comparing $\Pr[\mathcal{M}(D) = S]$ and $\Pr[\mathcal{M}(D') = S]$.

It may be the case that an algorithm provides ϵ-differential privacy most of the time, but fails to in the case of outputs that are unlikely. In this case, 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.

This definition doesn't give much information on the behaviour of a mechanism when it fails. For instance, the following algorithm, a variation on the Catastrophe Mechanism [10], is $(\log(3), 0.1)$-differentially private:

With probability 0.9: Run the Randomised Response Mechanism.
Otherwise: Leak the whole database.

This would clearly 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 much less catastrophically than the Catastrophe mechanism.

3.1.3 The Laplace and Gaussian Mechanisms

Note that in the Randomised Response Mechanism, we alter each data entry individually. Here we introduce two mechanisms which only alter a query about the database as a whole. This includes counting queries, where where we return the number of elements in a database that satisfy a certain property [11], with no other access to the data.

The Laplace Mechanism [12] and Gaussian Mechanism [13] are very similar. Differential privacy is guaranteed by adding random noise, sampled from either the Laplace distribution or the Gaussian 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 $\sigma^2 = \frac{\log(1.25/\delta)}{\epsilon^2}$.

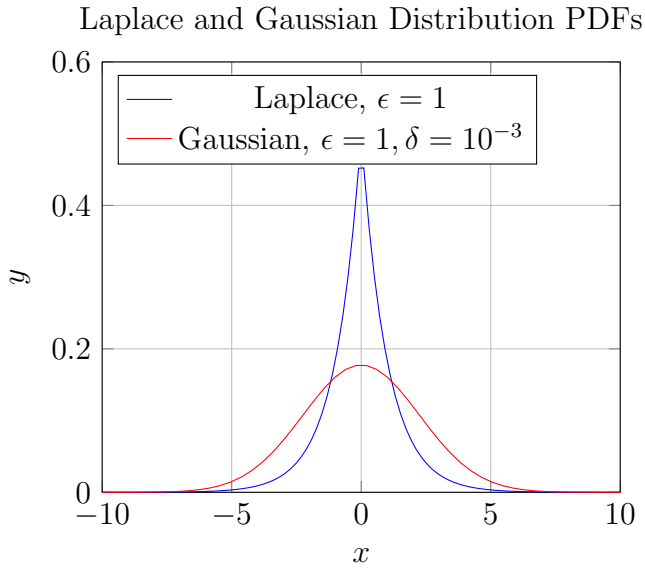

Figure 3.2: Comparison of PDFs of the Laplace and Gaussian distributions

-Figures of comparing counts for neighbouring databases, Explanation of why the Gaussian mechanism must have some delta-

In practice it is ideal to use the discrete versions of these distributions, which give integer output. These make up what are known as the Discrete Laplace Mechanism [14] and the Discrete Gaussian Mechanism [15]. These discretisations are preferred due to security issues related to using floating point arithmetic, as shown in [16], as well as for interpretability (it makes little sense to say 5424.834... respondents answered 'yes' to a survey question).

We can observe in Figure 3.2 that the Gaussian distribution has a wider spread than the Laplace distribution with same ϵ, meaning the output is expected to less accurate for a worse privacy guarantee. Considering these facts, one may ask why one would consider the Gaussian mechanism when

a direct upgrade exists in the Laplace mechanism. The advantage of using the Gaussian comes from using a different measure of sensitivity. Generally speaking, sensitivity is the amount a function will change when its input changes. The difference means that more complex multi-valued queries, much less noise may need to be added compared to the Laplace distribution for a similar privacy guarantee. This is beyond the discussion of this report, as all examples are for single-valued queries with sensitivity=1, since the count of entries in a database satisfying a property can change by no more than 1 between neighbouring databases. It is however worth noting that both of these mechanisms have use in practice.

3.2 Quantitative Information Flow Background

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 outputting statistics from data, necessarily involve leaking some information about the data that created that output. Other aspect of programs, such as timing information and power consumption can also unintentionally leak information about the secret. For this reason, rather that simply labelling programs as secure or insecure by identifying whether or not programs leak information, the theory of Quantitative Information Flow (QIF) is used to provide a rigorous framework for quantifying information leakage in programs. QIF provides numerical guarantees for the amount of information flow and the extent to which it can be tolerated, given the presence of an adversary that has the goal of exploiting leaked information [17].

3.2.2 Formal Definitions

The following formal definitions of the different aspects of Quantitative Information Flow have been detailed in [17]:

Secrets are values that are of interest to an adversary, such as a person's data, or a password, over which there is 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 $x : \mathcal{X}$.

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 $C : \mathcal{X} \times \mathcal{Y} \to [0,1]$. C_{xy} is the probability that y is observed given x is the secret. Given prior distributions on secrets, we can also calculate joint distributions of secrets and observables, marginal distributions of observables and posterior distributions of secrets. Combining marginal and posterior distributions, we get the idea of hyper-distributions (distributions of distributions) of type $\mathbb{D}^2\mathcal{X}$, a weighted $\oplus$-sum of posteriors (where weights are marginal probabilities).

Adversaries are in the form of loss functions $\ell : \mathcal{W}, \mathcal{X} \to \mathbb{R}$ (or often more intuitively formalised as its dual, the gain function [18]), which model the uncertainty surrounding a secret, depending on a choice of action ($w : \mathcal{W}$) and the secret's exact value (x). The choice of loss function can depend on what exactly the goal and capabilities of the adversary are. A popular loss function used in foundational studies on quantitative information flow is Shannon entropy [19], however it is known to not generalise well to many specific goals an adversary may have [20]. We can formally define the uncertainty $U_l[\pi]$ of the secret π with respect to ℓ as

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

3.2.3 Differential Privacy in QIF

Previous work [1] has been conducted in relating 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. Differential privacy mechanisms can be modelled as channels, where data entries are secrets, and the observables are the outputs of the differential privacy mechanisms.

Definitions are given in [2] for modelling ϵ-differential privacy using channel matrices:

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

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

Note, if this holds for a mechanism M for all possible pairs x, x' that correspond to neighbouring databases, then mechanism M satisfies ϵ-differential privacy.

[2] also define channel mechanism differential privacy in terms of loss functions...

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`), that is, what the adversary can observe. The entries of the matrix give the probabilities of the observation given the secret, for example, $\Pr[\texttt{result=0}|\texttt{resp=0}] = 3/4$. Using Definition 3, we can validate that the RRM is $\log(3)$-differentially private, and is that $\log(3)$ is the smallest ϵ for which the inequalities hold.

3.2.4 Kuifje

Kuifje [3] is a programming language designed for Quantitative Information Flow. When a program is run in Kuifje, whenever a probabilistic step is reached, such as tossing a coin in the Randomised Response Mechanism, 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 branch. That is, Kuifje outputs the hyper-distributions of each variable in a program, with the outer probability distribution of what the attacker can observe, and the inner probability distributions of the values of the variable (or secret) given

the attacker's observation.

– Give easy example - RRM –

4. Methods

4.1 Extension of Differential Privacy definitions in QIF

In order to allow for QIF analysis on any mechanism rather than only those that satisfy ϵ-differential privacy, I extend Definition 3 to the following for (ϵ, δ)-differential privacy:

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

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 $\Pr[M(D) \in S]$.

Aside from using channel matrices for determining differential privacy guarantees, [2] also show how to use the QIF concept of loss functions to do similarly, which can provide many useful calculations for determining the unpredictability of secrets in the context of differential privacy. The following is the extension of the definition of differential privacy using loss functions:

–Lots to define e.g. U, $\mathcal{X}$ and not part of main method so cut out?...–

Definition 4 *Given are $\epsilon > 0$, $\delta \geq 0$ and $\mathcal{W} := \mathcal{V} \cup \{\star\}$, where $\mathcal{V} \subseteq \mathcal{X} \times \mathcal{X}$ is symmetric and irreflexive. dp_ϵ, the ϵ-differentially private loss function*

relative to $\mathcal{W}$ is defined:

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

Theorem 1 in [2] gives the condition for a mechanism being ϵ-differentially private using this loss function. The following corollary extends this theorem for (ϵ, δ)-differential privacy:

Corollary 1 *Where v is the uniform distribution in $\mathbb{D}\mathcal{X}$, mechanism M is (ϵ, δ)-differentially private if and only if*

$$U_{dp_\epsilon}[v\rangle M] \geq -\delta$$

4.2 Programming Differential Privacy in Kuifje

4.2.1 Mechanism demonstration

As shown in Section 3.2.4, we can code a differential privacy mechanism in the Kuifje language. The general procedure used to demonstrate the workings of a mechanism in a counting query is as follows:

1. Take a fixed toy database containing 1's and 0's. 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.

The hyper-distribution output of the unknown response will be used for analysing the differential privacy guarantees. – Add diagram? –

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 in [15] for exact sampling from the discrete Laplace and Gaussian distributions. A number of steps were taken to simplify the algorithm, some being necessary, due to the range of the distributions being infinite and there being an infinite number of possible computation branches, meaning the program would fail to terminate in Kuifje. As a result the range of the mechanism was limited, and the tails of the distributions were squished, which will give slightly worsened differential privacy results.

This is particularly an issue in the case of the Discrete Laplace Mechanism, which is supposed to be ϵ-differentially private. The limit on the range of the distribution 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 gives complete certainty of 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 only satisfy (ϵ, δ)-differential privacy. It would be ideal, as future work, to add these distributions to Kuifje in a way that doesn't squish the distribution in such a crude fashion. However in this case, the imperfections of the distributions help in calculating differential privacy results when one may be unsure.

4.3 Demonstrating Differential Privacy Results

The output Kuifje provides is in the form of a hyper-distributions, containing the probabilities $\Pr[\text{observation} = y]$ and $\Pr[\text{secret} = x | \text{observation} = y]$.

In the case of our unknown response, which we call `resp`, and what the adversary observes the `noisyCount`, the Kuifje output gives $\Pr[\texttt{noisyCount} = y]$ and $\Pr[\texttt{resp} = x | \texttt{noisyCount} = y]$. Using Bayes' theorem, we can calculate the corresponding channel matrix value for x and y:

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

Using this equation, we can work out all values for 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 needs to have knowledge of this prior distribution, from when 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 *epsilon*-differentially privacy (in the case that no ratios are infinite).

One thing that would be of interest is for a channel matrix, fixing ϵ and find the minimum δ for which the mechanism satisfies (ϵ, δ)-differential privacy. 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.

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

Using the method described in 4.2.1 for programming a mechanism in Kuifje, and 4.3 for demonstrating differential privacy results, the channel matrix extracted from the output, displayed in A.1.2 for the Randomised Response Mechanism is the same as in 3.2.3. $\epsilon = \log(3)$ is the smallest *epsilon* for which this channel is ϵ-differentially private, which matches the theoretical result. Hence, the proposed method works for simple ϵ-differential privacy calculations.

5.2 Discrete Laplace and Gaussian

We program both the Discrete Laplace and Gaussian Mechanisms in Kuifje using the same method, with sampling from the distributions as described in 4.2.2, with full code shown in A.2.1. The resulting calculations in A.2.2 show that for the Discrete Laplace Mechanism with parameters that would theoretically provide $\frac{1}{3}$-differential privacy, we instead get 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. While these results are weaker than what the mechanism would give in theory, they are still expected and can be attributed to the squeezing of the distributions in our sampling method, as explained in 4.2.2. Therefore, these calculations demonstrate that the method used correctly calculates the (ϵ, δ)-differential privacy guarantees for the programmed mechanisms.

6. Conclusion

6.1 Summary

– Yet to write –

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.

There are a number of other useful concepts in the study of differential privacy that it would be helpful to provide examples of [brifely expand upon each of these] multi-valued queries, composition schemes, online vs offline. It would also be beneficial to test examples to the scale that mechanisms are applied in practice. Increasing scale wouldn't affect privacy guarantees, but it would serve to provide discussion on the utility of mechanisms.

Another area of interest is using relaxations of differential privacy beyond (ϵ, δ)-differential privacy. Whilst (ϵ, δ)-differential privacy can be used to capture the properties of any mechanism, there are a couple of drawbacks. These include the fact that mechanisms such as the Gaussian Mechanism satisfy a curve of (ϵ, δ)-differential privacy definitions rather than a single (ϵ, δ) pair, as well as the lack of descriptiveness on the behaviour of the mechanism that causes the delta (i.e. 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). Hence other differential privacy relaxations have been proposed and studied, such as Rényi differential privacy [21] and concentrated differential privacy [22]. It would therefore be helpful to look into how differential privacy guarantees could be validate under these definitions, as we do with pure and approximate differential privacy.

Differential privacy is proving to have many useful applications, including to 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 [23]. Other interesting work has been done linking differential privacy to robustness to adversarial inputs in machine learning models [24]. 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 with small examples of machine learning models.

A. Appendix

The following code is available at [cite repo?] which includes further experiments, including with the Randomised Response mechanism with different answer-flipping probabilities and with categorical data.

A.1 Randomised Response Mechanism in Kuifje

A.1.1 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`:

```
> 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.1.2 Differential Privacy calculations

Using Bayes' formula calculations as described in 4.3, 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 3.2.3 can be shown to be log(3)-differentially private.

A.2 Discrete Laplace and Gaussian in Kuifje

A.2.1 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, $\texttt{t} = 1/\epsilon$. 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.2.2 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 $\frac{e^{\frac{1}{3}}}{1+e^{\frac{1}{3}}} \approx 0.582570$. 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 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 $\delta \approx 0.000007$.

Note 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.

We can also observe that the smallest δ for which we have (ϵ, δ)-differential privacy is 0.000006, and use a similar method to calculate the smallest ϵ for which we get $(\epsilon, 0.000006)$-differential privacy.

Discrete Gaussian Mechanism

Using the same method in A.2.2, 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 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] Mário S Alvim et al. "Differential privacy versus quantitative information flow". In: *arXiv preprint arXiv:1012.4250* (2010).

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

[3] Jeremy Gibbons et al. "Quantitative information flow with monads in haskell". In: *Foundations of Probabilistic Programming* (2019).

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

[5] 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.

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

[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] 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.

[9] 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.

[10] Joseph P Near and Chiké Abuah. *Programming differential privacy*. 2021.

[11] 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.

[12] Cynthia Dwork et al. "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.

[13] Cynthia Dwork et al. "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.

[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 et al. *The Science of Quantitative Information Flow*. Springer, 2020.

[18] Mário S Alvim et al. "Measuring information leakage using generalized gain functions". In: *2012 IEEE 25th Computer Security Foundations Symposium*. IEEE. 2012, pp. 265–279.

[19] Claude Elwood Shannon. "A mathematical theory of communication". In: *The Bell system technical journal* 27.3 (1948), pp. 379–423.

[20] 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.

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

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

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

[24] Mathias Lecuyer et al. "Certified robustness to adversarial examples with differential privacy". In: *2019 IEEE symposium on security and privacy (SP)*. IEEE. 2019, pp. 656–672.