Journal of Privacy and Confidentiality (2013) **5**, Number 2, 57–109

Towards a Systematic Analysis of Privacy Definitions

Bing-Rong Lin* and Daniel Kifer†

1 Introduction

Data collection and analysis help drive innovation in business and science. In many cases, it is beneficial to share the data (e.g., to crowd-source the analysis [34], to enable collaborations, etc.). The data often contain sensitive information and so the straightforward sharing of data is not possible. In such cases, the original data should be passed to a *sanitizing* algorithm which processes the input data and outputs *sanitized* data. If the sanitizing algorithm is well-designed, it should be safe to release this sanitized data.

The design of a sanitizing algorithm is governed by a *privacy definition*. Mathematically, a privacy definition is simply a set of sanitizing algorithms. This set is often expressed as constraints on the behavior of an algorithm [12, 40]. Conceptually, a privacy definition acts like a mathematical contract—if the behavior of the algorithm satisfies its prespecified constraints, then certain types of sensitive inference are blocked. As is the case with legal contracts, privacy definitions are often subtle and their implications can be difficult to understand. In fact, highly publicized privacy breaches (e.g., [61, 52, 4]) have resulted from fundamental misunderstandings about what can be guaranteed by a particular class of sanitizing algorithms.

Most analyses of privacy definitions are highly targeted (e.g., [38, 45, 48, 61, 47, 66, 55, 28, 27, 53, 3, 50]) and examine whether a specific attack against a specific privacy definition or sanitizing algorithm can reveal specific types of information. Such attacks are great at raising awareness of privacy issues and types of flaws to avoid in the design of privacy definitions. However, it is often easy to modify a data sanitizer to defend against a specific attack algorithm (i.e., attacks need to be customized to a data sanitizer) without significantly improving the privacy protections of the sanitizer. Also, the failure of a specific attack does not necessarily guarantee that a sanitizing algorithm is safe.

For this reason, there is also interest in more systematic analyses with generalizable results. This include fundamental limits on accuracy of data generated by sanitizing algorithms [24, 18, 25, 36, 22, 13] and analyses of privacy with respect to a wide variety of Bayesian attackers as well as non-Bayesian attackers with different kinds of background knowledge [23, 41, 42, 8, 37, 54, 7, 26, 5, 29, 51].

Of particular interest are Bayesian approaches such as [37, 54, 26, 8, 42, 51] that

*Department of Computer Science & Engineering, Pennsylvania State University, mailto:blin@cse.psu.edu.

†Department of Computer Science & Engineering, Pennsylvania State University, mailto:dkifer@cse.psu.edu.

consider the safety of different types of sensitive information with respect to a wide variety of attackers. In this paper, we consider the inverse of this problem: given a privacy definition, who are the attackers and what types of information are being protected from them? An answer to this question can identify privacy definitions that are too strong (i.e., they may unnecessarily protect information that is not sensitive).[1]

In this paper, we introduce a novel methodology for this inverse problem which reduces it to a series of well-defined mathematical sub-problems. Given a privacy definition $\mathfrak{Priv}$, the first sub-problem is to derive its consistent normal form CNF($\mathfrak{Priv}$), which is defined with the help of two privacy axioms from [39, 40] (the definition can easily be extended to accommodate new axioms). Intuitively, CNF($\mathfrak{Priv}$) corresponds to a complete set of trusted sanitizing algorithms. The second mathematical sub-problem is the construction of the row cone of $\mathfrak{Priv}$, denoted by rowcone($\mathfrak{Priv}$), from CNF($\mathfrak{Priv}$). Mathematically, rowcone($\mathfrak{Priv}$) is just a convex set in a vector space. However, the supporting hyperplanes of this convex set can be re-interpreted as statements about prior and posterior distributions, and, as we discuss later, the row cone itself corresponds to the possible Bayesian inferences an attacker can make when data is sanitized using one of the trusted algorithms.

As a proof of concept, we apply the methodology to derive previously unknown privacy semantics for randomized response [62], FRAPP [1]/PRAM [32], and several algorithms that add integer-valued noise to their inputs. We show that their Bayesian privacy guarantees follow from the protection of various notions of parity of a dataset.

Along the way, we discuss other benefits of this methodology, such as providing guidelines for the design of privacy definitions (which influenced [42]) and new methods for relaxing privacy definitions. In particular, we show how Fourier-Motzkin elimination—a tool for working with linear inequalities—can be used to relax randomized response.

The remainder of the paper is organized as follows. We provide a detailed overview of our approach in Section 2. We discuss related work in Section 3. In Section 4, we review two privacy axioms from [39, 40] and then we show how to use them to obtain the consistent normal form. Using the consistent normal form, we formally define the row cone, a fundamental geometric object we use for extracting semantic guarantees, in Section 4.2. In Section 5, we then apply our framework to extract new semantic guarantees for randomized response (Section 5.1), FRAPP/PRAM (Section 5.2), and noise addition algorithms (Section 5.3). We discuss relaxations of privacy definitions in Section 5.4 and present conclusions in Section 6.

2 The Bird's-Eye View

We first present some basic concepts in Section 2.1 and then provide a high-level overview of our framework in Section 2.2.

[1] Of course, as with direct Bayesian approaches [37, 54, 26, 8, 42, 51], an answer to this problem can also identify privacy definitions that are too weak (i.e., those that do not protect the right information or do not protect against the right attackers).

2.1 Basic Concepts

Let $\mathbb{I} = \{D_1, D_2, \dots\}$ be the set of all possible databases. We now explain the roles played by data curators, attackers, and privacy definitions.

The Data Curator owns a dataset $D \in \mathbb{I}$. This dataset contains information about individuals, business secrets, etc., and therefore cannot be published as is. Thus the data curator will first choose a privacy definition and then an algorithm $\mathfrak{M}$ that satisfies this definition. The data curator will apply $\mathfrak{M}$ to the data D and will then release its output (i.e., $\mathfrak{M}(D)$), which we refer to as the *sanitized output*. We assume that the schema of D is public knowledge and that the data curator will disclose the privacy definition, release all details of the algorithm $\mathfrak{M}$ (except for the specific values of the random bits it used), and release the sanitized output $\mathfrak{M}(D)$.

The Attacker will use the information about the schema of D, the sanitized output $\mathfrak{M}(D)$, and knowledge of the algorithm $\mathfrak{M}$ to make inferences about the sensitive information contained in D. In our model, the attacker is computationally unbounded. However, we assume that when an attacker sees a sanitized output ω, the attacker's inference will depend on the likelihood vector $[P(\mathfrak{M}(D_1) = \omega), P(\mathfrak{M}(D_2) = \omega), \dots]$ of ω. More specifically, whenever ω_1 and ω_2 have likelihood vectors that are proportional to each other, the attacker would make the same inference upon observing ω_1 or ω_2. Note that this assumption is true for the vast majority of statistical procedures: Bayesian inference, maximum likelihood inference, likelihood ratio tests, etc. [10]. Our methodology will bundle up all of the likelihood vectors that an attacker could see as a result of using some privacy definition and will provide Bayesian restrictions on what *cannot* be inferred from those likelihood vectors.

A Privacy Definition is often expressed as a set of algorithms that we trust (e.g., [62, 40]), or a set of constraints on how an algorithm behaves (e.g., [20]), or on the type of output it produces (e.g., [57]). For our purposes, it is more convenient to treat privacy definitions as sets of algorithms, but no generality is lost—if a set of constraints is specified, a privacy definition becomes the set of algorithms that satisfy those constraints; if outputs in a certain form (such as k-anonymous tables [57]) are required, a privacy definition becomes the set of algorithms that produce those types of outputs, etc. More formally, a privacy definition is the set of algorithms *with the same input domain* that are trusted to produce nonsensitive outputs from sensitive inputs. We therefore use the notation $\mathfrak{Priv}$ to refer to a privacy definition and $\mathfrak{M} \in \mathfrak{Priv}$ to mean that the algorithm $\mathfrak{M}$ satisfies the privacy definition $\mathfrak{Priv}$.

The data curator will choose a privacy definition based on what it can guarantee about the privacy of sensitive information. If a privacy definition offers too little protection (relative to the application at hand), the data curator will avoid it because sensitive information may end up being disclosed, thereby causing harm to the data curator. On the other hand, if a privacy definition offers too much protection, the resulting sanitized data may not be useful for statistical analysis. Thus it is important for the data curator to know exactly what a privacy definition guarantees.

The Goal is to determine what guarantees a privacy definition provides. We are

most interested in the guarantees on attacker inferences that always hold regardless of what sanitized output is produced by an algorithm satisfying that privacy definition. We focus on computationally unbounded Bayesian attackers and look for bounds on how much their beliefs change after seeing sanitized data. It is important to note that the guarantees will depend on assumptions about the attacker's prior distribution. This is necessary, since it is well-known that without any assumptions, it is impossible to preserve privacy while providing useful sanitized data [23, 41, 42].

2.2 Overview

In a nutshell, our approach is to represent deterministic and randomized algorithms as matrices (with possibly infinitely many rows and columns) and to represent privacy definitions as sets of algorithms and hence as sets of matrices. If our goal is to analyze only a single algorithm, we simply treat it as a privacy definition (set) containing just one algorithm. The steps of our framework then require us to normalize the privacy definitions to remove some implicit assumptions (we call the result the *consistent normal form*), extract the set of all rows that appear in the resulting matrices (we call this the *row cone*), find linear inequalities describing those rows, reinterpret the coefficients of the linear inequalities as probabilities, and reinterpret the inequalities themselves as statements about probabilities to get semantic guarantees. In this section, we describe these steps in more detail and defer a technical exposition of the consistent normal form and row cone to Section 4.

2.2.1 Algorithms as Matrices

Since our approach relies heavily on linear algebra, it is convenient to represent algorithms as matrices. *Every* algorithm $\mathfrak{M}$, randomized or deterministic, that runs on a digital computer can be viewed as a matrix in the following way. An algorithm has an input domain $\mathbb{I} = \{D_1, D_2, \dots\}$ consisting of datasets D_i and a range $\{\omega_1, \omega_2, \dots\}$. The input domain $\mathbb{I}$ and range($\mathfrak{M}$) are necessarily countable because each $D_i \in \mathbb{I}$ and $\omega_j \in \text{range}(\mathfrak{M})$ must be encoded as finite bit strings. The probability $P(\mathfrak{M}(D_i) = \omega_j)$ is well defined for both randomized and deterministic algorithms. The *matrix representation* of an algorithm is defined as follows (see also Figure 2.2.1).

Definition 2.1 (Matrix representation of $\mathfrak{M}$). *Let $\mathfrak{M}$ be a deterministic or randomized algorithm with domain $\mathbb{I} = \{D_1, D_2, \dots\}$ and range $\{\omega_1, \omega_2, \dots\}$. The matrix representation of $\mathfrak{M}$ is a (potentially infinite) matrix whose columns are indexed by $\mathbb{I}$ and rows are indexed by* range($\mathfrak{M}$). *The value of each entry (i, j) is the quantity $P(\mathfrak{M}(D_j) = \omega_i)$.*

2.2.2 Consistent Normal Form of Privacy Definitions

Recall from Section 2.1 that we take the unifying view that a privacy definition is a set of algorithms (i.e., the set of algorithms that satisfy certain constraints or produce

$$\begin{array}{c} \\ \omega_1 \\ \omega_2 \\ \omega_3 \\ \vdots \end{array} \begin{array}{c} \begin{array}{ccc} D_1 & D_2 & \dots \end{array} \\ \begin{pmatrix} P(\mathfrak{M}(D_1) = \omega_1) & P(\mathfrak{M}(D_2) = \omega_1) & \dots \\ P(\mathfrak{M}(D_1) = \omega_2) & P(\mathfrak{M}(D_2) = \omega_2) & \dots \\ P(\mathfrak{M}(D_1) = \omega_3) & P(\mathfrak{M}(D_2) = \omega_3) & \dots \\ \vdots & \vdots & \vdots \end{pmatrix} \end{array}$$

Figure 1: The matrix representation of $\mathfrak{M}$. Columns are indexed by datasets $\in$ domain($\mathfrak{M}$) and rows are indexed by outputs $\in$ range($\mathfrak{M}$).

certain types of outputs).

Not surprisingly, there are many sets of algorithms that do not meet common expectations of what a privacy definition is [40]. For example, suppose that we decide to trust an algorithm $\mathfrak{M}$ to generate sanitized outputs from the sensitive input data D. Suppose we know that a researcher wants to run algorithm $\mathcal{A}$ on the sanitized data to build a histogram. If we are willing to release the sanitized output $\mathfrak{M}(D)$ publicly, then we should also be willing to release $\mathcal{A}(\mathfrak{M}(D))$. That is, if we trust $\mathfrak{M}$ then we should also trust $\mathcal{A} \circ \mathfrak{M}$ (the composition of the two algorithms). In other words, if $\mathfrak{M} \in \mathfrak{Priv}$, for some privacy definition $\mathfrak{Priv}$, then $\mathcal{A} \circ \mathfrak{M}$ should also be in $\mathfrak{Priv}$.

Many privacy definitions in the literature do not meet criteria such as this [40]. That is, $\mathfrak{M}$ may explicitly satisfy a given privacy definition but $\mathcal{A} \circ \mathfrak{M}$ may not. However, since the output of $\mathfrak{M}$ is made public and anyone can run $\mathcal{A}$ on it, these privacy definitions come with the implicit assumption that the composite algorithm $\mathcal{A} \circ \mathfrak{M}$ should be trusted.

Thus, given a privacy definition $\mathfrak{Priv}$, we first must expand it to include all of the algorithms we should trust. An analyst would do this by first choosing a set of axioms that have the following template: "if one trusts algorithms $\mathfrak{M}_1, \mathfrak{M}_2, \dots$ then one should trust algorithm $\mathfrak{M}_*$." The analyst would repeatedly apply these rules to sanitizing algorithms that belong to $\mathfrak{Priv}$ to obtain CNF($\mathfrak{Priv}$), the consistent normal form of $\mathfrak{Priv}$. Thus the CNF($\mathfrak{Priv}$) is the complete set of algorithms the analyst should trust if the analyst decides to trust $\mathfrak{Priv}$. In this paper, we use two privacy axioms proposed by [39, 40] to construct the consistent normal form. We present the full technical detail in Section 4.1.

2.2.3 The Row Cone

Recall that we represent algorithms as matrices (Definition 2.1) and privacy definitions as sets of algorithms. Therefore CNF($\mathfrak{Priv}$) is really a *set of matrices*. The row cone of $\mathfrak{Priv}$, denoted by rowcone($\mathfrak{Priv}$), is the set of vectors of the form $c\vec{x}$ where $c \geq 0$ and $\vec{x}$ is a row of a matrix corresponding to some algorithm $\mathfrak{M} \in$ CNF($\mathfrak{Priv}$).

How does the row cone capture the semantics of $\mathfrak{Priv}$? Suppose $\mathfrak{M} \in$ CNF($\mathfrak{Priv}$)

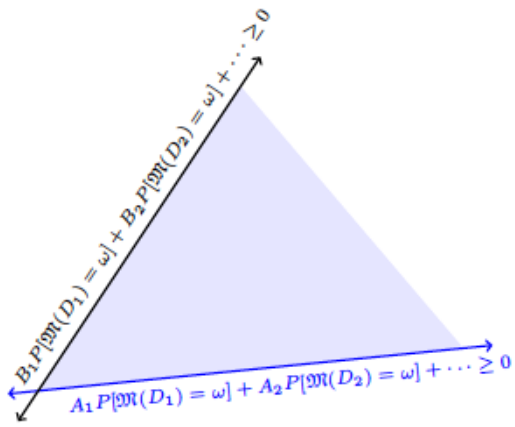

Figure 2: An example of a row cone (shaded) and its defining linear inequalities.

is one of the algorithms that we trust. Let D be the true input dataset and let $\omega = \mathfrak{M}(D)$ be the sanitized output that we publish. A Bayesian attacker who sees output ω and is trying to derive sensitive information will try to base decisions on the posterior distribution $P(\text{data} = D_i \mid \mathfrak{M}(\text{data}) = \omega)$ for all datasets D_i. This posterior distribution is a function of the attacker's prior $P(\text{data} = D_i)$ and the likelihood vector:

$$[P(\mathfrak{M}(D_1) = \omega), \quad P(\mathfrak{M}(D_2) = \omega), \quad \ldots]$$

This vector belongs to rowcone($\mathfrak{Priv}$) because it corresponds to some row of the matrix representation of $\mathfrak{M}$ (i.e., the row associated with output ω). Note that multiplying this likelihood vector by any positive constant will leave the attacker's posterior beliefs unchanged. The row cone is essentially the set of all such probability vectors that the attacker can ever see if we use a trusted algorithm (i.e., something belonging to CNF($\mathfrak{Priv}$)); therefore it determines all the ways an attacker's beliefs can change (from prior to posterior).

We can study the row cone by analyzing the properties that are common to all of the likelihood vectors that comprise the row cone. This is where the geometry of the row cone is important. In Figure 2.2.3 we illustrate a row cone in 2 dimensions (i.e., the input domain consists of only two datasets). Each vector in the row cone is represented as a point in 2-d space. As we show in Section 4.2, the row cone is actually a convex set and hence has an associated system of linear inequalities (corresponding to the intersection of halfspaces containing the row cone) as shown in Figure 2.2.3. The key insight of our work is that these linear constraints can be interpreted as statements about prior and posterior distributions; these statements are restrictions on an attacker's inference that must hold for every algorithm $\mathfrak{M} \in$ CNF($\mathfrak{Priv}$) and any sanitized output $\omega \in$ range($\mathfrak{M}$). We describe this interpretation of the linear inequalities next.

2.2.4 Extracting Semantic Guarantees From the Row Cone

The row cone is a convex set (in fact, a convex cone) and so satisfies a set of linear inequalities having the forms [9]:

$$\begin{aligned} A_1 P(\mathfrak{M}(D_1) = \omega) + A_2 P(\mathfrak{M}(D_2) = \omega) + \ldots &\geq 0 \text{ or} \\ A_1 P(\mathfrak{M}(D_1) = \omega) + A_2 P(\mathfrak{M}(D_2) = \omega) + \ldots &= 0 \text{ or} \\ A_1 P(\mathfrak{M}(D_1) = \omega) + A_2 P(\mathfrak{M}(D_2) = \omega) + \ldots &> 0 \end{aligned}$$

that must hold for all trusted algorithms $\mathfrak{M} \in \text{CNF}(\mathfrak{Priv})$ and sanitized outputs $\omega \in \text{range}(\mathfrak{M})$ they can produce.

The key insight is that we can re-interpret the magnitude of a coefficient $|A_i|$ as a prior probability $P(\text{data} = D_i)$ (dividing by $|A_1| + |A_2| + ...$ if necessary). Thus each $|A_i|$ is associated with a dataset $D_i \in \mathbb{I}$. Let S_+ be the set of D_i for which the corresponding A_i are positive and let S_- be the set of D_i for which the corresponding A_i are negative. A linear inequality can then be rewritten as a probabilistic statement such as $P(\text{data} \in S_+ \ \wedge \ \mathfrak{M}(\text{data}) = \omega) \geq P(\text{data} \in S_- \ \wedge \ \mathfrak{M}(\text{data}) = \omega)$, where data is a random variable (from the point of view of an attacker) corresponding to the true input database. Further manipulations, as described in Section 4.2.3, can result in statements with clearer semantic meanings.

We give a detailed example of such a re-interpretation in Section 5.1, where we apply our methodology to randomized response. The semantic guarantees we extract then have the form: "if the attacker's prior belongs to set X then here are restrictions on the posterior probabilities the attacker can form" (note that avoiding any assumptions on prior probabilities/knowledge is not possible if the goal is to release even marginally useful sanitized data [23, 41, 42]).

3 Related Work

3.1 Evaluating Privacy

Although most work in statistical privacy focuses on the design of sanitizing algorithms, there is active research on understanding the technical nature of privacy.

Dwork and Naor [23] formally proved that it is not possible to publish anonymized data that prevents an attacker from learning information about people who are not even part of the data unless the anonymized data has very little utility or some assumptions are made about the attacker's background knowledge. Lambert [43] suggests that harm can occur even when incorrect inference is made about and individual. For example, if sanitized data consists of anonymized records, harm can occur when an individual is linked to the wrong anonymized record (as long as the attacker's methods are plausible). Thus one of the biggest themes in privacy is hampering an attacker's ability to perform record linkage [17] even if external data [61] or other knowledge [48] is available.

The official statistics community routinely conducts re-identification experiments to

assess whether individuals can be identified from sanitized data records [12]. In many such experiments, software is used to link sanitized data records to the original records [65]. Reiter [55] provides a detailed example of how to apply the decision-theoretic framework of Duncan and Lambert [19] to measure disclosure risk. There are many other methods for assessing privacy for the purposes of official statistics; for surveys, see [12, 63, 64].

There are many attacks on noise-addition algorithms [35, 33, 47]. In particular, Dinur and Nissim [18] and subsequent work [22, 25, 13, 24, 36] showed fundamental limits to the amount of information that can be released even under very weak privacy definitions. Ganta et al. [28] demonstrated a composition attack where independent anonymized data releases can be combined to breach privacy; thus a desirable property of privacy definitions is to have privacy guarantees degrade gracefully in the presence of multiple independent releases of sanitized data. The minimality attack [66] showed that privacy definitions must account for attackers who know the algorithm used to generate sanitized data; otherwise the attackers may reverse-engineer the algorithm to cause a privacy breach. The de Finetti attack [38] shows that privacy definitions based on statistical models are susceptible to attackers who make inferences using different models and use those inferences to undo the anonymization process; thus it is important to consider a wide range of inference attacks. Also, one should consider the possibility that an attacker may be able to manipulate data (e.g., by creating many new accounts in a social network) prior to its release to help break the subsequent anonymization of the data [3]. Note also that privacy concerns can also be associated with aggregate information such as trade secrets (and not just rows in a table) [14, 42].

Work that considers the privacy of multiple pieces of sensitive information against attackers with a wide variety of prior beliefs or background knowledge (e.g., [23, 41, 42, 8, 37, 54, 7, 26, 5, 29, 51]) is of particular relevance to this paper. The methodology we present here is complementary to such work and seeks to answer the inverse question, that of identifying the specific attackers and the specific pieces of information that are hidden from them by a privacy definition. An initial discussion of our approach appeared in a report describing our invited talk [46].

3.2 Privacy Definitions

In this section, we review some privacy definitions that will be examined in this paper.

3.2.1 Syntactic Privacy Definitions

A large class of privacy definitions place restrictions on the format of the output of a randomized algorithm. Such privacy definitions are known as *syntactic privacy definitions*. The prototypical syntactic privacy definition is k-anonymity [57, 61]. In the k-anonymity model, a data curator first designates a set of attributes to be the *quasi-identifier*. An algorithm $\mathfrak{M}$ then satisfies k-anonymity if its input is a table T and its output is another table T^* that is *k-anonymous*—for every tuple in T^*, there are $k-1$ other tuples that

have the same value for the quasi-identifier attributes [57, 61]. Algorithms satisfying k-anonymity typically work by generalizing (coarsening) attribute values. For example, if the data contains an attribute representing the age of a patient, the algorithm could generalize this attribute into age ranges of size 10 (e.g., $[0-9], [10-19]$, etc.) or ranges of size 20, etc. Quasi-identifier attributes are repeatedly generalized until a table T^* satisfying k-anonymity is produced. The rationale behind k-anonymity is that quasi-identifier attributes may be recorded in publicly available datasets. Linking those datasets to the original table T may allow individual records to be identified, but linking to the k-anonymous table T^* will not result in unique matches.

3.2.2 Local Perturbation Methods

We will analyze the following local perturbation methods that independently modify records in the input datasets.

Randomized Response. Randomized response is a technique developed by Warner [62] to deal with privacy issues when answering sensitive questions in a face-to-face survey. There are many variations of randomized response. One of the most popular is the following: a respondent answers truthfully with probability p and lies with probability $(1-p)$, thus ensuring that the interviewer is not certain about the respondent's true answer. Thus the scenario where we can apply randomized response is the following: the input table T contains 1 binary attribute and k tuples. We can apply randomized response to T by applying the following procedure to each tuple: flip the binary attribute with probability $1-p$. The perturbed table, which we call T^*, is then released. Note that randomized response is a privacy definition that consists of exactly one algorithm: the algorithm that flips each bit independently with probability $1-p$. We use our framework to extract semantic guarantees for randomized response in Section 5.1.

PRAM and FRAPP. PRAM [32] and FRAPP [1] are generalizations of randomized response to tables where tuples can have more than one attribute and the attributes need not be binary. PRAM can be thought of as a set of algorithms that independently perturb tuples, while FRAPP is an extension of PRAM that adds formally specified privacy restrictions to these perturbations.

Let $\mathcal{TUP}$ be the domain of all tuples. Each algorithm $\mathfrak{M}_Q$ satisfying PRAM is associated with a transition matrix Q of transition probabilities, where the entry $Q_{b,a}$ is the probability $P(a \to b)$ that the algorithm changes a tuple with value $a \in \mathcal{TUP}$ to the value $b \in \mathcal{TUP}$. Given a dataset $D = \{t_1, \ldots, t_n\}$, the algorithm $\mathfrak{M}_Q$ assigns a new value to the tuple t_1 according to the transition probability matrix Q, then it independently assigns a new value to the tuple t_2, etc.[2] It is important to note that the matrix representation of $\mathfrak{M}_Q$ (as discussed in Section 2.2.1) *is not the same* as the transition matrix Q. As we will discuss in Section 5.2, the relationship between the two

[2]In principle, a different transition matrix $Q^{(i)}$ can be applied to each tuple t_i as long as the choice of transition matrix does not depend on sensitive information. We do not consider this extension here.

is that the matrix representation of $\mathfrak{M}_Q$ is equal to $\bigoplus_n Q$, where $\bigoplus$ is the Kronecker product.

FRAPP, with privacy parameter γ, imposes a restriction on these algorithms. This restriction, known as γ-amplification [26], requires that the transition matrices Q satisfy the constraints $\frac{Q_{b,a}}{Q_{c,a}} \leq \gamma$ for all $a, b, c \in \mathcal{TUP}$. This condition can also be phrased as $\frac{P(b \to a)}{P(c \to a)} \leq \gamma$.

3.2.3 Differential Privacy

Differential privacy [20, 21] is defined as follows:

Definition 3.1. *A randomized algorithm $\mathfrak{M}$ satisfies* ϵ-differential privacy *if for all pairs of databases T_1, T_2 that differ only in the value of one tuple and for all sets S, $P(\mathfrak{M}(T_1) \in S) \leq e^\epsilon P(\mathfrak{M}(T_2) \in S)$.*

Differential privacy guarantees that the sanitized data that is output has little dependence on the value of any individual's tuple (for small values of ϵ). It is known to be a weaker privacy definition than randomized response.[3] Using our framework, we show in Section 5.1.1 that the difference between the two is that randomized response provides additional protection for the parity of every subset of the data.

4 Consistent Normal Form and the Row Cone

In this section, we formally define the *consistent normal form* CNF($\mathfrak{Priv}$) and rowcone($\mathfrak{Priv}$) of a privacy definition $\mathfrak{Priv}$ and derive some of their important properties. These properties will later be used in Section 5 to extract novel semantic guarantees for randomized response, FRAPP/PRAM, and for several algorithms (including the geometric mechanism [30]) that add integer random noise to their inputs.

4.1 The Consistent Normal Form

Recall that we treat any privacy definition $\mathfrak{Priv}$ as the set of algorithms with the same input domain. For example, we view k-anonymity as the set of all algorithms that produce k-anonymous tables [57]. Such a set can often have inconsistencies. For example, consider an algorithm $\mathfrak{M}$ that first transforms its input into a k-anonymous table and then builds a statistical model from the result and outputs the parameters of that model. Technically, this algorithm $\mathfrak{M}$ does not satisfy k-anonymity because "model parameters" are not a "k-anonymous table." However, it would be strange if the data curator decided that releasing a k-anonymous table was acceptable but releasing a model built solely from that table (without any side information) was not acceptable. The motivation for

[3] Formally, randomized response with parameter p satisfies differential privacy with parameter $\epsilon = |\log \frac{p}{1-p}|$.

the consistent normal form is that it makes sense to enlarge the set $\mathfrak{Priv}$ by adding $\mathfrak{M}$ into this set.

It turns out that privacy axioms can help us identify the algorithms that should be added. For this purpose, we will use the following two axioms from [40].

Axiom 4.1 (Post-processing [40]). *Let $\mathfrak{Priv}$ be a privacy definition (set of algorithms). Let $\mathfrak{M} \in \mathfrak{Priv}$ and let $\mathcal{A}$ be any algorithm whose domain contains the range of $\mathfrak{M}$ and whose random bits are independent of the random bits of $\mathfrak{M}$. Then the composed algorithm $\mathcal{A} \circ \mathfrak{M}$ (which first runs $\mathfrak{M}$ and then runs $\mathcal{A}$ on the result) should also belong to $\mathfrak{Priv}$.*[4]

Note that Axiom 4.1 prevents algorithm $\mathcal{A}$ from using side information since its only input is $\mathfrak{M}(D)$. This axiom is useful when we consider computationally unbounded attackers. For computationally bounded attackers, a weaker axiom should be used since Axiom 4.1 often rules out privacy definitions that use encryption that is based on computational assumptions (see [40] for details).

Axiom 4.2 (Convexity [40]). *Let $\mathfrak{Priv}$ be a privacy definition (set of algorithms). Let $\mathfrak{M}_1 \in \mathfrak{Priv}$ and $\mathfrak{M}_2 \in \mathfrak{Priv}$ be two algorithms satisfying this privacy definition. Define the algorithm* $\text{choice}^{p}_{\mathfrak{M}_1,\mathfrak{M}_2}$ *to be the algorithm that runs $\mathfrak{M}_1$ with probability p and $\mathfrak{M}_2$ with probability $1-p$. Then* $\text{choice}^{p}_{\mathfrak{M}_1,\mathfrak{M}_2}$ *should belong to $\mathfrak{Priv}$.*

The justification in [40] for the convexity axiom (Axiom 4.2) is the following. If both $\mathfrak{M}_1$ and $\mathfrak{M}_2$ belong to $\mathfrak{Priv}$, then both are trusted to produce sanitized data from the input data. That is, the outputs of $\mathfrak{M}_1$ and $\mathfrak{M}_2$ leave some amount of uncertainty about the input data. If the data curator randomly chooses between $\mathfrak{M}_1$ and $\mathfrak{M}_2$, the sensitive input data is protected by two layers of uncertainty: the original uncertainty added by either $\mathfrak{M}_1$ or $\mathfrak{M}_2$ and the uncertainty about which algorithm was used. Further discussion can be found in [40].

Using these two axioms, we define the *consistent normal form* as follows:[5]

Definition 4.3. (CNF). *Given a privacy definition $\mathfrak{Priv}$, its consistent normal form, denoted by* $\text{CNF}(\mathfrak{Priv})$*, is the smallest set of algorithms that contains $\mathfrak{Priv}$ and satisfies Axioms 4.1 and 4.2.*

Essentially, the consistent normal form uses Axioms 4.1 and 4.2 to turn implicit assumptions about which algorithms we trust into explicit statements— if we are prepared to trust any $\mathfrak{M} \in \mathfrak{Priv}$ then by Axioms 4.1 and 4.2 we should also trust any $\mathfrak{M} \in \text{CNF}(\mathfrak{Priv})$. The set $\text{CNF}(\mathfrak{Priv})$ is also the largest set of algorithms we should trust if we are prepared to accept $\mathfrak{Priv}$ as a privacy definition.

[4]Note that if $\mathfrak{M}_1$ and $\mathfrak{M}_2$ are algorithms with the same range and domain such that $P(\mathfrak{M}_1(D_i) = \omega) = P(\mathfrak{M}_2(D_i) = \omega)$ for all $D_i \in \mathbb{I}$ and $\omega \in \text{range}(\mathfrak{M}_1)$, then we consider $\mathfrak{M}_1$ and $\mathfrak{M}_2$ to be equivalent.

[5]Note that this is a more general and useful idea than the observation in [40] that two specific variants of differential privacy do not satisfy the axioms but do imply a third variant that does satisfy the axioms.

The following theorem provides a useful characterization of CNF($\mathfrak{Priv}$) that will help us analyze privacy definitions in Section 5.

Theorem 4.4. *Given a privacy definition* $\mathfrak{Priv}$*, its consistent normal form* CNF($\mathfrak{Priv}$) *is equivalent to the following.*

1. *Define* $\mathfrak{Priv}^{(1)}$ *to be the set of all (deterministic and randomized algorithms) of the form* $\mathcal{A} \circ \mathfrak{M}$*, where* $\mathfrak{M} \in \mathfrak{Priv}$*,* range($\mathfrak{M}$) $\subseteq$ domain($\mathcal{A}$)*, and the random bits of* $\mathcal{A}$ *and* $\mathfrak{M}$ *are independent of each other.*

2. *For any positive integer* n*, finite sequence* $\mathfrak{M}_1, \ldots, \mathfrak{M}_n$ *and probability vector* $\vec{p} = (p_1, \ldots, p_n)$*, use the notation* choice$^{\vec{p}}(\mathfrak{M}_1, \ldots, \mathfrak{M}_n)$ *to represent the algorithm that runs* $\mathfrak{M}_i$ *with probability* p_i*. Define* $\mathfrak{Priv}^{(2)}$ *to be the set of all algorithms of the form* choice$^{\vec{p}}(\mathfrak{M}_1, \ldots, \mathfrak{M}_n)$ *where* n *is a positive integer,* $\mathfrak{M}_1, \ldots, \mathfrak{M}_n \in \mathfrak{Priv}^{(1)}$*, and* $\vec{p}$ *is a probability vector.*

3. *Set* CNF($\mathfrak{Priv}$) $= \mathfrak{Priv}^{(2)}$*.*

Proof. See Appendix 1. □

Corollary 4.5. *If* $\mathfrak{Priv} = \{\mathfrak{M}\}$ *consists of just one algorithm,* CNF($\mathfrak{Priv}$) *is the set of all algorithms of the form* $\mathcal{A} \circ \mathfrak{M}$*, where* range($\mathfrak{M}$) $\subseteq$ domain($\mathcal{A}$) *and the random bits in* $\mathcal{A}$ *and* $\mathfrak{M}$ *are independent of each other.*

Proof. See Appendix 2. □

4.1.1 Discussion and Generalizations

In this section we briefly outline some implications of our discussion about the consistent normal form.

- The consistent normal form of $\mathfrak{Priv}$ was computed using two privacy axioms and represents the complete set of algorithms an analyst should trust if the analyst is prepared to trust all algorithms in $\mathfrak{Priv}$ and accepts Axioms 4.1 and 4.2. If the analyst is willing to trust more algorithms, the analyst needs to specify additional axioms in order to obtain a consistent normal form.

- It should come as no surprise that computing the consistent normal form of an arbitrary privacy definition may not always be possible (or it may not always be possible to do it efficiently). However, since the consistent normal form corresponds to the complete set of algorithms we should trust, it seems reasonable to request that new privacy definitions should be designed in ways that allow characterizations of their consistent normal form to be derived (this bias towards privacy definitions with tractable consistent normal forms can affect utility and the tradeoff deserves more study). We give an example in the following note.

- Not all privacy axioms follow the template "if one trusts $\mathfrak{M}_1, \mathfrak{M}_2, \ldots$ then one should trust $\mathfrak{M}_*$" (although Axioms 4.1 and 4.2 do). Some axioms can tell us which sanitizing algorithms $\mathfrak{M}$ should not be trusted. One example is: "the algorithm that outputs its inputs should not be trusted." Such an axiom is not used to construct a consistent normal form, but it can be used to evaluate it; if CNF($\mathfrak{Priv}$) does not satisfy this axiom, then one shouldn't use $\mathfrak{Priv}$ to protect sensitive data. As an example, let us compute the consistent normal form for k-anonymity.[6]

  When the input table has a continuous attribute such as *Age*, it is possible to coarsen the age into ranges such as 0–15, 16–50, etc. However, a malicious algorithm could first generate a k-anonymous table and then refine the bucket boundaries in a way that leaks information while still preserving k-anonymity. For example, if *Age* is initially coarsened into the buckets mentioned previously, it is possible to refine their boundaries by changing the first bucket to 0–15.1010110... (where we append the binary representation of the input dataset). Since the tuple grouping did not change, the resulting table still satisfies k-anonymity. Let us call this malicious algorithm $\mathfrak{M}_{enc}$. Clearly there exists another algorithm $\mathfrak{M}_{dec}$ that can decode the output of $\mathfrak{M}_{enc}$ (by examining the boundaries of the first bucket) to obtain the original input dataset. Thus $\mathfrak{M}_{dec} \circ \mathfrak{M}_{enc}$ is the identity algorithm (which outputs its inputs) and by Axiom 4.1, if we trust the k-anonymous algorithm $\mathfrak{M}_{enc}$ then we should also trust the identity algorithm as well. If we trust the identity algorithm, then Axiom 4.1 implies that we should trust every algorithm. Thus the consistent normal form of k-anonymity is the set of all algorithms.

  Now, the consistent normal form of k-anonymity clearly does not satisfy the axiom "the algorithm that outputs its inputs should not be trusted" so this implies that k-anonymity by itself is not sufficient to guarantee privacy. While there have been many proposals for fixing k-anonymity (e.g., [45, 48, 66, 16, 67, 68, 31]), it is still not clear whether they permit leakages similar to $\mathfrak{M}_{enc}$ but on a smaller scale (where only some information about a table is encoded in the output). We believe that the consistent normal form will be a useful concept in the creation of a provably secure version of k-anonymity.

4.2 The Row Cone

Having motivated the row cone in Section 2.2.3, we now formally define it and derive its basic properties.

Definition 4.6 (Row Cone). *Let $\mathbb{I} = \{D_1, D_2, \ldots\}$ be the set of possible input datasets and let $\mathfrak{Priv}$ be a privacy definition. The* row cone *of $\mathfrak{Priv}$, denoted by* rowcone($\mathfrak{Priv}$)*, is defined as the set of vectors:*

$$\left\{ \Big(c\,P[\mathfrak{M}(D_1) = \omega],\ c\,P[\mathfrak{M}(D_2) = \omega], \ldots \Big) \ :\ c \geq 0,\ \ \mathfrak{M} \in \text{CNF}(\mathfrak{Priv}),\ \omega \in \text{range}(\mathfrak{M}) \right\}.$$

[6] Recall that k-anonymity [58] is used to anonymize tables. It prespecifies a set of attributes called a *quasi-identifier*. k-Anonymity allows algorithms to coarsen those attributes until each record in the table is indistinguishable from k-1 other records based on the quasi-identifier attributes.

Recalling the matrix representation of algorithms (as discussed in Section 2.2.1 and Figure 2.2.1), we see that a vector belongs to the row cone if and only if it is proportional to some row of the matrix representation of some trusted algorithm $\mathfrak{M} \in \text{CNF}(\mathfrak{Priv})$.

Given a $\mathfrak{M} \in \text{CNF}(\mathfrak{Priv})$ and $\omega \in \text{range}(\mathfrak{M})$, the attacker uses the vector $(P[\mathfrak{M}(D_1) = \omega],\ P[\mathfrak{M}(D_2) = \omega], \dots) \in \text{rowcone}(\mathfrak{Priv})$ to convert the prior distribution $P(\text{data} = D_i)$ to the posterior $P(\text{data} = D_i \mid \mathfrak{M}(\text{data}) = \omega)$. Scaling this likelihood vector by $c > 0$ does not change the posterior distribution, but it does make it easier to work with the row cone.

Constraints satisfied by rowcone($\mathfrak{Priv}$) are therefore constraints shared by all of the likelihood vectors $(P[\mathfrak{M}(D_1) = \omega],\ P[\mathfrak{M}(D_2) = \omega], \dots) \in \text{rowcone}(\mathfrak{Priv})$, and therefore they constrain the ways an attacker's beliefs can change no matter what trusted algorithm $\mathfrak{M} \in \text{CNF}(\mathfrak{Priv})$ is used and what sanitized output $\omega \in \text{range}(\mathfrak{M})$ is produced.

The row cone has an important geometric property:

Theorem 4.7. rowcone($\mathfrak{Priv}$) *is a convex cone.*

Proof. See Appendix 3. □

The fact that the row cone is a convex cone means that it satisfies an associated set of linear constraints (from which we derive semantic privacy guarantees). For technical reasons, the treatment of these constraints differs slightly depending on whether the row cone is finite dimensional (which occurs if the number of possible datasets is finite) or infinite dimensional (if the set of possible datasets is countably infinite). We discuss this next.

4.2.1 Finite Dimensional Row Cones

A closed convex set in finite dimensions is expressible as the solution set to a system of linear inequalities [9]. When the row cone is *closed* then the linear inequalities have the form:

$$\begin{array}{rcl} A_{1,1}P[\mathfrak{M}(D_1) = \omega] + \cdots + A_{1,n}P[\mathfrak{M}(D_n) = \omega] & \geq & 0 \\ A_{2,1}P[\mathfrak{M}(D_1) = \omega] + \cdots + A_{2,n}P[\mathfrak{M}(D_n) = \omega] & \geq & 0 \\ \vdots \qquad\qquad\qquad\qquad \vdots \qquad\qquad\qquad\qquad \vdots & & \vdots \end{array}$$

(with possibly some equalities of the form $B_1P[\mathfrak{M}(D_1) = \omega] + \cdots + B_nP[\mathfrak{M}(D_n) = \omega] = 0$ thrown in). When the row cone is not closed, it is still well-approximated by such linear inequalities: their solution set contains the row cone, and the row cone contains the solution set when the '$\geq$' in the constraints is replaced with '$>$'.

4.2.2 Infinite Dimensional Row Cones

When the domain of the data is countably infinite,[7] vectors in the row cone have infinite length since there is one component for each possible dataset. The vectors in the row cone belong to the vector space ℓ_∞, the set of vectors whose components are bounded. Linear constraints in this vector space can have the form:

$$A_1 P[\mathfrak{M}(D_1) = \omega] + A_2 P[\mathfrak{M}(D_2) = \omega] + \cdots \geq 0 \quad \text{(where } \sum_i |A_i| < \infty\text{)}, \tag{1}$$

but, if one accepts the Axiom of Choice, linear constraints are much more complicated and are generally defined via finitely additive measures [59].[8] On the other hand, in constructive mathematics,[9] more complicated linear constraints cannot be proven to exist ([59], Sections 14.77, 23.10, and 27.45; and [44]). Therefore we only consider the types of linear constraints shown in Equation 1.

4.2.3 Interpretation of Linear Constraints

Starting with a linear inequality of the form $A_1 P(\mathfrak{M}(D_1) = \omega) + A_2 P(\mathfrak{M}(D_2) = \omega) + \cdots \geq 0$, we can separate out the positive coefficients, say $A_{i_1}, A_{i_2}, \ldots$, from the negative coefficients, say $A_{i'_1}, A_{i'_2}, \ldots$, to rewrite it in the form:

$$A_{i_1} P(\mathfrak{M}(D_{i_1}) = \omega) + A_{i_2} P(\mathfrak{M}(D_{i_2}) = \omega) + \cdots \geq |A_{i'_1}|\ P(\mathfrak{M}(D_{i'_1}) = \omega) + |A_{i'_2}|\ P(\mathfrak{M}(D_{i'_2}) = \omega) + \ldots,$$

where all of the coefficients are now positive. We can view each A_{i_j} as a possible value for the prior probability $P(\text{data} = D_{i_j})$ (or a value proportional to a prior probability), setting $S_1 = \{D_{i_1}, D_{i_2}, \ldots\}$ and $S_2 = \{D_{i'_1}, D_{i'_2}, \ldots\}$. This allows us to interpret the linear constraints as statements such as $\alpha P(\text{data} \in S_1, \mathfrak{M}(\text{data}) = \omega) \geq P(\text{data} \in S_2, \mathfrak{M}(\text{data}) = \omega)$. Further algebraic manipulations (and a use of quantities that are constants with respect to $\mathfrak{M}$) result in statements such as:

$$\alpha \geq \frac{P(\text{data} \in S_2 \mid \mathfrak{M}(\text{data}) = \omega)}{P(\text{data} \in S_1 \mid \mathfrak{M}(\text{data}) = \omega)} \tag{2}$$

$$\alpha' \geq \frac{P(\text{data} \in S_2 \mid \mathfrak{M}(\text{data}) = \omega)}{P(\text{data} \in S_1 \mid \mathfrak{M}(\text{data}) = \omega)} \Big/ \frac{P(\text{data} \in S_2)}{P(\text{data} \in S_1)}. \tag{3}$$

Equation 2 means that if an attacker uses a certain class of prior distributions then after seeing the sanitized data, the probability of some set S_2 is no more than α times

[7] We need not consider uncountably infinite domains since digital computers can only process finite bit strings, of which there are countably many.

[8] One example is a "uniform" finitely additive distribution over integers where the set of integers has probability 1 but each integer has probability 0. By finite additivity, each finite set of integers has probability 0 but countably infinite sets of integers can have probability greater than 0 (because countable additivity does not have to hold for finitely additive measures). One should recall the analogy to the uniform distribution over the unit interval $[0, 1]$—each real number has probability 0, but the interval $[0, 1/2)$ has probability $1/2$ and the interval $[0, 1]$ has probability 1. This uniform distribution is countably additive (so any countable set of real numbers has probability 0) but is not *uncountably* additive (so uncountable sets of real numbers can have positive probability).

[9] More precisely, mathematics based on Zermelo-Fraenkel set theory plus the Axiom of Dependent Choice [59].

the probability of some set S_1 (this is a generalization of the idea of γ-amplification for tuples [26] and is related to ideas in [18, 6]). Equation 3 means that if an attacker uses a certain class of priors, then the relative odds of S_2 vs. S_1 can increase by at most α' after seeing the sanitized data.[10]

Of particular importance are the sets S_1 and S_2 of possible input datasets, whose relative probabilities are constrained by the privacy definition. In an ideal world they would correspond to something we are trying to protect (for example, S_1 could be the set of databases in which Bob has cancer and S_2 could be the set of databases in which Bob is healthy). If a privacy definition is not properly designed, S_1 and S_2 could correspond to concepts that may not need protection for certain applications (for example, S_1 could be the set of databases with even parity and S_2 could be the set of databases with odd parity). In any case, it is important to examine existing privacy definitions and even specific algorithms to see which sets they end up protecting.

Note that other methods of interpreting linear constraints (such as [37]) can provide additional notions of semantic guarantees. We believe that studying additional Bayesian interpretations of linear constraints will be useful for future privacy technology.

4.2.4 Discussion and Generalizations

In this section we briefly outline some implications of our discussion about the row cone.

- The development of the row cone was based on two ideas. The first idea is that we would like to understand the protections that a privacy definition offers with respect to attackers who use likelihood-based inference (i.e., the only property of the sanitizing algorithm $\mathfrak{M}$ and output ω that affects their inference is the likelihood vector of ω and this inference is invariant to rescaling of the likelihood vector). The second idea is that we are interested in guarantees that hold no matter which $\mathfrak{M} \in \text{CNF}(\mathfrak{Priv})$ is used to sanitize data and no matter which $\omega \in \text{range}(\mathfrak{M})$ is output. Other types of semantics are also possible; this includes semantics that hold with a high probability (instead of all of the time). The row cone might not be useful for extracting other types of semantics and so other types of mathematical constructs will need to be defined (this is an interesting direction of future work).

- Computing the row cone of a privacy definition involves finding supporting hyperplanes of a convex set. This can be a computationally challenging task. If one is interested in the Bayesian guarantees provided by a privacy definition, then it makes little sense to create a privacy definition for which the row cone is hard to compute (because then it is not clear what are the Bayesian guarantees it provides). One way of simplifying the computation of the row cone is to define a privacy definition in terms of the row cone directly. That is, a sanitizing al-

[10] This idea has influenced the subsequent development of the Pufferfish privacy framework [42]. The linear constraints that characterize privacy definitions in [42] are precisely the constraints on the row cone.

gorithm $\mathfrak{M}$ should belong to a privacy definition $\mathfrak{Priv}$ if every likelihood vector $[P(\mathfrak{M}(D_1) = \omega), P(\mathfrak{M}(D_2) = \omega), \ldots]$ satisfies certain linear constraints. This is the approach implicitly taken by differential privacy [21, 20], Pufferfish [42], and γ-amplification [26].

5 Applications

In this section, we present the main technical contributions of this paper—applications of our framework for the extraction of novel semantic guarantees provided by randomized response, FRAPP/PRAM, and several algorithms (including a generalization of the geometric mechanism [30]) that add integer-valued noise to their inputs. We show that randomized response and FRAPP offer particularly strong protections on different notions of parity of the input data. Since such protections are often unnecessary, we show in Section 5.4 how to manipulate the row cone to relax privacy definitions.

We will make use of the following theorem which shows how to derive CNF($\mathfrak{Priv}$) and rowcone($\mathfrak{Priv}$) for a large class of privacy definitions that are based on a single algorithm.

Theorem 5.1. *Let $\mathbb{I}$ be a finite or countably infinite set of possible datasets. Let $\mathfrak{M}^*$ be an algorithm with* $\text{domain}(\mathfrak{M}^*) = \mathbb{I}$. *Let M^* be the matrix representation of $\mathfrak{M}^*$ (Definition 2.1). If $(M^*)^{-1}$ exists and the L_1 norm of each column of $(M^*)^{-1}$ is bounded by a constant C then*

(1) A bounded row vector $\vec{x} \in \text{rowcone}(\{\mathfrak{M}^*\})$ if and only if $\vec{x} \cdot m \geq 0$ for every column m of $(M^*)^{-1}$.

(2) An algorithm $\mathfrak{M}$, with matrix representation M, belongs to $\text{CNF}(\{\mathfrak{M}^*\})$ if and only if the matrix $M(M^*)^{-1}$ contains no negative entries.

(3) An algorithm $\mathfrak{M}$, with matrix representation M, belongs to $\text{CNF}(\{\mathfrak{M}^*\})$ if and only if every row of M belongs to $\text{rowcone}(\{\mathfrak{M}^*\})$.

Proof. See Appendix 4. □

Note that one of our applications, namely the study of FRAPP/PRAM, does not satisfy the hypothesis of this theorem as it is not based on a single algorithm. Nevertheless, this theorem still turns out to be useful for analyzing FRAPP/PRAM.

5.1 Randomized Response

In this section, we apply our framework to extract Bayesian semantic guarantees provided by randomized response. Recall that randomized response applies to tables with k tuples and a single binary attribute. Thus each database can be represented as a bit string of length k. We formally define the domain of datasets and the randomized response algorithm as follows.

Definition 5.2 (Domain of randomized reponse). *Let the input domain* $\mathbb{I} = \{D_1, \ldots, D_{2^k}\}$ *be the set of all bit strings of length* k*. The bit strings are ordered in reverse lexicographic order. Thus* D_1 *is the string whose bits are all* 1 *and* D_{2^k} *is the string whose bits are all* 0*.*

Definition 5.3 (Randomized response algorithm). *Given a privacy parameter* $p \in [0, 1]$*, let* $\mathfrak{M}_{rr(p)}$ *be the algorithm that, on input* $D \in \mathbb{I}$*, independently flips each bit of* D *with probability* $1 - p$*.*

For example, when $k = 2$ then $|\mathbb{I}| = 4$ and the matrix representation of $\mathfrak{M}_{rr(p)}$ is

$$\begin{array}{c} \\ \omega_1 = 11 \\ \omega_2 = 10 \\ \omega_3 = 01 \\ \omega_4 = 00 \end{array} \begin{array}{c} \begin{array}{cccc} D_1 = 11 & D_2 = 10 & D_3 = 01 & D_4 = 00 \end{array} \\ \begin{pmatrix} p^2 & p(1-p) & p(1-p) & (1-p)^2 \\ p(1-p) & p^2 & (1-p)^2 & p(1-p) \\ p(1-p) & (1-p)^2 & p^2 & p(1-p) \\ (1-p)^2 & p(1-p) & p(1-p) & p^2 \end{pmatrix} \end{array}.$$

Note that randomized response, as a privacy definition, is equal to $\{\mathfrak{M}_{rr(p)}\}$. The next lemma says that without loss of generality, we may assume that $p > 1/2$.

Lemma 5.4. *Given a privacy parameter* p*, define* $q = \max(p, 1 - p)$*. Then*

- $\mathrm{CNF}(\{\mathfrak{M}_{rr(p)}\}) = \mathrm{CNF}(\{\mathfrak{M}_{rr(q)}\})$.
- *If* $p = 1/2$ *then* $\mathrm{CNF}(\{\mathfrak{M}_{rr(p)}\})$ *consists of the set of algorithms whose outputs are statistically independent of their inputs (i.e., those algorithms* $\mathfrak{M}$ *where* $P[\mathfrak{M}(D_i) = \omega] = P[\mathfrak{M}(D_j) = \omega]$ *for all* $D_i, D_j \in \mathbb{I}$*, and* $\omega \in \mathrm{range}(\mathfrak{M})$*), and therefore attackers learn nothing from those outputs.*

Proof. See Appendix 5. □

Therefore, in the remainder of this section, we assume $p > 1/2$ without loss of generality. Now we derive the consistent normal form and row cone of randomized response.

Theorem 5.5 (CNF and row cone). *Given input space* $\mathbb{I} = \{D_1, \ldots, D_{2^k}\}$ *of bit strings of length* k *and a privacy parameter* $p > 1/2$*,*

- *A vector* $\vec{x} = (x_1, \ldots, x_{2^k}) \in \mathrm{rowcone}(\{\mathfrak{M}_{rr(p)}\})$ *if and only if for every bit string* s *of length* k*,*

$$\sum_{i=1}^{2^k} p^{\mathrm{ham}(s, D_i)} (p-1)^{k - \mathrm{ham}(s, D_i)} x_i \geq 0,$$

  where $\mathrm{ham}(s, D_i)$ *is Hamming distance between* s *and* D_i*.*

- *An algorithm* $\mathfrak{M}$ *with matrix representation* M *belongs to* $\mathrm{CNF}(\{\mathfrak{M}_{rr(p)}\})$ *if and only if every row of* M *belongs to* $\mathrm{rowcone}(\{\mathfrak{M}_{rr(p)}\})$*.*

Proof. See Appendix 6. □

We illustrate this theorem with our running example of tables with $k = 2$ tuples.

Example 5.6. (CNF of randomized response, $k = 2$). *Let $p > 1/2$. With two tuples and one binary attribute, the domain $\mathbb{I} = \{11, 10, 01, 00\}$. An algorithm $\mathfrak{M}$ with matrix representation M belongs to the* CNF *of randomized response (with privacy parameter p) if for every vector $\vec{x} = (x_{11}, x_{10}, x_{01}, x_{00})$ that is a row of M, the following four constraints hold:*

$$p^2 x_{00} + (1-p)^2 x_{11} \geq p(1-p)x_{01} + p(1-p)x_{10}. \tag{4}$$
$$(1-p)^2 x_{00} + p^2 x_{11} \geq p(1-p)x_{01} + p(1-p)x_{10}. \tag{5}$$
$$p^2 x_{01} + (1-p)^2 x_{10} \geq p(1-p)x_{00} + p(1-p)x_{11}. \tag{6}$$
$$(1-p)^2 x_{01} + p^2 x_{10} \geq p(1-p)x_{00} + p(1-p)x_{11}. \tag{7}$$

We use Example 5.6 to explain the intuition behind the process of extracting Bayesian semantic guarantees from the row cone of randomized response, as given by the constraints in Equations 4, 5, 6, and 7. Let us consider the following three attackers.

Attacker 1. This attacker has the prior beliefs that $P(\text{data} = 11) = p^2$, $P(\text{data} = 00) = (1-p)^2$, and $P(\text{data} = 01) = P(\text{data} = 10) = p(1-p)$, so that each bit is independent and equals 1 with probability p (this p is the same as the privacy parameter p in randomized response). Let us consider the effect of the constraint in Equation 4 on the attacker's inference. This constraint says that for all $\mathfrak{M}$ in the CNF of randomized response and for all $\omega \in \text{range}(\mathfrak{M})$,

$$p^2 P[\mathfrak{M}(11) = \omega] + (1-p)^2 P[\mathfrak{M}(00) = \omega] \;\geq\; p(1-p)P[\mathfrak{M}(01) = \omega] + p(1-p)P[\mathfrak{M}(10) = \omega]. \tag{8}$$

Note that the coefficients in the linear constraints have the same values as the prior probabilities of the possible input datasets. Substituting those prior beliefs into Equation 8, we get the constraint that for all $\omega \in \text{range}(\mathfrak{M})$:

$$P(\text{data} = 11)P[\mathfrak{M}(11) = \omega] + P(\text{data} = 00)P[\mathfrak{M}(00) = \omega] \;\geq\; P(\text{data} = 01)P[\mathfrak{M}(01) = \omega] + P(\text{data} = 10)P[\mathfrak{M}(10) = \omega],$$

which in turn is equal to the constraint on the attacker's belief about the joint distribution of the input and output of $\mathfrak{M}$:

$$P[\text{parity}(\text{data}) = 0 \wedge \mathfrak{M}(\text{data}) = \omega] \;\geq\; P[\text{parity}(\text{data}) = 1 \wedge \mathfrak{M}(\text{data}) = \omega].$$

Dividing both sides by $P(\mathfrak{M}(\text{data}) = \omega)$ (where data is a random variable), we get the following constraints that $\mathfrak{M}$ imposes on the attacker's posterior distribution:

$$P[\text{parity}(\text{data}) = 0 \mid \mathfrak{M}(\text{data}) = \omega] \;\geq\; P[\text{parity}(\text{data}) = 1 \mid \mathfrak{M}(\text{data}) = \omega].$$

Thus $\mathfrak{M}$ guarantees that if an attacker believes that bits in the database are generated independently with probability p, then after seeing the sanitized output, the attacker will believe that the true input is more likely to have even parity. Also, note that the attacker's *prior* belief about even parity (which is $p^2 + (1-p)^2$) is greater than the attacker's prior belief about odd parity (which is $2p(1-p)$). Therefore $\mathfrak{M}$ guarantees that the attacker will not change his mind about which parity, even or odd, is more likely.

Attacker 2. Now consider a different attacker who believes that the first bit in the true database is 1 with probability $1-p$ and the second bit is 1 with probability p (both bits are still independent). Then, by similar calculations, Equation 6, implies that for this attacker

$$P[\text{parity(data)} = 1 \mid \mathfrak{M}(\text{data}) = \omega] \;\geq\; P[\text{parity(data)} = 0 \mid \mathfrak{M}(\text{data}) = \omega].$$

Thus, after seeing any sanitized output, the attacker will believe that the true input was more likely to have *odd* parity than *even* parity. This attacker's prior belief about odd parity (which is $p^2 + (1-p)^2$) is greater than this attacker's prior belief about even parity (which is $2p(1-p)$). Thus again, any $\mathfrak{M}$ in the CNF of randomized response will ensure that the attacker will not change his mind about the which parity is more likely.

Attacker 3. This attacker believes that the first bit is 1 with probability 1/2 and believes the second bit is 1 with probability p (the bits are independent of each other). In this case, the attacker's prior beliefs are that odd parity and even parity are *equally likely*. It is easy to see that now the output of $\mathfrak{M}$ can make the attacker change his mind about which parity is more likely (for example, consider what happens when $\mathfrak{M}_{rr(p)}$ outputs 01 or 00). This is true because the attacker was so unsure about parity that even the slightest amount of evidence can change his beliefs about which parity is (slightly) more likely. However, the attacker will not change his mind about the parity of the second bit, for which he has greater confidence. This result is a consequence of Theorem 5.7 below, which formally presents the semantic guarantees of randomized response.

The difference between Attacker 3 and Attackers 1, 2 is that Attacker 3 expressed the weakest prior preference between even and odd parity (i.e., 1/2 vs. 1/2). Attackers 1 and 2 had stronger prior beliefs about which parity is more likely and as a result randomized response guarantees that they will not change their minds about which parity is more likely.

The following theorem generalizes these observations to show that randomized response protects the parity of any set of bits whose prior probabilities are $\geq p$ or $\leq 1-p$ (where p is the privacy parameter). It also shows that the only algorithms that have this property are the ones that belong to the trusted set $\text{CNF}(\{\mathfrak{M}_{rr(p)}\})$. Also note that, by Theorem 5.5, an algorithm $\mathfrak{M}$ with matrix representation M belongs to $\text{CNF}(\{\mathfrak{M}_{rr(p)}\})$ if and only if every row of M belongs to $\text{rowcone}(\{\mathfrak{M}_{rr(p)}\})$. Thus the following theorem completely characterizes the privacy guarantees provided by randomized response.[11]

Theorem 5.7. *Let p be a privacy parameter and let $\mathbb{I} = D_1, \ldots, D_{2^k}$. Let $\mathfrak{M}$ be an algorithm that has a matrix representation whose every row belongs to the row cone of randomized response. If the attacker believes that the bits in the data are independent and bit i is equal to 1 with probability q_i, then $\mathfrak{M}$ protects the parity of any subset of bits that have prior probability $\geq p$ or $\leq 1-p$. That is, for any subset $\{\ell_1, \ldots, \ell_m\}$ of bits of the input data such that $q_{\ell_j} \geq p \;\vee\; q_{\ell_j} \leq 1-p$ for $j = 1, \ldots, m$, the following holds:*

[11] All other guarantees are a consequence of them.

- *If* $P(\text{parity}(J) = 0) \geq P(\text{parity}(J) = 1)$ *then* $P(\text{parity}(J) = 0 \mid \mathfrak{M}(\text{data})) \geq P(\text{parity}(J) = 1 \mid \mathfrak{M}(\text{data}))$.
- *If* $P(\text{parity}(J) = 1) \geq P(\text{parity}(J) = 0)$ *then* $P(\text{parity}(J) = 1 \mid \mathfrak{M}(\text{data})) \geq P(\text{parity}(J) = 0 \mid \mathfrak{M}(\text{data}))$.

Furthermore, an algorithm $\mathfrak{M}$ *can only provide these guarantees if every row of its matrix representation belongs to* $\text{rowcone}(\{\mathfrak{M}_{rr(p)}\})$.

Proof. See Appendix 7. □

In many cases, protecting the parity of an entire dataset is not necessary in privacy preserving applications (in fact, some people find it odd).[12] Using the row cone, it is possible to relax a privacy definition to get rid of such unnecessary protections. We discuss this idea in Section 5.4.

5.1.1 The Relationship between Randomized Response and Differential Privacy

When setting $\epsilon = \log \frac{p}{1-p}$, it is well known that randomized response satisfies ϵ-differential privacy. Also, for this parameter setting, differential privacy provides the same protection as randomized response for any given bit in the dataset—a bit corresponds to the record of one individual and differential privacy would allow a bit's value to be retained with probability at most $e^\epsilon/(1 + e^\epsilon) = p$ (and therefore flipped with probability $1 - p$). However, Theorem 5.7 shows that randomized response goes beyond the protection afforded by differential privacy by requiring stronger protection of the parity of larger sets of bits as well.

5.2 FRAPP and PRAM

In some cases, it may be difficult to derive the row cone of a privacy definition $\mathfrak{Priv}$. In these cases, it helps to have some notion of an approximation to a row cone from which semantic guarantees can still be extracted. One might wonder whether the Hausdorff distance [2] or some other measure of distance between sets might be a meaningful measure of the quality of an approximation. Unfortunately it is not at all clear what such a distance measure means in terms of semantic guarantees; finding a meaningful quantitative measure is an interesting open problem.

Thus we take the following approach. If we cannot derive $\text{rowcone}(\mathfrak{Priv})$, our goal becomes to find a strictly larger convex cone r' that contains $\text{rowcone}(\mathfrak{Priv})$. The reason is that any linear inequality satisfied by r' is also satisfied by $\text{rowcone}(\mathfrak{Priv})$; the semantic interpretation of the linear inequality is then a guarantee provided by $\mathfrak{Priv}$. **Thus the approximation may lose some semantics but never generates incorrect semantics.** This idea leads to the following definition.

[12] In this setting, we are normally interested only in the parity of individual bits since each bit corresponds to the value of one individual's record.

Definition 5.8 (Approximation cone). *Given a privacy definition* $\mathfrak{Priv}$*, an* approximation cone *of* $\mathfrak{Priv}$ *is a closed convex cone* r' *such that* rowcone($\mathfrak{Priv}$) $\subseteq r'$.

In this section, we apply this approximation idea to FRAPP [1], which is a privacy definition based on the perturbation technique PRAM [32]. Recall from Section 3.2.2 that the types of algorithms considered by FRAPP are algorithm $\mathfrak{M}_Q$ that have a transition matrix Q where the (a, b) entry, denoted by $P_Q(b \to a)$, is the probability that a tuple with value b gets changed to a. The algorithm $\mathfrak{M}_Q$ modifies each tuple independently using this transition matrix.

Definition 5.9 (Domain of FRAPP). *Define* $\mathcal{TUP} = \{a_1, a_2, \ldots, a_N\}$ *to be the domain of tuples. Choose an arbitrary ordering for these values. Define the data domain to be* $\mathbb{I} = \{D_1, D_2, \ldots\}$*, where each* D_i *is a sequence of* k *tuples from* $\mathcal{TUP}$ *and the list* $D_1, D_2, \ldots$ *is in lexicographic order.*

Definition 5.10 (γ-FRAPP [1]). *Given a privacy parameter* $\gamma \geq 1$*,* γ-FRAPP *is the privacy definition containing all algorithms* $\mathfrak{M}_Q$ *that use transition matrices* Q *with the* γ*-amplification property [26]: for all tuple values* $a, b, c \in \mathcal{TUP}$*,* $\frac{P_Q(b \to a)}{P_Q(c \to a)} \leq \gamma$.

We now construct an approximation cone for γ-FRAPP. If $\mathfrak{M}_Q$ is an algorithm in γ-FRAPP with transition matrix Q, then it is easy to see that the matrix representation of $\mathfrak{M}_Q$, denoted by M_Q, is:

$$M_Q = \bigotimes_{i=1}^{k} Q$$

(where k is the number of tuples in databases from $\mathbb{I}$, and $\bigotimes$ is the Kronecker product).

Let e_j be the column vector of length N that has a 1 in position j and 0 in all other positions. Write $p = \frac{\gamma}{1+\gamma}$ (so that $\gamma = \frac{p}{1-p}$). The constraints imposed on Q by γ-FRAPP can then be written as:

$$\forall i, j \in \{1, \ldots, N\} \ : \ Q(pe_i - (1-p)e_j) \succeq \vec{0},$$

where $\vec{0}$ is the vector containing only 0 components and $\vec{a} \succeq \vec{b}$ means that $\vec{a} - \vec{b}$ has no negative components. Therefore every vector $\vec{x}$ that is the row vector of M_Q, the matrix representation of $\mathfrak{M}_Q$, must satisfy the constraints:

$$\forall i_1, \ldots, i_k, j_1, \ldots, j_k \in \{1, \ldots, N\} \ : \ M_Q \left(\bigotimes_{\ell=1}^{k} (pe_{i_\ell} - (1-p)e_{j_\ell}) \right) \succeq \vec{0}. \tag{9}$$

Using these constraints we can define the Kronecker approximation cone for FRAPP.

Definition 5.11. (Kronecker approximation cone $\tilde{K}_p$). *Given a privacy parameter* γ*, let* $p = \frac{\gamma}{\gamma+1}$*. Define the* Kronecker approximation cone*, denoted by* $\tilde{K}_p$ *to be the set of vectors* $\vec{x}$ *that satisfy the linear constraints in Equation 9 (where* e_{j_ℓ} *is the* j_ℓ^{th} *column vector of the* $N \times N$ *identity matrix).*

Lemma 5.12. *Let $p = \frac{\gamma}{\gamma+1}$. Then $\tilde{K}_p$ is an approximation cone for γ-FRAPP.*

Proof. See Appendix 8. □

The connection between the approximation cone $\tilde{K}_p$ of FRAPP and rowcone($\mathfrak{M}_{rr(p)}$), the row cone of randomized response, is clear once we rephrase the linear constraints that define rowcone($\mathfrak{M}_{rr(p)}$) in Theorem 5.5 as follows:

$$\vec{x} \in \text{rowcone}(\mathfrak{M}_{rr(p)}) \Leftrightarrow \ \forall\ i_1, \ldots, i_k, j_1, \ldots, j_k \in \{1,2\}\ :\ \vec{x} \cdot \left(\bigotimes_{\ell=1}^{k} (p e'_{i_\ell} - (1-p) e'_{j_\ell}) \right) \geq 0,$$

where e_{j_ℓ} is the j_ℓ^{th} column vector of the 2×2 identity matrix.

Thus we can use Theorem 5.7, which gave a semantic interpretation for randomized response to derive some of the semantic guarantees provided by FRAPP.

These guarantees are as follows. Suppose Bob is an attacker who satisfies the following conditions:

- Bob believes that the tuples in the true dataset are independent.
- Bob has ruled out all but two values for the tuple of each individual. That is, for each i, Bob knows that the value of tuple t_i is either some value $a_i \in \mathcal{TUP}$ or $b_i \in \mathcal{TUP}$.
- For each tuple t_i, Bob believes that $t_i = a_i$ with probability q_i, and $t_i = b_i$ with probability $1 - q_i$.

Then for any subset J of the tuples such that $t_i \in J$ only if $q_i \geq p = \frac{\gamma}{1+\gamma}$, if Bob believes $P(\text{parity}(J) = 1) \geq P(\text{parity}(J) = 0)$ then after seeing output ω, Bob believes $P(\text{parity}(J) = 1 \mid \omega) \geq P(\text{parity}(J) = 0 \mid \omega)$, and if Bob believes $P(\text{parity}(J) = 0) \geq P(\text{parity}(J) = 1)$ then $P(\text{parity}(J) = 0 \mid \omega) \geq P(\text{parity}(J) = 1 \mid \omega)$. Here parity can be defined arbitrarily by either treating a_i or b_i as a 1 bit.

In the case of FRAPP, we also see that one of its guarantees is the protection of parity. This seems to be a general property of privacy definitions that are based on algorithms that operate on individual tuples independently.

5.3 Additive Noise

In this section, we analyze a different class of algorithms—those that add noise to their inputs. In the cases we study, the input domain is $\mathbb{I} = \{\ldots, -2, -1, 0, 1, 2, \ldots\}$, and the algorithm being analyzed adds an integer-valued random variable to its input. In the first case that we study (Section 5.3.1), the algorithm adds a random variable of the form $Z = X - Y$ where X and Y have the negative binomial distribution; this includes the geometric mechanism [30] as a special case. In the second case (Section 5.3.2), the algorithm adds a random variable from a Skellam distribution [60], which has the form $Z = X - Y$ where X and Y have Poisson distributions.

5.3.1 Differenced Negative Binomial Mechanism

The Geometric(p) distribution is a probability distribution over nonnegative integers k with mass function $p^k(1-p)$. The negative binomial distribution, NB(p, r), is a probability distribution over nonnegative integers k with mass function $\binom{k+r-1}{k}p^k(1-p)^r$. It is well-known (and easy to show) that an NB(p, r) random variable has the same distribution as the sum of r independent Geometric(p) random variables. In order to get a distribution over the entire set of integers, we can use the difference of two independent NB(p, r) random variables. This leads to the following noise addition algorithm:

Definition 5.13. (Differenced Negative Binomial Mechanism $\mathfrak{M}_{DNB(p,r)}$). *Define $\mathfrak{M}_{DNB(p,r)}$ to be the algorithm that adds $X-Y$ to its input, where X and Y are two independent random variables having the negative binomial distribution with parameters p and r. We call $\mathfrak{M}_{DNB(p,r)}$ the* differenced negative binomial mechanism.

The relationship to the geometric mechanism [30], which adds a random integer k with distribution $\frac{1-p}{1+p}p^{|k|}$, is captured in the following lemma:

Lemma 5.14. *$\mathfrak{M}_{DNB(p,1)}$, the differenced negative binomial mechanism with $r = 1$, is the geometric mechanism.*

Proof. See Appendix 10. □

The following theorem gives us the row cone of the differenced negative binomial mechanism.

Theorem 5.15. *A bounded row vector $\vec{x} = (\dots, x_{-2}, x_{-1}, x_0, x_1, x_2, \dots)$ belongs to* rowcone($\{\mathfrak{M}_{DNB(p,r)}\}$) *if for all integers k,*

$$\forall k: \quad \sum_{j=-r}^{r} (-1)^j f_B\left(j; \frac{p}{1+p}, r\right) x_{k+j} \geq 0,$$

where p and r are the parameters of the differenced negative binomial distribution and $f_B(\cdot; p/(1+p), r)$ is the probability mass function of the difference of two independent binomial (not negative binomial) distributions whose parameters are $p/(1+p)$ (success probability) and r (number of trials).

Proof. See Appendix 11. □

To interpret Theorem 5.15 note that (1) the coefficients of the linear inequality are given by the distribution of the difference of two binomials, (2) the coefficients alternate in signs, and (3) for each integer k, the corresponding linear inequality has the coefficients shifted over by k spots.

One interpretation of Theorem 5.15, therefore, is that if an attacker has managed to rule out all possible inputs except $k-r, k-r+1, \dots, k+r-1, k+r$ and has

a prior on these inputs that corresponds to the difference of two binomials (centered at k), then after seeing the sanitized output of $\mathfrak{M}_{DNB(p,r)}$, the attacker will believe that the set of possible inputs $\{\ldots, k-3, k-1, k+1, \ldots\}$ is not more likely than $\{\ldots, k-4, k-2, k, k+2, \ldots\}$. Again we see a notion of protection of parity but for a smaller set of possible inputs, and note that *initially* this looks like a one-sided guarantee—the posterior probability of odd offsets from k does not increase beyond the posterior probability of the even offsets from k.

However, what is surprising to us is that this kind of guarantee has many strong implications. To illustrate this point, consider $\mathfrak{M}_{DNB(p,1)}$ which is equivalent to the geometric mechanism. The linear inequalities in Theorem 5.15 then simplify (after some simple manipulations) to $-x_{k-1}+(p+1/p)x_k-x_{k+1} \geq 0$ which means that a mechanism must satisfy for all k, $-P[\mathfrak{M}(k-1)=\omega]+(p+1/p)P[\mathfrak{M}(k)=\omega]-P[\mathfrak{M}(k+1)=\omega] \geq 0$. Using these inequalities in the following telescoping sum, we see that they imply the familiar ϵ-differential privacy constraints with $\epsilon = -\log p$ (so $e^\epsilon = 1/p$)).

$$
\begin{aligned}
& p^{-1}P[\mathfrak{M}(k)=\omega] - P[\mathfrak{M}(k-1)=\omega] \\
= & \sum_{j=0}^{\infty} p^j \left(-P[\mathfrak{M}(k-1+j)=\omega] + (p+1/p)P[\mathfrak{M}(k+j)=\omega] - P[\mathfrak{M}(k+1+j)=\omega]\right) \geq 0 \\
& p^{-1}P[\mathfrak{M}(k)=\omega] - P[\mathfrak{M}(k+1)=\omega] \\
= & \sum_{j=0}^{\infty} p^j \left(-P[\mathfrak{M}(k-1-j)=\omega] + (p+1/p)P[\mathfrak{M}(k-j)=\omega] - P[\mathfrak{M}(k+1-j)=\omega]\right) \geq 0.
\end{aligned}
$$

The take-home message, we believe, from this example is that protections on parity, even one-sided protections can be very powerful (for example, we saw how the one-sided protections in Theorem 5.15, can imply the two-sided protections in differential privacy). Thus an interesting direction for future work is to develop methods for analyzing how different guarantees relate to each other; for example, if we protect a fact X, then what else do we end up protecting?

5.3.2 Skellam Noise

In the previous section, we saw how (differenced) negative binomial noise was related to protections against attackers with (differenced) binomial priors, thus exhibiting a dual relationship between the binomial and negative binomial distributions. In this section, we study noise distributed according to the Skellam distribution [60], which turns out to be its own dual.

The Poisson(λ) distribution is a probability distribution over nonnegative integers k with distribution $e^{-\lambda}\frac{\lambda^k}{k!}$. A random variable Z has the Skellam(λ_1, λ_2) distribution if it is equal to the difference $X - Y$ of two independent random variables X and Y having the Poisson(λ_1) and Poisson(λ_2) distributions, respectively [60].

Theorem 5.16. *Let the input domain $\mathbb{I} = \{\ldots, -2, -1, 0, 1, 2, \ldots\}$ be the set of integers. Let $\mathfrak{M}_{skell(\lambda_1,\lambda_2)}$ be the algorithm that adds to its input a random integer k with the Skellam(λ_1, λ_2) distribution and let $f_Z(\cdot; \lambda_1, \lambda_2)$ be the probability mass function of the*

Skellam(λ_1, λ_2) *distribution. A bounded row vector* $\vec{x} = (\ldots, x_{-2}, x_{-1}, x_0, x_1, x_2, \ldots)$ *belongs to* rowcone$(\{\mathfrak{M}_{skell(\lambda_1,\,\lambda_2)}\})$ *if for all integers* k*,*

$$\sum_{j=-\infty}^{\infty} (-1)^j f_Z(j; \lambda_1, \lambda_2) x_{k+j} \geq 0.$$

Proof. See Appendix 9. □

As before, we see that Skellam noise protects parity if the attacker uses a Skellam prior that is shifted[13] by k so that the posterior probability of the set $\{\ldots, k-3, k-1, k+1, k+3, \ldots\}$ cannot be higher than that of the set $\{\ldots, k-2, k, k+2, \ldots\}$.

5.3.3 Other Distributions

When the input domain is the set of integers there is a general technique for deriving the row cone corresponding to an algorithm that adds integer-valued noise to its inputs. If the noise distribution has probability mass function f, then the matrix representation of the noise-addition algorithm is a matrix M (with rows and columns indexed by integers) whose (i, j) entry is $f(i - j)$. One can take the Fourier series transform (characteristic function) $\widehat{f}(t) = \sum_{\ell=-\infty}^{\infty} f(\ell) e^{i\ell t}$. Let g be the inverse transform of $1/\widehat{f}(t)$, if it exists. Then the inverse of the matrix M is a matrix whose (i, j) entries are $g(i - j)$. In combination with Theorem 5.1, this allows one to derive the linear constraints defining the row cone. We used this approach to derive the results of Sections 5.3.1 and 5.3.2 and the proof of Theorem 5.15 provides a formal justification for this technique.

5.4 Relaxing Privacy Definitions

As we saw in Section 5.1, a privacy definition $\mathfrak{Priv}$ may end up protecting more than we want. In such cases, we can manipulate the rowcone$(\mathfrak{Priv})$ to relax it. This will give us a new row cone r and will allow us to create a privacy definition $\mathfrak{Priv}'$ of the form: $\mathfrak{M} \in \mathfrak{Priv}'$ if and only if every row of the matrix representation of $\mathfrak{M}$ belongs to r.

To relax rowcone$(\mathfrak{Priv})$, we will replace the linear constraints that define it with weaker linear constraints. An appropriate tool is Fourier-Motzkin elimination [15], which will produce a new set of linear constraints which are implied by the old constraints. The new constraints will have fewer variables per constraint.

We illustrate this technique by continuing Example 5.6 (randomized response on databases with $k = 2$ tuples). Rewriting Equations 4 and 7 to isolate x_{01} and setting $\alpha = p/(1 - p)$, we get

$$\begin{aligned} \alpha x_{00} + x_{11}/\alpha - x_{10} \geq x_{01} \;\geq\; \alpha x_{00} + \alpha x_{11} - \alpha^2 x_{10} \\ \Rightarrow x_{11} \leq \alpha x_{10}. \end{aligned}$$

[13] I.e., the prior has the distribution of $Z+k$ where k is a constant and Z has the Skellam distribution.

Recalling that x_{11} is shorthand for $P(\mathfrak{M}(11) = \omega)$ and x_{10} is shorthand for $P(\mathfrak{M}(10) = \omega)$, we see that Fourier-Motzkin elimination on the original constraints yielded one of the constraints of $(\ln \frac{p}{1-p})$-differential privacy. Applying Fourier-Motzkin elimination on the other equations in Example 5.6 yields the rest of the differential privacy constraints. Thus we see that differential privacy is a natural relaxation of randomized response.

6 Conclusions

We view privacy as a type of theory of information where the goal is to study how different algorithms filter out certain pieces of information. To this end we proposed the first (to the best of our knowledge) framework for extracting semantic guarantees from privacy definitions. The framework depends on the concepts of consistent normal form $\mathrm{CNF}(\mathfrak{Priv})$ and $\mathrm{rowcone}(\mathfrak{Priv})$. The consistent normal form corresponds to an explicit set of trusted algorithms and the row cone corresponds to the type of information that is always protected by an output of an algorithm belonging to a given privacy definition. The usefulness of these concepts comes from their geometric nature and relations to linear algebra and convex geometry.

There are many important directions for future work. These include extracting semantic guarantees that fail with a small probability, such as various probabilistic relaxations of differential privacy (e.g., [49, 11]). In contrast, the row cone is only useful for finding guarantees that always hold. It is also important to study formal ways of relaxing/strengthening privacy definitions and exploring the relationships between different types of semantic guarantees.

Acknowledgments

This material is based upon work supported by NSF under Grant No. 1054389.

Appendix

1 Proof of Theorem 4.4

Theorem 1.1. (Restatement and proof of Theorem 4.4). *Given a privacy definition* $\mathfrak{Priv}$*, its consistent normal form* $\mathrm{CNF}(\mathfrak{Priv})$ *is equivalent to the following:*

1. *Define* $\mathfrak{Priv}^{(1)}$ *to be the set of all (deterministic and randomized algorithms) of the form* $\mathcal{A} \circ \mathfrak{M}$*, where* $\mathfrak{M} \in \mathfrak{Priv}$*,* $\mathrm{range}(\mathfrak{M}) \subseteq \mathrm{domain}(\mathcal{A})$*, and the random bits of* $\mathcal{A}$ *and* $\mathfrak{M}$ *are independent of each other.*

2. *For any positive integer* n*, finite sequence* $\mathfrak{M}_1, \ldots, \mathfrak{M}_n$ *and probability vector* $\vec{p} = (p_1, \ldots, p_n)$*, use the notation* $\mathrm{choice}^{\vec{p}}(\mathfrak{M}_1, \ldots, \mathfrak{M}_n)$ *to represent the algorithm that runs* $\mathfrak{M}_i$ *with probability* p_i*. Define* $\mathfrak{Priv}^{(2)}$ *to be the set of all algorithms of the form* $\mathrm{choice}^{\vec{p}}(\mathfrak{M}_1, \ldots, \mathfrak{M}_n)$ *where* n *is a positive integer,* $\mathfrak{M}_1, \ldots, \mathfrak{M}_n \in \mathfrak{Priv}^{(1)}$*, and* $\vec{p}$ *is a probability vector.*

3. *Set* $\mathrm{CNF}(\mathfrak{Priv}) = \mathfrak{Priv}^{(2)}$*.*

Proof. We need to show that $\mathfrak{Priv}^{(2)}$ satisfies Axioms 4.1 and 4.2 consistently and that any other privacy definition that satisfies both axioms and contains $\mathfrak{Priv}$ must also contain $\mathfrak{Priv}^{(2)}$.

By construction, $\mathfrak{Priv}^{(2)}$ satisfies Axiom 4.2 (convexity). To show that $\mathfrak{Priv}^{(2)}$ satisfies Axiom 4.1 (post-processing), choose any $\mathfrak{M} \in \mathfrak{Priv}^{(2)}$ and a postprocessing algorithm $\mathcal{A}$. By construction of $\mathfrak{Priv}^{(2)}$, there exists an integer m, a sequence of algorithms $\mathfrak{M}_1^{(1)}, \ldots, \mathfrak{M}_m^{(1)}$ with each $\mathfrak{M}_i^{(1)} \in \mathfrak{Priv}^{(1)}$, and a probability vector $\vec{p} = (p_1, \ldots, p_m)$ such that $\mathfrak{M} = \mathrm{choice}^{p}(\mathfrak{M}_1^{(1)}, \ldots, \mathfrak{M}_m^{(1)})$. It is easy to check that $\mathcal{A} \circ \mathfrak{M} = \mathrm{choice}^{p}(\mathcal{A} \circ \mathfrak{M}_1^{(1)}, \ldots, \mathcal{A} \circ \mathfrak{M}_m^{(1)})$. By construction of $\mathfrak{Priv}^{(1)}$, $\mathcal{A} \circ \mathfrak{M}_i^{(1)} \in \mathfrak{Priv}^{(1)}$ because $\mathfrak{M}_i^{(1)} \in \mathfrak{Priv}^{(1)}$. Therefore, by construction of $\mathfrak{Priv}^{(2)}$, $\mathcal{A} \circ \mathfrak{M} \in \mathfrak{Priv}^{(2)}$, and so $\mathfrak{Priv}^{(2)}$ satisfies Axiom 4.1 (post-processing).

Now let $\mathfrak{Priv}'$ be some privacy definition containing $\mathfrak{Priv}$ and satisfying both axioms. By Axiom 4.1 (post-processing), $\mathfrak{Priv}^{(1)} \subseteq \mathfrak{Priv}'$. By Axiom 4.2 (convexity) it follows that $\mathfrak{Priv}^{(2)} \subseteq \mathfrak{Priv}'$. Therefore $\mathrm{CNF}(\mathfrak{Priv}) = \mathfrak{Priv}^{(2)} \subseteq \mathfrak{Priv}'$. □

2 Proof of Corollary 4.5

Corollary 2.1. (Restatement of Corollary 4.5).
If $\mathfrak{Priv} = \{\mathfrak{M}\}$ *consists of just one algorithm,* $\mathrm{CNF}(\mathfrak{Priv})$ *is the set of all algorithms of the form* $\mathcal{A} \circ \mathfrak{M}$*, where* $\mathrm{range}(\mathfrak{M}) \subseteq \mathrm{domain}(\mathcal{A})$ *and the random bits in* $\mathcal{A}$ *and* $\mathfrak{M}$ *are independent of each other.*

Proof. We use the notation defined in Theorem 4.4. The corollary follows easily from the process described in Theorem 4.4 and the fact that

$$\text{choice}^{\vec{p}}(\mathcal{A}_1 \circ \mathfrak{M}, \ldots, \mathcal{A}_n \circ \mathfrak{M}) = \Big(\text{choice}^{\vec{p}}(\mathcal{A}_1, \ldots, \mathcal{A}_n)\Big) \circ \mathfrak{M}$$

so that the process of computing CNF($\mathfrak{Priv}$) has stopped after the first step. □

3 Proof of Theorem 4.7

Theorem 3.1. (Restatement and proof of Theorem 4.7). rowcone($\mathfrak{Priv}$) *is a convex cone.*

Proof. Choose any $\vec{v} = (v_1, v_2, \ldots) \in \text{rowcone}(\mathfrak{Priv})$. Then by definition $c\vec{v} \in \text{rowcone}(\mathfrak{Priv})$ for any $c \geq 0$. This takes care of the cone property so that we only need to show that rowcone($\mathfrak{Priv}$) is a convex set.

Choose any vectors $\vec{x} = (x_1, x_2, \ldots) \in \text{rowcone}(\mathfrak{Priv})$, $\vec{y} = (y_1, y_2, \ldots) \in \text{rowcone}(\mathfrak{Priv})$, and number t such that $0 \leq t \leq 1$. We show that $t\vec{x} + (1-t)\vec{y} \in \text{rowcone}(\mathfrak{Priv})$. If either $\vec{x} = \vec{0}$ or $\vec{y} = \vec{0}$ then we are done by the cone property we just proved. Otherwise, by definition of row cone, there exist constants $c_1, c_2 > 0$, algorithms $\mathfrak{M}_1, \mathfrak{M}_2 \in \text{CNF}(\mathfrak{Priv})$, and sanitized outputs $\omega_1 \in \text{range}(\mathfrak{M}_1)$, $\omega_2 \in \text{range}(\mathfrak{M}_2)$ such that $\vec{x}/c_1$ is a row of the matrix representation of $\mathfrak{M}_1$ and $\vec{y}/c_2$ is a row of the matrix representation of $\mathfrak{M}_2$:

$$\begin{aligned}
\vec{x} &= \Big(c_1 P[\mathfrak{M}_1(D_1) = \omega_1],\ c_1 P[\mathfrak{M}_1(D_2) = \omega_1], \ldots\Big) \\
\vec{y} &= \Big(c_2 P[\mathfrak{M}_2(D_1) = \omega_2],\ c_2 P[\mathfrak{M}_2(D_2) = \omega_2], \ldots\Big).
\end{aligned}$$

Let $\mathcal{A}_1$ be the algorithm that outputs ω if its input is ω_1 and ω' otherwise. Similarly, let $\mathcal{A}_2$ be the algorithm that outputs ω if its input is ω_2 and ω' otherwise. Define $\mathfrak{M}_1' \equiv \mathcal{A}_1 \circ \mathfrak{M}_1$ and $\mathfrak{M}_2' \equiv \mathcal{A}_2 \circ \mathfrak{M}_2$. Then by Theorem 4.4 (and the post-processing Axiom 4.1), $\mathfrak{M}_1', \mathfrak{M}_2' \in \text{CNF}(\mathfrak{Priv})$ and

$$\begin{aligned}
\vec{x} &= \Big(c_1 P[\mathfrak{M}_1'(D_1) = \omega],\ c_1 P[\mathfrak{M}_1'(D_2) = \omega], \ldots\Big) \\
\vec{y} &= \Big(c_2 P[\mathfrak{M}_2'(D_1) = \omega],\ c_2 P[\mathfrak{M}_2'(D_2) = \omega], \ldots\Big).
\end{aligned}$$

Now consider the algorithm $\mathfrak{M}^*$ which runs $\mathfrak{M}_1'$ with probability $\frac{tc_1}{tc_1 + (1-t)c_2}$ and runs $\mathfrak{M}_2'$ with probability $\frac{(1-t)c_2}{tc_1 + (1-t)c_2}$. By Theorem 4.4, $\mathfrak{M}^* \in \text{CNF}(\mathfrak{Priv})$. Then for all $i = 1, 2, \ldots,$

$$\begin{aligned}
P(\mathfrak{M}^*(D_i) = \omega) &= \frac{tc_1 P(\mathfrak{M}_1'(D_i) = \omega) + (1-t)c_2 P(\mathfrak{M}_2'(D_i) = \omega)}{tc_1 + (1-t)c_2} \\
&= \frac{tx_i + (1-t)y_i}{tc_1 + (1-t)c_2}.
\end{aligned}$$

Thus the vector $\frac{t\vec{x}+(1-t)\vec{y}}{tc_1+(1-t)c_2}$ is the row vector corresponding to ω of the matrix representation of $\mathfrak{M}^*$ and is therefore in rowcone($\mathfrak{Priv}$). Multiplying by the nonnegative constant $tc_1+(1-t)c_2$, we get that $t\vec{x}+(1-t)\vec{y} \in$ rowcone($\mathfrak{Priv}$) and so rowcone($\mathfrak{Priv}$) is convex. □

4 Proof of Theorem 5.1

Theorem 4.1. (Restatement and proof of Theorem 5.1). *Let $\mathbb{I}$ be a finite or countably infinite set of possible datasets. Let $\mathfrak{M}^*$ be an algorithm with* domain($\mathfrak{M}^*$) $= \mathbb{I}$. *Let M^* be the matrix representation of $\mathfrak{M}^*$ (Definition 2.1). If $(M^*)^{-1}$ exists and the L_1 norm of each column of $(M^*)^{-1}$ is bounded by a constant C then*

(1) A bounded row vector $\vec{x} \in$ rowcone($\{\mathfrak{M}^*\}$) *if and only if $\vec{x} \cdot m \geq 0$ for every column m of $(M^*)^{-1}$.*

(2) An algorithm $\mathfrak{M}$, with matrix representation M, belongs to CNF($\{\mathfrak{M}^*\}$) *if and only if the matrix $M(M^*)^{-1}$ contains no negative entries.*

(3) An algorithm $\mathfrak{M}$, with matrix representation M, belongs to CNF($\{\mathfrak{M}^*\}$) *if and only if every row of M belongs to* rowcone($\{\mathfrak{M}^*\}$).

Proof. We first prove (1). If $\vec{x}$ is the 0 vector then this is clearly true. Thus assume $\vec{x} \neq \vec{0}$. If $\vec{x} \in$ rowcone($\{\mathfrak{M}^*\}$) then by definition of the row cone and by Corollary 4.5, $\vec{x} = \vec{y}M^*$ where $\vec{y}$ is a bounded row vector and has nonnegative components. Then $\vec{x}(M^*)^{-1} = \vec{y}M^*(M^*)^{-1} = \vec{y}$ and so $\vec{x} \cdot m \geq 0$ for every column m of $(M^*)^{-1}$.

For the other direction, we must construct an algorithm $\mathcal{A}$ with matrix representation A such that for some $c > 0$, $c\vec{x}$ is a row of AM^* (by definition of row cone and Corollary 4.5). Thus, by hypothesis, suppose $\vec{x} \cdot m \geq 0$ for each column vector m of $(M^*)^{-1}$ and consider the row vector $\vec{y} = \vec{x}(M^*)^{-1}$ which therefore has nonnegative entries. Since $\vec{x}$ is bounded and $||m||_1 \leq C$ for each column vector m of $(M^*)^{-1}$ then $|\vec{x} \cdot m| \leq ||\vec{x}||_\infty ||m||_1 \leq ||\vec{x}||_\infty C$ (by Hölder's Inequality [56]) so that $\vec{y}$ is bounded. Choose a c so that $c\vec{y}$ is bounded by 1. Consider the algorithm $\mathcal{A}$ that has a matrix representation A with two rows, the first row being $c\vec{y}$ and the second row being $1 - c\vec{y}$ ($\mathcal{A}$ is an algorithm since $c\vec{y}$ and $1 - c\vec{y}$ have nonnegative components and the column sums of A are clearly 1). $\mathcal{A}$ is the desired algorithm since $c\vec{x}$ is a row of AM^*.

To prove (2) and (3), note that if an algorithm has matrix representation M, then $M(M^*)^{-1}$ contains all the dot products between rows of M and columns of $(M^*)^{-1}$. Therefore, the entries of $M(M^*)^{-1}$ are nonnegative if and only if every row of M is in the rowcone($\{\mathfrak{M}^*\}$) (this follows directly from the first part of the theorem). Thus (2) and (3) are equivalent and therefore we only need to prove (2).

To prove (2), first note the trivial direction. If $\mathfrak{M} \in$ CNF($\{\mathfrak{M}^*\}$) then by definition every row of M is in the row cone (and so by (1) all entries of $M(M^*)^{-1}$ are nonnegative). For the other direction, let $A = M(M^*)^{-1}$ (which has no negative entries by hypothesis). If we can show that the column sums of A are all 1 then, since A contains no negative

entries, A would be a column stochastic matrix and therefore it would be the matrix representation of some algorithm $\mathcal{A}$. From this it would follow that $AM^* = M$ and therefore $\mathcal{A} \circ \mathfrak{M}^* = \mathfrak{M}$ (in which case $\mathfrak{M} \in \text{CNF}(\mathfrak{M}^*)$ by Theorem 4.4).

So all we need to do is to prove that the column sums of A are all 1. Let $\vec{1}$ be a column vector whose components are all 1. Then since M is a matrix representation of an algorithm (Definition 2.1), M has column sums equal to 1, and similarly for M^*. Thus:

$$\begin{aligned} \vec{1}^T &= \vec{1}^T M^* (M^*)^{-1} \\ &= \vec{1}^T (M^*)^{-1} \\ & \quad \text{and therefore} \\ \vec{1}^T A &= \vec{1}^T M (M^*)^{-1} \\ &= \vec{1}^T (M^*)^{-1} \\ &= \vec{1}^T, \end{aligned}$$

and so the column sums of A are equal to 1. This completes the proof of this theorem. □

5 Proof of Lemma 5.4

Lemma 5.1. (Restatement and proof of Lemma 5.4). *Given a privacy parameter p, define $q = \max(p, 1-p)$. Then*

- $\text{CNF}(\{\mathfrak{M}_{rr(p)}\}) = \text{CNF}(\{\mathfrak{M}_{rr(q)}\})$.
- *If $p = 1/2$ then $\text{CNF}(\{\mathfrak{M}_{rr(p)}\})$ consists of the set of algorithms whose outputs are statistically independent of their inputs (i.e., those algorithms $\mathfrak{M}$ where $P[\mathfrak{M}(D_i) = \omega] = P[\mathfrak{M}(D_j) = \omega]$ for all $D_i, D_j \in \mathbb{I}$ and $\omega \in \text{range}(\mathfrak{M})$), and therefore attackers learn nothing from those outputs.*

Proof. Consider the algorithm $\mathfrak{M}_{rr(0)}$ which always flips each bit in its input. It is easy to see that $\mathfrak{M}_{rr(0)} \circ \mathfrak{M}_{rr(p)} = \mathfrak{M}_{rr(1-p)}$ and $\mathfrak{M}_{rr(0)} \circ \mathfrak{M}_{rr(1-p)} = \mathfrak{M}_{rr(p)}$. From Theorem 4.4, it follows that $\text{CNF}(\{\mathfrak{M}_{rr(p)}\}) = \text{CNF}(\{\mathfrak{M}_{rr(1-p)}\})$ and therefore $\text{CNF}(\{\mathfrak{M}_{rr(p)}\}) = \text{CNF}(\{\mathfrak{M}_{rr(q)}\})$.

Clearly, the output of $\mathfrak{M}_{rr(1/2)}$ is independent of whatever was the true input table $D \in \mathbb{I}$. By Theorem 4.4, all algorithms in $\text{CNF}(\{\mathfrak{M}_{rr(1/2)}\})$ have outputs independent of their inputs. For the other direction, choose any algorithm $\mathfrak{M}$ whose outputs are statistically independent of their inputs. Then it is easy to see that $\mathfrak{M} = \mathfrak{M} \circ \mathfrak{M}_{rr(1/2)}$; that is, $\mathfrak{M}$ and $\mathfrak{M} \circ \mathfrak{M}_{rr(1/2)}$ have the same range and $P(\mathfrak{M}(D_i) = \omega) = P([\mathfrak{M} \circ \mathfrak{M}_{rr(1/2)}](D_i) = 0)$ for all $D_i \in \mathbb{I}$ and $\omega \in \text{range}(\mathfrak{M})$. Thus $\mathfrak{M} \in \text{CNF}(\{\mathfrak{M}_{rr(1/2)}\})$.

Clearly, when the output is statistically independent of the input, an attacker can learn nothing about the input after observing the output. □

6 Proof of Theorem 5.5

Theorem 6.1. (Restatement and proof of Theorem 5.5). *Given input space* $\mathbb{I} = \{D_1, \ldots, D_{2^k}\}$ *of bit strings of length* k *and a privacy parameter* $p > 1/2$,

- *A vector* $\vec{x} = (x_1, \ldots, x_{2^k}) \in \text{rowcone}(\{\mathfrak{M}_{rr(p)}\})$ *if and only if for every bit string* s *of length* k,
$$\sum_{i=1}^{2^k} p^{\text{ham}(s,D_i)}(p-1)^{k-\text{ham}(s,D_i)} x_i \geq 0,$$
where $\text{ham}(s, D_i)$ *is the Hamming distance between* s *and* D_i.
- *An algorithm* $\mathfrak{M}$ *with matrix representation* M *belongs to* $\text{CNF}(\{\mathfrak{M}_{rr(p)}\})$ *if and only if every row of* M *belongs to* $\text{rowcone}(\{\mathfrak{M}_{rr(p)}\})$.

Proof. Our strategy is to first derive the matrix representation of $\mathfrak{M}_{rr(p)}$, which we denote by $M_{rr(p)}$. Then we find the inverse of $M_{rr(p)}$ and apply Theorem 5.1. Accordingly, we break the proof down into three steps.

Step 1: Derive $M_{rr(p)}$. Define B to be the matrix

$$B = \begin{pmatrix} p & 1-p \\ 1-p & p \end{pmatrix}.$$

Recall that the Kronecker product $C \oplus D$ of an $m \times n$ matrix C and $m' \times n'$ matrix D is the block matrix $\begin{pmatrix} c_{11}D & \ldots & c_{1n}D \\ \vdots & \ddots & \vdots \\ c_{m1}D & \ldots & c_{mn}D \end{pmatrix}$ of dimension $mm' \times nn'$. An easy induction shows that the matrix representation $M_{rr(p)}$ is equal to the k-fold Kronecker product of B with itself:

$$M_{rr(p)} = \bigotimes_{i=1}^{k} B.$$

The entry in row i and column j of $M_{rr(p)}$ is equal to $P[\mathfrak{M}_{rr(P)}(D_j) = D_i]$ and a direct computation shows that this is equal to

$$p^{\text{ham}(D_i,D_j)}(1-p)^{k-\text{ham}(D_i,D_j)}$$

Step 2: Derive $(M_{rr(p)})^{-1}$. It is easy to check that

$$B^{-1} = \frac{1}{2p-1}\begin{pmatrix} p & -(1-p) \\ -(1-p) & p \end{pmatrix}$$

and therefore

$$(M_{rr(p)})^{-1} = \bigotimes_{i=1}^{k} B^{-1}.$$

A comparison with $\bigotimes_{i=1}^{k} B^{-1}$ shows that the we can calculate the entry in row i and column j of $(M_{rr(p)})^{-1}$ by taking the corresponding entry of $M_{rr(p)}$ and replacing every occurrence of $1-p$ with $-(1-p) = p-1$. Thus the entry in row i and column j of $(M_{rr(p)})^{-1}$ is equal to

$$\frac{1}{(2p-1)^k} p^{\text{ham}(D_i,D_j)} (p-1)^{k-\text{ham}(D_i,D_j)}.$$

Therefore each column of $(M_{rr(p)})^{-1}$ has the form:

$$\frac{1}{(2p-1)^k} \begin{bmatrix} p^{\text{ham}(s,D_1)}(p-1)^{k-\text{ham}(s,D_1)} \\ p^{\text{ham}(s,D_2)}(p-1)^{k-\text{ham}(s,D_2)} \\ \vdots \\ p^{\text{ham}(s,D_{2^k})}(p-1)^{k-\text{ham}(s,D_{2^k})} \end{bmatrix}.$$

Step 3: Now we apply Theorem 5.1 and observe that if $m^{(i)}$ is the i^{th} column of $(M_{rr(p)})^{-1}$, then, since $p > 1/2$ and $2p-1 > 0$, the condition $\vec{x} \cdot m^{(i)}$ is equal to the condition

$$\sum_{j=1}^{2^k} p^{\text{ham}(s,D_j)} (p-1)^{k-\text{ham}(s,D_j)} x_j \geq 0,$$

where $s = D_i$. □

7 Proof of Theorem 5.7

Theorem 7.1. (Restatement and proof of Theorem 5.7). *Let p be a privacy parameter and let $\mathbb{I} = D_1, \ldots, D_{2^k}$. Let $\mathfrak{M}$ be an algorithm that has a matrix representation whose every row belongs to the row cone of randomized response. If the attacker believes that the bits in the data are independent and bit i is equal to 1 with probability q_i, then $\mathfrak{M}$ protects the parity of any subset of bits that have prior probability $\geq p$ or $\leq 1-p$. That is, for any subset $\{\ell_1, \ldots, \ell_m\}$ of bits of the input data such that $q_{\ell_j} \geq p \ \vee \ g_{\ell_j} \leq 1-p$ for $j = 1, \ldots, m$, the following holds:*

- *If* $P(\text{parity}(J) = 0) \geq P(\text{parity}(J) = 1)$ *then* $P(\text{parity}(J) = 0 \mid \mathfrak{M}(\text{data})) \geq P(\text{parity}(J) = 1 \mid \mathfrak{M}(\text{data}))$.
- *If* $P(\text{parity}(J) = 1) \geq P(\text{parity}(J) = 0)$ *then* $P(\text{parity}(J) = 1 \mid \mathfrak{M}(\text{data})) \geq P(\text{parity}(J) = 0 \mid \mathfrak{M}(\text{data}))$.

Furthermore, an algorithm $\mathfrak{M}$ can only provide these guarantees if every row of its matrix representation belongs to $\text{rowcone}(\{\mathfrak{M}_{rr(p)}\})$.

Proof. We break this proof up into a series of steps. We first reformulate the statements to make them easier to analyze mathematically, then we specialize to the case where

$J = \{1, \ldots, k\}$ is the set of all bits in the database. We then show that every $\mathfrak{M}$ whose rows (in the corresponding matrix representation) belong to rowcone($\mathfrak{M}_{rr(p)}$) has these semantic guarantees. We then show that only those $\mathfrak{M}$ provide these semantic guarantees. Finally we show that those results imply that the theorem holds for all J whose bits have prior probability $\geq p$ or $\leq 1 - p$.

Step 1: Problem reformulation and specialization to the case when $J = \{1, \ldots, k\}$. Assume $J = \{1, \ldots, k\}$ so that for all bits j, either $q_j \geq p$ or $q_j \leq 1 - p$.

First, Lemma 5.4 allows us to assume that the privacy parameter $p > 1/2$ without any loss of generality: the case of $p = 1/2$ is trivial since the output provides no information about the input so that parity is preserved; in the case of $p < 1/2$, the row cone and CNF are unchanged if we replace p with $1 - p$.

Second, we need a few results about parity. An easy induction shows that:

$$\begin{aligned} P(\text{parity(data)} = 1) &= \frac{1 - \prod_{j=1}^{k} (1 - 2q_j)}{2} \\ P(\text{parity(data)} = 0) &= \frac{1 + \prod_{j=1}^{k} (1 - 2q_j)}{2}. \end{aligned}$$

In particular, if all of the $q_j \neq 1/2$, then $P(\text{parity(data)} = 1) \neq P(\text{parity(data)} = 0)$ so that one parity has higher prior probability than the other.

When J is the set of all k bits, then for all q_j, $q_j \neq 1/2$ and so the parities cannot be equally likely a priori, the statement about protection of parity can be rephrased as $P(\text{parity(data)} = 0) - P(\text{parity(data)} = 1)$, and $P(\text{parity(data)} = 0 \mid \mathfrak{M}(\text{data})) - P(\text{parity(data)} = 1 \mid \mathfrak{M}(\text{data}))$ have the same sign or the posterior probabilities of parity are the same. Equivalently,

$$\begin{aligned} 0 \leq & \Big(P[\text{parity(data)} = 0] - P[\text{parity(data)} = 1]\Big) \\ & \times \Big(P[\text{parity(data)} = 0 \mid \mathfrak{M}(\text{data})] - P[\text{parity(data)} = 1 \mid \mathfrak{M}(\text{data})]\Big). \end{aligned} \quad (10)$$

Now, it is easy to see that

$$\begin{aligned} & P(\text{parity(data)} = 0) - P(\text{parity(data)} = 1) \\ = & \left[\bigotimes_{j=1}^{k} (-q_j,\ 1 - q_j)\right] \cdot \left[\bigotimes_{j=1}^{k} (1, 1)\right] \\ = & \prod_{j=1}^{k} \Big[(-q_j,\ 1 - q_j) \cdot (1, 1)\Big] \end{aligned} \quad (11)$$

and

$$P[\text{parity(data)} = 0 \mid \mathfrak{M}(\text{data})] - P[\text{parity(data)} = 1 \mid \mathfrak{M}(\text{data})] = \alpha \left[\bigotimes_{j=1}^{k} (-q_j, 1-q_j) \right] \cdot \vec{x}, \tag{12}$$

where α is a positive normalizing constant and $\vec{x}$ is a vector of the matrix representation of $\mathfrak{M}$. So, by Equations 10, 11, and 12, the statement about protecting parity is equivalent to

$$\forall \vec{x} \in \text{rowcone}(\{\mathfrak{M}_{rr(p)}\}) \ : \ 0 \leq \left(\prod_{j=1}^{k} \left[(-q_j,\ 1-q_j) \cdot (1,1) \right] \right) * \left(\left[\bigotimes_{j=1}^{k} (-q_j,\ 1-q_j) \right] \cdot \vec{x} \right). \tag{13}$$

Step 2: Show that if for all j, $q_j \geq p \vee q_j \leq 1-p$ then the constraints in Equation 13 hold (i.e., the most likely parity a priori is the most likely parity a posteriori).

It follows from Corollary 4.5 that every $\mathfrak{M} \in \text{CNF}(\{\mathfrak{M}_{rr(p)}\})$ has the form $\mathcal{A} \circ \mathfrak{M}_{rr(p)}$ and so, by Theorem 5.1, $\vec{x}$ is a row from the matrix representation of an $\mathfrak{M} \in \text{CNF}(\{\mathfrak{M}_{rr(p)}\})$ if and only if $\vec{x} \in \text{rowcone}(\{\mathfrak{M}_{rr(p)}\})$. This means that every such $\vec{x}$ is a nonnegative linear combination of rows of the randomized response algorithm $\mathfrak{M}_{rr(p)}$. Thus it suffices to show that

$$0 \quad \leq \quad \left(\prod_{j=1}^{k} \left[(-q_j,\ 1-q_j) \cdot (1,1) \right] \right) * \left(\left[\bigotimes_{j=1}^{k} (-q_j,\ 1-q_j) \right] \cdot \vec{m} \right) \tag{14}$$

for each vector $\vec{m}$ in $M_{rr(p)}$ (the matrix representation of $\mathfrak{M}_{rr(p)}$). It is easy to check that

$$M_{rr(p)} = \bigotimes_{i=1}^{k} \begin{pmatrix} p & 1-p \\ 1-p & p \end{pmatrix},$$

and so every vector $\vec{m}$ that is a row of $M_{rr(p)}$ has the form $\bigotimes_{i=1}^{k} v_i$ where $v_i = (p,\ 1-p)$ or $(1-p,\ p)$. Thus right hand side of Equation 14 has the form:

$$\prod_{j=1}^{k} \left[\Big((-q_j,\ 1-q_j) \cdot (1,1) \Big) * \Big((-q_j,\ 1-q_j) \cdot v_i \Big) \right], \tag{15}$$

where $v_i = (p,\ 1-p)$ or $(1-p,\ p)$. Each term in this product is either

$$(1-2q_j) * [(1-p)(1-q_j) - q_j p] = (1-2q_j)[1-p-q_j]$$

or

$$(1-2q_j) * [p(1-q_j) - q_j(1-p)] = (1-2q_j)[p-q_j].$$

Recalling that we had assumed $p > 1/2$ without any loss of generality, both of these terms are nonnegative if $q_j \geq p > 1/2$, and they are also nonnegative when $q_i \leq (1-p) < 1/2$. Thus the product in Equation 15 is nonnegative from which it follows that the conditions in Equation 14 and 13 are satisfied which implies Equation 10 is satisfied, which proves half of the theorem when restricted to the special case of $J = \{1, \ldots, k\}$.

Step 3: Show that if $\mathfrak{M}$ is a mechanism that protects parity whenever $q_j \geq p \ \vee \ q_j \leq 1-p$ for $i = 1, \ldots, k$ then every row $\vec{x}$ in its matrix representation belongs to rowcone($\{\mathfrak{M}_{rr(p)}\}$).

We actually prove a more general statement: if $\mathfrak{M}$ is a mechanism that protects parity whenever $q_j = p \ \vee \ q_j = 1-p$ for $i = 1, \ldots, k$ then every row $\vec{x}$ in its matrix representation belongs to rowcone($\{\mathfrak{M}_{rr(p)}\}$).

Recalling the argument leading up to Equation 13 in Step 2 (where we reformulated the problem into a statement that is more amenable to mathematical manipulation), we need to show that if

$$0 \leq \left(\prod_{j=1}^{k} \Big[(-q_j, \ 1-q_j) \cdot (1,1) \Big] \right) * \left(\left[\bigotimes_{j=1}^{k} (-q_j, \ 1-q_j) \right] \cdot \vec{x} \right) \tag{16}$$

whenever $q_j = p$ or $q_j = 1-p$, then $\vec{x} \in$ rowcone($\{\mathfrak{M}_{rr(p)}\}$).

Define the function:

$$\text{sign}(\alpha) = \begin{cases} -1 & \text{if } \alpha < 0 \\ 0 & \text{if } \alpha = 0 \\ 1 & \text{if } \alpha > 0 \end{cases}.$$

Simplifying Equation 16 (by computing the dot product in the first term, looking just at the sign of that dot product, and then combining both terms), our goal is to show that if

$$0 \quad \leq \quad \left(\left[\bigotimes_{j=1}^{k} (-q_j, \ 1-q_j) * \text{sign}(1-2q_j) \right] \cdot \vec{x} \right) \tag{17}$$

whenever $q_j = p$ or $q_j = 1-p$, then $\vec{x} \in$ rowcone($\{\mathfrak{M}_{rr(p)}\}$).

Now, when $q_j = p$ (and recalling that we have assumed $p > 1/2$ with no loss of generality in Step 1), then

$$(-q_j, \ 1-q_j) * \text{sign}(1-2q_j) = (p, \ -(1-p)),$$

and when $q_j = 1-p$ then

$$(-q_j, \ 1-q_j) * \text{sign}(1-2q_j) = (-(1-p), \ p).$$

Thus asserting that Equation 17 holds whenever q_j equals p or $1-p$ is the same as asserting that the vector:

$$\vec{x}^T \bigoplus_{i=1}^{k} \frac{1}{2p-1} \begin{pmatrix} p & -(1-p) \\ -(1-p) & p \end{pmatrix} \tag{18}$$

has no negative components. However, the randomized response algorithm $\mathfrak{M}_{rr(p)}$ has a matrix representation $M_{rr(p)}$ whose inverse (which we also derived in the proof of Theorem 5.5) is

$$(M_{rr(p)})^{-1} = \bigoplus_{i=1}^{k} \frac{1}{2p-1} \begin{pmatrix} p & -(1-p) \\ -(1-p) & p \end{pmatrix}.$$

Thus the condition that the vector in Equation 18 has no negative entries means that $\vec{x}^T (M_{rr(p)})^{-1}$ has no negative entries and so the dot product of $\vec{x}$ with any column of $(M_{rr(p)})^{-1}$ is nonnegative. By Theorem 5.5, this means that $\vec{x} \in \text{rowcone}(\{\mathfrak{M}_{rr(p)}\})$.

This concludes the proof for the entire theorem specialized to the case where $J = \{1, \ldots, k\}$. In the next step, we generalize this to arbitrary J.

Step 4: Now let $J = \{\ell_1, \ldots, \ell_m\}$. First consider an "extreme" attacker whose prior beliefs q_j are such that $q_j = 0$ or $q_j = 1$ whenever $j \notin J$. It follows from the previous steps that such an attacker would not change his mind about the parity of the whole dataset. Since the attacker is completely sure about the values of bits outside of J, this means that after seeing a sanitized output ω, the attacker will not change his mind about the parity of the bits in J.

Now, note that showing

- If $P(\text{parity}(J) = 0) \geq P(\text{parity}(J) = 1)$ then
  $P(\text{parity}(J) = 0 \mid \mathfrak{M}(\text{data}) = \omega) \geq P(\text{parity}(J) = 1 \mid \mathfrak{M}(\text{data}) = \omega)$
- If $P(\text{parity}(J) = 1) \geq P(\text{parity}(J) = 0)$ then
  $P(\text{parity}(J) = 1 \mid \mathfrak{M}(\text{data}) = \omega) \geq P(\text{parity}(J) = 0 \mid \mathfrak{M}(\text{data}) = \omega)$

is equivalent to showing

- If $P(\text{parity}(J) = 0) \geq P(\text{parity}(J) = 1)$ then
  $P(\text{parity}(J) = 0 \ \wedge \ \mathfrak{M}(\text{data})) \geq P(\text{parity}(J) = 1 \ \wedge \ \mathfrak{M}(\text{data}))$
- If $P(\text{parity}(J) = 1) \geq P(\text{parity}(J) = 0)$ then
  $P(\text{parity}(J) = 1 \ \wedge \ \mathfrak{M}(\text{data}) = \omega) \geq P(\text{parity}(J) = 0 \ \wedge \ \mathfrak{M}(\text{data}) = \omega)$,

since we just multiply the equations on both sides of the inequalities by the positive number $P(\mathfrak{M}(\text{data}) = \omega)$.

Now consider an attacker Bob such that $q_j \geq p$ or $q_j \leq 1-p$ whenever $j \in J$, and there are no restrictions on q_j for $j \notin J$. There is a corresponding set of $2^{k-|J|}$

"extreme" attackers for whom $P(\text{bit } j = 1) = q_j$ for $j \in J$ and $P(\text{bit } j = 1) \in \{0, 1\}$ otherwise.

Bob's vector of prior probabilities over possible datasets

$$(P[\text{data} = D_1], P[\text{data} = D_2], \dots)$$

is a convex combination of the corresponding vectors for the extreme attackers. and thus Bob's joint distributions:

$$P(\text{parity}(J) = 1 \ \wedge \ \mathfrak{M}(\text{data}) = \omega)$$

and

$$P(\text{parity}(J) = 0 \ \wedge \ \mathfrak{M}(\text{data}) = \omega)$$

are convex combinations of the corresponding posteriors for the extreme attackers, and the coefficients of this convex combination are the same.

Note that Bob and all of the extreme attackers have the same prior on the parity of J. However, we have shown that the extreme attackers will not change their minds about the parity of J. Therefore if they believe $P(\text{parity}(J) = 1 \ \wedge \ \mathfrak{M}(\text{data}) = \omega)$ is larger than the corresponding probability for even parity, then Bob will have the same belief. If the extreme attackers believe, after seeing the sanitized output ω, that even parity is more likely, then so will Bob. Thus Bob will not change his belief about the parity of the input dataset. □

8 Proof of Lemma 5.12

Lemma 8.1. (Restatement and proof of Lemma 5.12). *Let $p = \frac{\gamma}{\gamma+1}$. Then $\tilde{K}_p$ is an approximation cone for* γ-FRAPP.

Proof. Clearly $\tilde{K}_p$ is a closed convex cone. Thus we just need to prove that rowcone(γ-FRAPP) $\subseteq \tilde{K}_p$.

Choose any $\mathfrak{M}_Q \in \gamma$-FRAPP, with matrix representation M_Q. Clearly

$$M_Q = \bigotimes_{i=1}^{k} Q,$$

and Q satisfies the constraints

$$\forall i, j \in \{1, \dots, N\} \ : \ Q(pe_i - (1-p)e_j) \succeq \vec{0}$$

where e_i is the i^{th} column vector of the $N \times N$ identity matrix and $\vec{a} \succeq \vec{b}$ means that $\vec{a} - \vec{b}$ has no negative components. It follows from the properties of the Kronecker product that

$$\forall i_1, \dots, i_k, j_1, \dots, j_k \in \{1, \dots, N\} \ : \ M_Q \left(\bigotimes_{\ell=1}^{k} (pe_{i_\ell} - (1-p)e_{j_\ell}) \right) \succeq \vec{0}. \tag{19}$$

Thus each row of the matrix representation of $\mathfrak{M}_Q$ satisfies a set of linear constraints.

From Theorem 4.4, we see that CNF(γ-FRAPP) can be obtained by first creating all algorithms of the form $\mathcal{A} \circ \mathfrak{M}_Q$ (for $\mathfrak{M}_Q \in \gamma$-FRAPP) and then by taking the convex combination of those results (i.e., creating algorithms that randomly chooses to run one of the algorithms generated in the previous step). However, the matrix representation of $\mathcal{A} \circ \mathfrak{M}_Q$ is equal to AM_Q (where A is the matrix representation of $\mathcal{A}$) and every row in AM_Q is a positive linear combination of rows in $\mathfrak{M}_Q$. Thus every row of the matrix representation of $\mathcal{A} \circ \mathfrak{M}_Q$ also satisfies the constraints defining $\tilde{K}_p$. Finally, creating an algorithm $\mathcal{A}^*$ that randomly choose to run one algorithm in $\{\mathcal{A}_1 \circ \mathfrak{M}_{Q_1}, \ldots, \mathcal{A}_h \circ \mathfrak{M}_{Q_h}\}$ means that the rows in the matrix representation of $\mathcal{A}^*$ are a convex combination of the rows appearing in the matrix representations of the $\mathcal{A}_i \circ \mathfrak{M}_{Q_i}$ and so those rows also satisfy the constraints that define $\tilde{K}_p$. Therefore rowcone(γ-FRAPP) $\subseteq \tilde{K}_p$. □

9 Proof of Theorem 5.16

Theorem 9.1. *(*Restatement and proof of Theorem 5.16*) Let the input domain* $\mathbb{I} = \{\ldots, -2, -1, 0, 1, 2, \ldots\}$ *be the set of integers. Let* $\mathfrak{M}_{skell(\lambda_1, \lambda_2)}$ *be the algorithm that adds to its input a random integer* k *with the Skellam*(λ_1, λ_2) *distribution and let* $f_Z(\cdot; \lambda_1, \lambda_2)$ *be the probability mass function of the Skellam*(λ_1, λ_2) *distribution. A bounded row vector* $\vec{x} = (\ldots, x_{-2}, x_{-1}, x_0, x_1, x_2, \ldots)$ *belongs to* rowcone$(\{\mathfrak{M}_{skell(\lambda_1, \lambda_2)}\})$ *if for all integers* k*,*

$$\sum_{j=-\infty}^{\infty} (-1)^j f_Z(j; \lambda_1, \lambda_2) x_{k+j} \geq 0.$$

Proof. For integers k, define the functions

$$f_X(k) = \begin{cases} e^{-\lambda_1} \frac{\lambda_1^k}{k!} & \text{if } k \geq 0 \\ 0 & \text{if } k < 0 \end{cases}$$

$$f_Y(k) = \begin{cases} e^{-\lambda_2} \frac{\lambda_2^{-k}}{(-k)!} & \text{if } k \leq 0 \\ 0 & \text{if } k > 0. \end{cases}$$

Note that f_X is the probability mass function for a Poisson(λ_1) random variable X while f_Y is the probability mass function of the **negative** of a Poisson(λ_2) random variable Y.

With this notation, the Skellam distribution is the distribution of the sum $X + Y$. Therefore its probability mass function satisfies the following relation

$$f_Z(k; \lambda_1, \lambda_2) = \sum_{j=-\infty}^{\infty} f_X(k - j) f_Y(j) = (f_X \star f_Y)(k),$$

where $f_X \star f_Y$ is the convolution operation.

Now for each integer k define

$$\begin{aligned}
g_X(k) &= (-1)^k f_X(k) \\
g_Y(k) &= (-1)^k f_Y(k) \\
g_Z(k) &= (g_X \star g_Y)(k) \\
&= \sum_{j=-\infty}^{\infty} g_X(k-j) g_Y(j) \\
&= \sum_{j=-\infty}^{\infty} (-1)^{k-j} f_X(k-j)(-1)^j f_Y(j) \\
&= (-1)^k \sum_{j=-\infty}^{\infty} f_X(k-j) f_Y(j) \\
&= (-1)^k f_Z(k; \lambda_1, \lambda_2).
\end{aligned}$$

We will need the following calculations:

$$\begin{aligned}
(g_X \star f_X)(k) &= \sum_{j=-\infty}^{\infty} g_X(k-j) f_X(j) \\
&= \sum_{j=-\infty}^{\infty} (-1)^{k-j} f_X(k-j) f_X(j) \\
&= \sum_{j=0}^{k} (-1)^{k-j} f_X(k-j) f_X(j) \\
&\quad \text{(since } f_X \text{ is 0 for negative integers also note the summation is 0 if } j > k\text{)} \\
&= \mathbb{1}_{\{k \geq 0\}} e^{-2\lambda_1} \sum_{j=0}^{k} \frac{(-\lambda_1)^{k-j}}{(k-j)!} \frac{\lambda_1^j}{j!} \\
&= \mathbb{1}_{\{k \geq 0\}} \frac{e^{-2\lambda_1}}{k!} \sum_{j=0}^{k} \binom{k}{j} (-\lambda_1)^{k-j} \lambda_1^j \\
&= \mathbb{1}_{\{k \geq 0\}} \frac{e^{-2\lambda_1}}{k!} (\lambda_1 - \lambda_1)^k \\
&= \begin{cases} e^{-2\lambda_1} \text{ if } k = 0 \\ 0 \text{ otherwise.} \end{cases}
\end{aligned}$$

Similarly,

$$\begin{aligned}
(g_Y \star f_Y)(k) &= \sum_{j=-\infty}^{\infty} g_Y(k-j) f_Y(j) \\
&= \sum_{j=-\infty}^{\infty} (-1)^{k-j} f_Y(k-j) f_Y(j) \\
&= \sum_{j=k}^{0} (-1)^{k-j} f_Y(k-j) f_Y(j) \\
&\quad \text{(since } f_Y \text{ is 0 for positive integers also note the summation is 0 if } k > j\text{)} \\
&= \mathbb{1}_{\{k \le 0\}} e^{-2\lambda_2} \sum_{j=k}^{0} \frac{(-\lambda_2)^{-(k-j)}}{(-(k-j))!} \frac{\lambda_2^{-j}}{(-j)!} \\
&= \mathbb{1}_{\{k \le 0\}} e^{-2\lambda_2} \sum_{j=0}^{-k} \frac{(-\lambda_2)^{(-k)-j}}{[(-k)-j]!} \frac{\lambda_2^{j}}{j!} \\
&\quad \text{(replacing the dummy index } j \text{ with } -j\text{)} \\
&= \mathbb{1}_{\{k \le 0\}} \frac{e^{-2\lambda_2}}{(-k)!} \sum_{j=0}^{-k} \binom{(-k)}{j} (-\lambda_2)^{(-k)-j} \lambda_2^{j} \\
&= \mathbb{1}_{\{k \le 0\}} \frac{e^{-2\lambda_2}}{(-k)!} (\lambda_2 - \lambda_2)^{(-k)} \\
&= \begin{cases} e^{-2\lambda_2} \text{ if } k = 0 \\ 0 \text{ otherwise.} \end{cases}
\end{aligned}$$

From these calculations we can conclude that

$$\begin{aligned}
(g_Z \star f_Z(\cdot; \lambda_1, \lambda_2))(k) &= ((g_X \star g_Y) \star (f_X \star f_Y))(k) \\
&= ((g_X \star f_X) \star (g_Y \star f_Y))(k) \\
&\quad \text{(since convolutions are commutative and associative)} \\
&= \begin{cases} e^{-2(\lambda_1+\lambda_2)} & \text{if } k = 0 \\ 0 & \text{otherwise.} \end{cases}
\end{aligned}$$

These convolution calculations show that the matrices $M^{(f)}$ and $M^{(g)}$, whose rows and columns are indexed by the integers and which are defined below, are inverses of each other.

$$\begin{aligned}
M^{(f)}_{(i,j)} &\equiv (i,j) \text{ entry of } M^{(f)} \\
&= f_Z(i-j; \lambda_1, \lambda_2) \\
M^{(g)}_{(i,j)} &\equiv (i,j) \text{ entry of } M^{(g)} \\
&= e^{2(\lambda_1+\lambda_2)} g_Z(i-j).
\end{aligned}$$

To see that they are inverses, note that the dot product between row r of $M^{(f)}$ and column c of $M^{(g)}$ is

$$\begin{aligned}
\sum_{j=-\infty}^{\infty} M^{(f)}_{(r,j)} M^{(g)}_{(j,c)} &= \sum_{j=-\infty}^{\infty} f_Z(r-j;\lambda_1,\lambda_2) e^{2(\lambda_1+\lambda_2)} g_Z(j-c) \\
&= \sum_{j=-\infty}^{\infty} f_Z(r-c-j;\lambda_1,\lambda_2) e^{2(\lambda_1+\lambda_2)} g_Z(j) \\
&= e^{2(\lambda_1+\lambda_2)} (f_Z(\cdot;\lambda_1,\lambda_2) \star g_Z)(r-c) \\
&= e^{2(\lambda_1+\lambda_2)} (g_Z \star f_Z(\cdot;\lambda_1,\lambda_2))(r-c) \\
&= \begin{cases} 1 & \text{if } r = c \\ 0 & \text{otherwise.} \end{cases}
\end{aligned}$$

Now, clearly $M^{(f)}$ is the matrix representation of $\mathfrak{M}_{\text{skell}(\lambda_1,\,\lambda_2)}$ so that we can again use Theorem 5.1 and the observation that $g_Z(k) = (-1)^k f_Z(k;\lambda_1,\lambda_2)$ so that column c of $M^{(g)} = (M^{(f)})^{-1}$ is the column vector whose entry j is $(-1)^{j-c} f_Z(j-c;\lambda_1,\lambda_2)$.

Note that the columns of $M^{(g)}$ have bounded L_1 norm since the absolute value of the entries in any column are proportional to the probabilities given by the Skellam distribution.

The proof is completed by the observation that for any $\vec{x} = (\dots, x_{-2}, x_{-1}, x_0, x_1, x_2, \dots)$,

$$\sum_{j=-\infty}^{\infty} (-1)^{j-c} f_Z(j-c;\lambda_1,\lambda_2) x_j = \sum_{j=-\infty}^{\infty} (-1)^j f_Z(j;\lambda_1,\lambda_2) x_{j+c}.$$

□

10 Proof of Lemma 5.14

Lemma 10.1. (Proof and restatement of Lemma 5.14). *$\mathfrak{M}_{DNB(p,1)}$, the differenced negative binomial mechanism with $r = 1$, is the geometric mechanism.*

Proof. We need to show that the difference between two independent Geometric(p) distributions has the probability mass function $f(k) = \frac{1-p}{1+p} p^{|k|}$.

Let X and Y be independent Geometric(p) random variables and let $Z = X - Y$. Then

$$P(Z=k) = \begin{cases} \sum_{j=0}^{\infty} P(X=j+k)P(Y=j) & \text{if } k \geq 0 \\ \sum_{i=0}^{\infty} P(X=j)P(Y=j+|k|) & \text{if } k < 0. \end{cases}$$

Combining both cases, we get

$$\begin{aligned} P(Z=k) &= \sum_{j=0}^{\infty}(1-p)p^{j+|k|}(1-p)p^{j} \\ &= (1-p)^2p^{|k|}\sum_{j=0}^{\infty}(p^2)^j \\ &= (1-p)^2p^{|k|}\frac{1}{1-p^2} \\ &= (1-p)^2p^{|k|}\frac{1}{(1-p)(1+p)} \\ &= \frac{1-p}{1+p}p^{|k|}. \end{aligned}$$

□

11 Proof of Theorem 5.15

We first need an intermediate result.

Lemma 11.1. *Let X and Y be independent random variables with the Binomial$(\frac{p}{1+p}, r)$ distribution (where $p/(1+p)$ is the success probability and r is the number of trials). Let $Z = X - Y$ and let $f_B\left(k; \frac{p}{p+1}, r\right) = P(Z = k)$ for integers $k = -r, \ldots, 0, \ldots r$. Define the function h as $h(k) = (-1)^k f_B\left(k; \frac{p}{p+1}, r\right)$. The Fourier series transform $\widehat{h}$ of h (defined as $\widehat{h}(t) = \sum_{\ell=-\infty}^{\infty} h(\ell)e^{i\ell t}$) is equal to*

$$\widehat{h}(t) = \frac{1}{(1+p)^{2r}}(1-pe^{it})^r(1-pe^{-it})^r.$$

Proof. Define the random variable $Y' = -Y$. Then $X + Y' = Z$. Thus

$$\begin{aligned}
\widehat{h}(t) &= \sum_{\ell=-\infty}^{\infty} h(\ell) e^{i\ell t} \\
&= \sum_{\ell=-\infty}^{\infty} (-1)^{\ell} f_B\left(\ell; \frac{p}{p+1}, r\right) e^{i\ell t} \\
&= \sum_{\ell=-\infty}^{\infty} (-1)^{\ell} e^{i\ell t} P(Z = \ell) \\
&= \sum_{\ell=-\infty}^{\infty} e^{i\ell t} (-1)^{\ell} \sum_{j=-\infty}^{\infty} P(X = \ell - j) P(Y' = j) \\
&= \sum_{\ell=-\infty}^{\infty} e^{i\ell t} \sum_{j=-\infty}^{\infty} (-1)^{\ell-j} P(X = \ell - j)(-1)^{j} P(Y' = j) \\
&= \sum_{\ell=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} (-1)^{\ell-j} e^{i(\ell-j)t} P(X = \ell - j)(-1)^{j} e^{ijt} P(Y' = j) \\
&= \sum_{j=-\infty}^{\infty} (-1)^{j} e^{ijt} P(Y' = j) \sum_{\ell=-\infty}^{\infty} (-1)^{\ell-j} e^{i(\ell-j)t} P(X = \ell - j).
\end{aligned}$$

Now,

$$\begin{aligned}
&\sum_{\ell=-\infty}^{\infty} (-1)^{\ell-j} e^{i(\ell-j)t} P(X = \ell - j) \\
&\quad = \sum_{\ell=-\infty}^{\infty} (-1)^{\ell} e^{i\ell t} P(X = \ell) \\
&\quad = \sum_{\ell=0}^{r} (-1)^{\ell} e^{i\ell t} P(X = \ell) \\
&\quad \text{(Since } X \text{ can only be } 0, \ldots, r\text{)} \\
&\quad = \sum_{\ell=0}^{r} (-1)^{\ell} e^{i\ell t} \binom{r}{\ell} \left(\frac{p}{1+p}\right)^{\ell} \left(\frac{1}{1+p}\right)^{r-\ell} \\
&\quad = \frac{1}{(1+p)^r} \sum_{\ell=0}^{r} (-1)^{\ell} e^{i\ell t} \binom{r}{\ell} p^{\ell} \\
&\quad = \frac{1}{(1+p)^r} \sum_{\ell=0}^{r} \binom{r}{\ell} (-pe^{it})^{\ell} \\
&\quad = \frac{1}{(1+p)^r} (1 - pe^{it})^{r} \text{ by the Binomial theorem.}
\end{aligned}$$

Thus continuing our previous calculation,

$$\begin{aligned}
\widehat{h}(t) &= \sum_{j=-\infty}^{\infty} (-1)^j e^{ijt} P(Y' = j) \frac{1}{(1+p)^r} (1 - pe^{it})^r \\
&= \sum_{j=-r}^{0} (-1)^j e^{ijt} P(Y' = j) \frac{1}{(1+p)^r} (1 - pe^{it})^r \\
&\quad \text{(since } Y' \text{ can only be } -r, \ldots, 0) \\
&= \sum_{j=-r}^{0} (-1)^j e^{ijt} P(Y = -j) \frac{1}{(1+p)^r} (1 - pe^{it})^r \\
&\quad \text{(since } Y' = -Y \text{)} \\
&= \sum_{j=0}^{r} (-1)^j e^{-ijt} P(Y = j) \frac{1}{(1+p)^r} (1 - pe^{it})^r.
\end{aligned}$$

Now, similar to what we did before, we can derive that $\sum_{j=0}^{r} (-1)^j e^{-ijt} P(Y = j) = \frac{1}{(1+p)^r}(1 - pe^{-it})^r$ and therefore

$$\widehat{h}(t) = \frac{1}{(1+p)^{2r}} (1 - pe^{it})^r (1 - pe^{-it})^r.$$

□

Theorem 11.2. (Restatement and proof of Theorem 5.15). *A bounded row vector* $\vec{x} = (\ldots, x_{-2}, x_{-1}, x_0, x_1, x_2, \ldots)$ *belongs to* rowcone($\{\mathfrak{M}_{DNB(p,r)}\}$) *if for all integers* k,

$$\forall k: \quad \sum_{j=-r}^{r} (-1)^j f_B\left(j; \frac{p}{1+p}, r\right) x_{k+j} \geq 0.$$

where p *and* r *are the parameters of the differenced negative binomial distribution and* $f_B(\cdot; p/(1+p), r)$ *is the probability mass function of the difference of two independent binomial (not negative binomial) distributions whose parameters are* $p/(1+p)$ *(success probability) and* r *(number of trials).*

Proof. For convenience, define the function h as follows:

$$h(j) = (-1)^j f_B\left(j; \frac{p}{1+p}, r\right).$$

Let $g_{NB}(\cdot; p, r)$ be the probability distribution function for the difference of two independent NB(p, r) random variables. Then the matrix representation $M_{DNB(p,r)}$ of the differenced negative binomial mechanism $\mathfrak{M}_{DNB(p,r)}$ is the matrix whose rows and columns are indexed by the integers and whose entries are defined as:

$$(i, j) \text{ entry of } M_{DNB(p,r)} = g_{NB}(i - j; p, r).$$

By Theorem 5.1 we need to show that $M_{DNB(p,r)}$ is the inverse of $\frac{(1+p)^{2r}}{(1-p)^{2r}}H$ where H is the matrix whose rows and columns are indexed by the integers and whose entries are defined as:

$$(i,j) \text{ entry of } H \quad = \quad h(i-j) = (-1)^{i-j} f_B\left(i-j; \frac{p}{1+p}, r\right)$$

(to see how Theorem 5.1 is applied, note that each entry of the product $\vec{x}H$ has the form $\sum_{j=-r}^{r} (-1)^j f_B\left(j; \frac{p}{1+p}, r\right) x_{k+j}$).

Now, to show that $M_{DNB(p,r)}$ and $\frac{(1+p)^{2r}}{(1-p)^{2r}}H$ are inverses of each other, we note that

$$\begin{aligned}
&(i,j) \text{ entry of } \left(M_{DNB(p,r)}H\right) \\
&= \sum_{\ell=-\infty}^{\infty} g_{NB}(i-\ell; p, r) h(\ell - j) \\
&= \sum_{\ell'=-\infty}^{\infty} g_{NB}(i-j-\ell'; p, r) h(\ell') \\
&= \sum_{\ell'=-r}^{r} g_{NB}(i-j-\ell'; p, r) h(\ell'). \qquad (20)
\end{aligned}$$

The last step follows from the fact that $f_B(\ell'; p, r)$ and $h(\ell')$ are nonzero only when ℓ' is between $-r$ and r since $f_B(\cdot; p, r)$ is the probability mass function of the difference of two binomial random variables (each of which is bounded between 0 and r).

Now, Equation 20 is the definition of the convolution [56] of $g_{NB}(\cdot; p, r)$ and h at the point $i-j$. That is,

$$(g_{NB}(\cdot; p, r) \star h)(k) = \sum_{\ell'=-r}^{r} g_{NB}(k-\ell'; p, r) h(\ell'),$$

and thus to show that $M_{DNB(p,r)}$ and $\frac{(1+p)^{2r}}{(1-p)^{2r}}H$ are inverses of each other, we just need to show that the convolution of $g_{NB}(\cdot; p, r)$ and h at the point 0 is equal to $\frac{(1-p)^{2r}}{(1+p)^{2r}}$, and that the convolution at all other integers is 0. In other words, we want to show that for all integers k,

$$(g_{NB}(\cdot; p, r) \star h)(k) = \frac{(1-p)^{2r}}{(1+p)^{2r}}\ \delta(k), \qquad (21)$$

where δ is the function that $\delta(0) = 1$ and $\delta(k) = 0$ for all other integers. Take the Fourier series transform of both sides while noting two facts: (1) the Fourier series transform of δ is $\widehat{\delta}(t) = \sum_{\ell=-\infty}^{\infty} \delta(\ell) e^{i\ell t} \equiv 1$, and (2) the Fourier transform of a convolution is the

product of the Fourier transforms [56]. Then the transformed version of Equation 21 becomes

$$\widehat{g_{NB}}(t)\ \widehat{h}(t) = \frac{(1-p)^{2r}}{(1+p)^{2r}}\widehat{\delta}(t) \equiv \frac{(1-p)^{2r}}{(1+p)^{2r}} \tag{22}$$

for all real t, where $\widehat{g_{NB}}$, $\widehat{h}$, $\widehat{\delta}$ are the Fourier series transforms of $g_{NB}(\cdot; p, r)$, h, and δ, respectively. Once we prove that Equation 22 is true, this implies Equation 21 is true (by the inverse Fourier transform) which then implies that $M_{DNB(p,r)}$ and $\frac{(1+p)^{2r}}{(1-p)^{2r}}H$ are inverses of each other and this would finish the proof (by Theorem 5.1).

Thus our goal is to prove Equation 22. The Fourier series transform (i.e., characteristic function), as a function of t, of the NB(p, r) distribution is known to be:

$$\left(\frac{1-p}{1-pe^{it}}\right)^r,$$

so $g_{NB}(\cdot; p, r)$, being the difference of two independent negative binomial random variables, has the Fourier series transform (as a function of t)

$$\widehat{g_{NB}}(t) \quad = \quad \left(\frac{1-p}{1-pe^{it}}\right)^r \left(\frac{1-p}{1-pe^{-it}}\right)^r.$$

By Lemma 11.1,

$$\widehat{h}(t) = \frac{1}{(1+p)^{2r}}(1-pe^{it})^r(1-pe^{-it})^r.$$

Thus Equation 22 is true and we are done. □

References

[1] Agrawal, S. and Haritsa, J. R. (2005). A framework for high-accuracy privacy-preserving mining. In *Proceedings of the 21st International Conference on Data Engineering (ICDE 2005), Tokyo, Japan*. IEEE Computer Society.

[2] Aliprantis, C. D. and Border, K. C. (2007). *Infinite Dimensional Analysis: A Hitchhiker's Guide*. 3rd edition. Springer.

[3] Backstrom, L., Dwork, C., and Kleinberg, J. (2007). Wherefore art thou r3579x?: Anonymized social networks, hidden patterns, and structural steganography. In *Proceedings of the 16th International World Wide Web Conference (WWW2007), Banff, Alberta, Canada*. ACM Press. 181–190.

[4] Barbaro, M. and Zeller, T. (2006). A face is exposed for AOL searcher no. 4417749. *New York Times*, August 9. `http://www.nytimes.com/2006/08/09/technology/09aol.html?pagewanted=all&_r=0`

[5] Bhaskar, R., Bhowmick, A., Goyal, V., Laxman, S., and Thakurta, A. (2011). Noiseless database privacy. In Lee, D. H. and Wang, X. (eds), *Advances in Cryptology - ASIACRYPT 2011- 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea,* vol. 7073 of *LNCS*. Springer. 215–232.

[6] Blum, A., Dwork, C., McSherry, F., and Nissim, K. (2005). Practical privacy: The SuLQ framework. In *Proceedings of the 24th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS)*. ACM Press. 128–138.

[7] Blum, A., Ligett, K., and Roth, A. (2008). A learning theory approach to non-interactive database privacy. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC 2008), Victoria, British Columbia, Canada*. ACM Press. 609–618.

[8] Boreale, M. and Paolini, M. (2012). Worst- and average-case privacy breaches in randomization mechanisms. In *Proceedings of the IFIP Theoretical Computer Science Conference (TCS 2012)*, vol. 7604 of *LNCS*. Springer. 72–86.

[9] Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. New York, NY: Cambridge University Press.

[10] Casella, G. and Berger, R. L. (2002). *Statistical Inference*. Duxbury, 2nd edition.

[11] Chaudhuri, K. and Mishra, N. (2006). When random sampling preserves privacy. In *Proceedings of the 26th Annual International Conference on Advances in Cryptology (CRYPTO '06)*. Springer-Verlag. 198–213.

[12] Chen, B.-C., Kifer, D., LeFevre, K., and Machanavajjhala, A. (2009). Privacy-preserving data publishing. *Foundations and Trends in Databases*, 2(1-2):1–167.

[13] Choromanski, K. and Malkin, T. (2012). The power of the Dinur-Nissim algorithm: Breaking privacy of statistical and graph databases. In *Proceedings of the 31st Symposium on Principles of Database Systems (PODS '12)*. ACM Press. 65–76.

[14] Clifton, C., Kantarcioglu, M., and Vaidya, J. (2002). Defining privacy for data mining. In *Proceedings of the National Science Foundation Workshop on Next Generation Data Mining*. 126–133.

[15] Cook, W. J., Cunningham, W. H., Pulleyblank, W. R., and Schrijver, A. (1998). *Combinatorial Optimization*. New York, NY: John Wiley & Sons, Inc.

[16] Cormode, G., Srivastava, D., Li, N., and Li, T. (2010). Minimizing minimality and maximizing utility: Analyzing method-based attacks on anonymized data. *Proceedings of the VLDB Endowment*, 3(1–2):1045–1056.

[17] Dalenius, T. (1986). Finding a needle in a haystack, or identifying anonymous census records. *Journal of Official Statistics*, 2(3):329–336.

[18] Dinur, I. and Nissim, K. (2003). Revealing information while preserving privacy. In *Proceedings of the 22nd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS '03)*. ACM Press. 202–210.

[19] Duncan, G. T. and Lambert, D. (1989). The risk of disclosure for microdata. *Journal of Business and Economic Statistics*, 7(2):207–217.

[20] Dwork, C. (2006). Differential privacy. In Bugliesi, M., Preneel, B., Sassone, V., and Wegener, I. (eds), *Automata, Languages and Programming, 33rd International Colloquium (ICALP 2006), Venice, Italy, Proceedings, Part II*, vol. 4052 of *LNCS*. Springer. 1–12.

[21] Dwork, C., McSherry, F., Nissim, K., and Smith, A. (2006). Calibrating noise to sensitivity in private data analysis. In *Proceedings of the 3rd Conference on Theory of Cryptography (TCC '06)*. Springer. 265–284.

[22] Dwork, C., McSherry, F., and Talwar, K. (2007). The price of privacy and the limits of LP decoding. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing (STOC '07)*. ACM Press. 85–94.

[23] Dwork, C. and Naor, M. (2010). On the difficulties of disclosure prevention in statistical databases or the case for differential privacy. *Journal of Privacy and Confidentiality*, 2(1):93–107.

[24] Dwork, C., Naor, M., Reingold, O., N.Rothblum, G., and Vadhan, S. (2009). On the complexity of differentially private data release: Efficient algorithms and hardness results. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing (STOC '09)*. ACM Press. 381–390.

[25] Dwork, C. and Yekhanin, S. (2008). New efficient attacks on statistical disclosure control mechanisms. In *Advances in Cryptology (CRYPTO 2008)*, vol. 5157 of *LNCS*. Springer. 469–480.

[26] Evfimievski, A., Gehrke, J., and Srikant, R. (2003). Limiting privacy breaches in privacy-preserving data mining. In *Proceedings of the 22nd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS '03)*. ACM Press. 211–222.

[27] Fang, C. and Chang, E.-C. (2008). Information leakage in optimal anonymized and diversified data. In *Information Hiding*, vol. 5284 of *LNCS*. Springer. 30–44.

[28] Ganta, S. R., Kasiviswanathan, S. P., and Smith, A. (2008). Composition attacks and auxiliary information in data privacy. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '08)*. ACM Press. 265–273.

[29] Gehrke, J., Lui, E., and Pass, R. (2011). Towards privacy for social networks: A zero-knowledge based definition of privacy. In *Proceedings of the 8th Conference on Theory of Cryptography (TCC '11)*. Springer. 432–449.

[30] Ghosh, A., Roughgarden, T., and Sundararajan, M. (2009). Universally utility-maximizing privacy mechanisms. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing (STOC '09)*. ACM Press. 351–360.

[31] Gionis, A., Mazza, A., and Tassa, T. (2008). k-anonymization revisited. In *Proceedings of the 2008 IEEE 24th International Conference on Data Engineering (ICDE '08)*. Washington, D.C.: IEEE Computer Society. 744–753.

[32] Gouweleeuw, J., Kooiman, P., Willenborg, L., and de Wolf, P.-P. (1998). Post randomisation for statistical disclosure control: Theory and implementation. *Journal of Official Statistics*, 14(4):463–478.

[33] Huang, Z., Du, W., and Chen, B. (2004). Deriving private information from randomized data. In *Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data (SIGMOD '05)*. ACM Press. 37–48.

[34] Kaggle (2013). http://www.kaggle.com.

[35] Kargupta, H., Datta, S., Wang, Q., and Sivakumar, K. (2003). On the privacy preserving properties of random data perturbation techniques. In *Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM '03)*. Washington, D.C: IEEE Computer Society. 99.

[36] Kasiviswanathan, S. P., Rudelson, M., Smith, A., and Ullman, J. (2010). The price of privately releasing contingency tables and the spectra of random matrices with correlated rows. In *Proceedings of the 42nd ACM Symposium on Theory of Computing (STOC '10)*. ACM Press. 775–784.

[37] Kasiviswanathan, S. P. and Smith, A. (2008). A note on differential privacy: Defining resistance to arbitrary side information. `http://arxiv.org/abs/0803.3946`.

[38] Kifer, D. (2009). Attacks on privacy and de Finetti's theorem. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data (SIGMOD '09)*. ACM Press. 127–138.

[39] Kifer, D. and Lin, B.-R. (2010). Towards an axiomatization of statistical privacy and utility. In *Proceedings of the 29th ACM SIGMOD-SIGART-SIGART Symposium on Principles of Database Systems (PODS '10)*. ACM Press. 147-158.

[40] — (2012). An axiomatic view of statistical privacy and utility. *Journal of Privacy and Confidentiality*, 4(1):5–49.

[41] Kifer, D. and Machanavajjhala, A. (2011). No free lunch in data privacy. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data (SIGMOD '11)*. ACM Press. 193–204.

[42] — (2012). A rigorous and customizable framework for privacy. In *Proceedings of the 31st Symposium on Principles of Database Systems (PODS '12)*. ACM Press. 77–88.

[43] Lambert, D. (1993). Measures of disclosure risk and harm. *Journal of Official Statistics*, 9(2):313–331.

[44] Lauwers, L. (2010). Purely Finitely Additive Measures are Non-constructible Objects. Technical report, Center for Economic Studies, K.U. Leuven. Working paper.

[45] Li, N., Li, T., and Venkatasubramanian, S. (2007). t-Closeness: Privacy beyond k-anonymity and l-diversity. In *Proceedings of the 23rd International Conference on Data Engineering (ICDE 2007), Istanbul, Turkey*.Washington, D.C.: IEEE Computer Society. 106–115.

[46] Lin, B.-R. and Kifer, D. (2012). Reasoning about privacy using axioms. In *Proceedings of the 2012 Conference Record of the 46th Asilomar Conference on Signals, Systems and Computers (ASILOMAR), Pacific Grove, CA, USA.* Washington, D.C.: IEEE Computer Society. 975–979.

[47] Liu, K., Giannella, C., and Kargupta, H. (2008). *A survey of attack techniques on privacy-preserving data perturbation methods*, chapter 15 in Aggarwal, C. C. and Yu, P. S. (eds), *Privacy-Preserving Data Mining*, vol. 34 of *Advances in Database Systems*. Springer. 357–380.

[48] Machanavajjhala, A., Gehrke, J., Kifer, D., and Venkitasubramaniam, M. (2006). ℓ-diversity: Privacy beyond k-anonymity. In *Proceedings of the 23rd International Conference on Data Engineering (ICDE 2007), Istanbul, Turkey.* Washington, D.C.: IEEE Computer Society. 106–115.

[49] Machanavajjhala, A., Kifer, D., Abowd, J., Gehrke, J., and Vilhuber, L. (2008). Privacy: From theory to practice on the map. In *Proceedings of the 24th International Conference on Data Engineering (ICDE 2008), Cancun, Mexico.* Washington, D.C.: IEEE Computer Society. 277–286.

[50] McClure, D. and Reiter, J. P. (2012). Differential privacy and statistical disclosure risk measures: An investigation with binary synthetic data. *Transactions on Data Privacy*, 5(3):535–552.

[51] Miklau, G. and Suciu, D. (2004). A formal analysis of information disclosure in data exchange. In *Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data (SIGMOD '04)*. ACM Press. 575–586.

[52] Narayanan, A. and Shmatikov, V. (2006). How to break anonymity of the netflix prize dataset. `http://www.citebase.org/abstract?id=oai:arXiv.org:cs/0610105`

[53] — (2009). De-anonymizing social networks. In *Proceedings of the 2009 30th IEEE Symposium on Security and Privacy, Berkeley, CA, USA*. Washington, D.C.: IEEE Computer Society. 173–187.

[54] Rastogi, V., Hay, M., Miklau, G., and Suciu, D. (2009). Relationship privacy: Output perturbation for queries with joins. In *Proceedings of the 28th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS '09)*. ACM Press. 107–116.

[55] Reiter, J. (2005). Estimating risks of identification disclosure for microdata. *Journal of the American Statistical Association*, 100:1103–1113.

[56] Rudin, W. (1987). *Real & Complex Analysis*. McGraw-Hill, 3$^{\text{rd}}$ edition.

[57] Samarati, P. (2001). Protecting respondents' identities in microdata release. *IEEE Transactions on Knowledge and Data Engineering*, 13(6):1010–1027.

[58] Samarati, P. and Sweeney, L. (1998). Protecting Privacy When Disclosing Information: k-Anonymity and its Enforcement through Generalization and Suppression. Technical report, Carnegie Mellon University, Software Research Institute.

[59] Schechter, E. (1997). *Handbook of Analysis and Its Foundations*. Orlando, FL: Academic Press.

[60] Skellam, J. G. (1946). The frequency distribution of the difference between two Poisson variates belonging to different populations. *Journal of the Royal Statistical Society*, 109(3):296.

[61] Sweeney, L. (2002). k-anonymity: A model for protecting privacy. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10(5):557–570.

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

[63] Willenborg, L. and de Waal, T. (1996). *Statistical Disclosure Control in Practice*. Springer-Verlag.

[64] — (2000). *Elements of Statistical Disclosure Control*. Springer.

[65] Winkler, W. E. (2004). Re-identification methods for masked microdata. In *Privacy in Statistical Databases*, vol. 3050 of *LNCS*. Springer. 216–230.

[66] Wong, R., Fu, A., Wang, K., and Pei, J. (2007). Minimality attack in privacy preserving data publishing. In *Proceedings of the 33rd International Conference on Very Large Data Bases (VLDB '07)*. VLDB Endowment. 543–554.

[67] Xiao, X., Tao, Y., and Koudas, N. (2010). Transparent anonymization: Thwarting adversaries who know the algorithm. *ACM Transactions on Database Systems (TODS)*, 35(2). Article 8.

[68] Zhang, L., Jajodia, S., and Brodsky, A. (2007). Information disclosure under realistic assumptions: Privacy versus optimality. In *Proceedings of the 14th ACM Conference on Computer and Communications Security (CCS '07)*. ACM Press. 573–583.