The Pennsylvania State University
The Graduate School
College of Engineering

A FIRST PRINCIPLE APPROACH TOWARD DATA PRIVACY AND UTILITY

A Dissertation in
Computer Science and Engineering
by
Bing-Rong Lin

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

May 2014

The dissertation of Bing-Rong Lin was reviewed and approved* by the following:

Daniel Kifer
Assistant Professor of Computer Science and Engineering
Dissertation Advisor, Chair of Committee

Adam D. Smith
Associate Professor of Computer Science and Engineering

Wang-Chien Lee
Associate Professor of Computer Science and Engineering

David J. Miller
Professor of Electrical Engineering

Lee Coraor
Director of Academic Affairs and Associate Professor of Computer Science and Engineering

*Signatures are on file in the Graduate School.

Abstract

Individual data are collected by government and online service providers such as social networks, search engines and shopping websites. The huge amount of data about users' activities and personal information are collected and analyzed to improve the quality of service or to serve as scientific research data. However, the individual's privacy may not be protected.

The protection of privacy relies on the privacy definitions. Privacy definitions act as a contract defines the behavior of algorithms on processing sensitive data and producing non-sensitive sanitized data. Most of privacy definitions are constructed from intuition and intuition alone leads us astray. Examples include the release of AOL search log and Netflix competition dataset. The released data were anonymized based on intuition but, only a few days later, the journalist and researchers figure out means to identify people in the dataset. One goal of this thesis is to provide a systematic approach to extract semantic guarantees of privacy definitions.

Most of privacy definitions are constructed from intuition. In most cases, it is not clear what these privacy definitions actually guarantee. To the best of our knowledge, we present the first general framework for extracting semantic guarantees from privacy definitions. The privacy guarantees we can extract are Bayesian in nature and deal with changes in an attacker's beliefs. The framework can be applied to privacy definitions or even to individual algorithms to identify the types of inferences they defend against. We illustrate the use of our framework with analyses of several definitions and algorithms for which we can derive previously unknown semantics.

The other goal of this thesis is to systematically study utility measures via the first principle approach. The privacy definitions constrain the behavior of privacy algorithms. The utility measures are designed to help to choose which privacy algorithm to be used to process the sensitive data. In statistical privacy, utility refers to two concepts: information preservation – how much statistical information is retained by a sanitizing algorithm, and usability – how (and with how much difficulty) does one extract this information to build statistical models, answer queries, etc. We analyze the information-preserving properties of utility measures with utility axioms and study how violations of an axiom can be fixed. We show that the average error of Bayesian decision makers forms the unique class of utility measures that satisfy all of the axioms. The axioms are agnostic to Bayesian concepts such as subjective probabilities and hence strengthen support for Bayesian views in privacy research. This result connects information preservation to aspect of usability. Utility is measured in Bayesian manner. Shouldn't Bayesian decision theory be a good choice to work with sanitized data? An application to the sorted histogram problem shows that our decision-theoretic post-processing algorithm can estimate the underlying sorted histogram with much greater accuracy than prior work.

Table of Contents

List of Figures

List of Tables

Acknowledgments

It would not have been possible to have this thesis without the support and guidance of people around me, to only some of whom will be mentioned here.

I would like to thank my adviser Dr. Daniel Kifer. I am incredibly fortunate to have him as my adviser. He is extremely smart, kind, understanding and helpful. His advice is always constructive. Without his guidance, I won't have a chance to finish this thesis. I would like to thank Dr. Wang-Chien Lee who supports me personally and academically at all times. Under his guidance, I learned many handful and necessary skills to thrive as a researcher. I would like to thank Dr. Adam Smith who gave wonderful classes and taught me a lot. In addition, his advise on my works is very insightful and has great impact on my thesis. I would like to thank Dr. David Miller who serves as a committee member. He is very kind and thoughtful and willing to sacrifice his time and schedule to accommodate my needs.

Labmates have contributed immensely to my personal and professional time. I would like to thank for their helpful discussion as well as their humor.

Finally, I would like to thank my family. My parents are always supportive and grant me 100% freedom to purse my dream ever since I was little. My sisters have given me their unequivocal support throughout. Lastly, I would like to thank my wife Hong-Ning Fang for her support and patience at all times.

Chapter 1

Introduction

With increased capabilities for data collection, organizations are struggling to find ways to share data with the public, researchers, and other organizations. While sharing aggregate and statistical views of the data can benefit the public, spur research, and improve business ties, sharing individual records (such as transaction data) can violate the privacy of individuals.

One goal of privacy research is to identify the kinds of views (e.g., possibly noisy aggregate or statistical data) that are safe to grant access to. In this setting, an organization first chooses a privacy definition, then finds an algorithm that satisfies the privacy definition. Finally, sensitive data are fed into the algorithm and the algorithm outputs non-sensitive sanitized data that are considered safe by the chosen privacy definition.

Thus, a privacy definition is a contract: satisfying the privacy definition results in a "protection" of "privacy." For many privacy definitions, this contract is very vague – it is not clear what information they protect. Do they offer protections that the organization is interested in? Do they offer additional protections that the organization is not interested in (meaning that the view that is generated will contain too much distortion)? One goal of this thesis is to build a framework to extract the exact semantics guarantees of privacy definitions.

Once we have a concise privacy definition, a dataset containing sensitive information must be sanitized by an algorithm satisfying the privacy definition. Many algorithms can satisfy the same privacy definitions but their utility differ. Utility measures are designed to measure how "useful" an algorithm is. It assigns a numerical score to each possible algorithm, which reflects the ability of the data sanitizer to preserve statistical information that is useful for a given application. The data owner then uses the chosen data sanitizer to perturb the data and then releases the resulting sanitized data. The other goal of this thesis is to study the utility measures.

1.1 Extract Semantics Guarantees of Privacy Definitions

In August, 2006, AOL wanted to release the search log intended for research purpose [1]. In order to protect its users' privacy, AOL needed to first anonymize the search log. It was intuitive that the user account name might be used to uncover their identities because users tended to create account name based on their names. The anonymization algorithm replaced the user account names with random numbers such that each user associated with a unique number. Then, the anonymized search log was released to the public. Not for long, a journalist uncovered the identity of a user [2]. This is because the query log itself contained identifying information, for example, names.

Another example [3] of failure to protect users' privacy happened two months after the AOL incident. Netflix shared users' rating with competitors of Netflix Prize [4]. Intuitively, users' rating over movies did not contain information to link to users' identity. It turned out that the users' rating over movies might contain identification information if one could also access IMDb's data which are open to the public. Two weeks after Netflix announcing the $1-million Netflix competition, Narayanan and Shmatikov submitted a draft on how to break anonymity [5].

Both AOL and Netflix incidents are examples that privacy definitions are too "vague" in a way too weak to protect users' privacy. Privacy definitions can also be too "vague" in a way too strong to provide unnecessary protections. In Chapter 4, we present, to the best of our knowledge, the first general framework for extracting semantic guarantees from privacy definitions. Instead of trying to answer very narrow questions such as "does a privacy definition or algorithm protect a specific type of information X?", our goal is to address the more general question "*what* are the types of information that the privacy definition or algorithm protects?". Thus, an organization can judge whether a privacy definition is too weak or too strong for its needs.

The guarantees extracted from our framework are applicable to computationally unbounded Bayesian attackers. We look at the least common denominator: what can be guaranteed no matter what sanitized output is generated by any algorithm satisfying the given privacy definition. We also show how to use our framework to relax privacy definitions in an attempt to remove any unnecessary protections.

We apply our framework to analyze the privacy protections of randomized response [6], FRAPP [7]/PRAM [8], and random sampling. We derive new semantic guarantees for them and show that they provide privacy protections that are often considered unnecessary (they protect various notions of parity of the input datasets).

1.2 Information Preserving Measures

Many algorithms can satisfy a given privacy definition, but their utility often differ. Some algorithms preserve statistical information better than others, and some produce outputs that are easier to use than others (as an extreme example, an algorithm that encrypts its input also perfectly preserves information, but this information is computationally difficult to access).

Some applications incentivize a separation between information preservation and usability. They treat measures of information preservation as optimization criteria in algorithm design [9, 10, 11, 12, 13, 14]. The resulting algorithm is used to generate sanitized data and the data users must process it to extract useful information.

One example is privacy-preserving data publishing, where the data owner plans a one-time publication of privacy-enhanced data. In this case, the data owner seeks to maximize the information content of the sanitized data (subject to privacy constraints). Then end-users, with the help of statistical experts, have to process this data to build models, answer queries, etc.

Another example occurs when users must pay for privacy-preserving query answers [15, 16]. In this case, the data user wants the data owner to run sanitizing algorithms that maximize information content subject to privacy and cost constraints.

Finally, theoretical [11] and empirical [17] results suggest that information maximization followed by various kinds of postprocessing allows sanitized data to be used effectively for multiple purposes.

Can any information preservation measures for a sanitizing algorithm be justified from first principles? We analyzed three utility axioms presented by [18] and derived the following result: those axioms imply that the ability of a data sanitizer $\mathfrak{M}$ to preserve information should be measured as the average (over possible outputs of $\mathfrak{M}$) error of a Bayesian decision maker. This result is important because it provides support for a rarely used utility measure (we are only aware of its use by Alvim et al. [12, 13]).

1.3 Processing Sanitized Data

Consider the scenario that the data are first sanitized and then released to the public. End-users need to process sanitized data to extract query answers and build predictive models while taking advantage of probabilistic knowledge and constraints that are known to hold (e.g., [11, 19, 17, 20, 21, 22, 23, 24, 25, 26]). Since the ability of a sanitizing algorithm to preserve information should (according to the axioms) be measured as the expected error of a Bayesian decision maker, it stands to reason that this Bayesian methodology should play a more prominent role in the analysis of sanitized data. We apply this insight to the sorted histogram problem (also known as the unattributed histogram) first studied by Hay et al. [21]. The goal is to reconstruct a sorted histogram from differentially private sanitized data. The resulting sorted histogram can be used to study edge distributions in social networks, identify power laws, etc. [21]. Using hidden Markov models, we develop a reconstruction algorithm for the sorted histogram problem based on Bayesian decision theory. We show experimentally that our technique outperforms both the least-squares approach of Hay et al. [21] and maximum likelihood estimation.

1.4 Outline

We introduce our view of privacy definitions and algorithms in Section 1.5. We discuss related work in Chapter 2, review privacy and utility axioms from [18] and define new utility axioms in Chapter 3. After that we present a framework to extract semantics guarantees from privacy definitions in Chapter 4. In Chapter 5, we present an axiomatically justifiable measure of utility. After that, we present our decision-theoretic algorithm for the sorted histogram problem in Chapter 6. Conclusions and open questions are presented in Chapter 7.

1.5 Views of Privacy Definitions and Algorithms

Throughout the thesis, we heavily depend on linear algebra. We introduce our views of privacy definitions and algorithms in the language of linear algebra.

1.5.1 Privacy Definitions as Sets of Algorithms

A Privacy Definition is often expressed as a set of algorithms that we trust (e.g., [6]), or a set of constraints on how an algorithm behaves (e.g., [27]), or on the type of output it produces (e.g., [28]). Note that treating a privacy definition as a set of algorithms is the more general approach – a privacy definition becomes a given set of algorithms, or the set of algorithms that satisfy a set of constraints, or a set of algorithms that produce certain types of outputs, etc. It also allows us to manipulate privacy definitions using set theory. Thus this is the approach we take in this thesis.

Formally, A privacy definition is a 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 $\mathfrak{Priv}$.

1.5.2 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 $\omega_j \in$ 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 1.5.2).

Definition 1.5.1 (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 $\mathfrak{M}$*

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

Figure 1.1. The matrix representation of $\mathfrak{M}$. Columns are indexed by datasets $\in$ domain($\mathfrak{M}$) and rows are indexed by outputs $\in$ range($\mathfrak{M}$).

is a (potentially infinite) matrix whose columns are indexed by $\mathbb{I}$*, rows are indexed by* range($\mathfrak{M}$)*. The value of each entry* (i, j) *is the quantity* $P(\mathfrak{M}(D_j) = \omega_i)$*.*

Chapter 2

Related Work

2.1 Evaluating Privacy

Research in statistical privacy mainly focuses on developing privacy definitions and algorithms for publishing sanitized data (i.e., nonsensitive information) derived from sensitive data. To the best of our knowledge, this thesis provides the first framework for extracting semantic guarantees from privacy definitions. Other work on evaluating privacy definitions looks for the presence or absence of specific vulnerabilities in privacy definitions or sanitized data.

In the official statistics community, re-identification experiments are performed to assess whether individuals can be identified from sanitized data records [29]. In many such experiments, software is used to link sanitized data records to the original records [30]. Reiter [31] provides a detailed example of how to apply the decision-theoretic framework of Duncan and Lambert [32] to measure disclosure risk. There are many other methods for assessing privacy for the purposes of official statistics; for surveys, see [29, 33, 34].

Other work in statistical privacy seeks to identify and exploit specific types of weaknesses that may be present in privacy definitions. Dwork and Naor [35] 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 [36] suggests that harm can occur even 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 preventing an attacker from linking an individual to an "anonymized" record [37], possibly using publicly available data [38] or other knowledge [39]. Dinur and Nissim [40] and later Dwork et al. [41] showed fundamental limits to the amount of information that can be released even under very weak privacy definitions (information-theoretically and computationally [42]). These attacks generally work by removing noise that was added in the sanitization process [43, 44, 45]. Ganta et al. [46] 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 [47] 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 [48] 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 [49]. Note also that privacy concerns can also be associated with aggregate information such as trade secrets (and not just rows in a table) [50].

2.2 Privacy Definitions

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

2.2.1 Syntactic Privacy Definitions

A large class of privacy definitions places restrictions on the format of the output of a randomized algorithm. Such privacy definitions are known as *syntactic privacy definition*. The prototypical syntactic privacy definition is k-anonymity [28, 38].

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 [28, 38]. For example, consider the input table in Figure 2.1(a) where the attributes {"nationality", "age", "zip code"} have been designated as the quasi-identifier. Given the input table from Figure 2.1(a), an algorithm $\mathfrak{M}$ that satisfies 3-anonymity could output the 3-anonymous table in Figure 2.1(b). Algorithms satisfying k-anonymity typically work by generalizing (coarsening) attribute values. For example, the age attribute may be generalized into an age range of size 10 (e.g., $[0-9], [10-19]$, etc.) or ranges of size 20. Quasi-identifier attributes are repeatedly generalized 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.

Many variants of k-anonymity exist to handle different applications (such as social networks [51]) or to address vulnerabilities such as the need to protect sensitive attributes [39, 52]. For further details, see [29, 53]. We will examine these approaches in Section 4.2.3.2.

Zip Code	Age	Nationality	Disease
13053	25	Indian	Cold
13068	39	Russian	Stroke
13053	27	American	Flu
14850	43	American	Cancer
14850	57	Russian	Cancer
14853	40	Indian	Cancer

(a) Original Table

Zip Code	Age	Nationality	Disease
130**	< 40	*	Cold
130**	< 40	*	Stroke
130**	< 40	*	Flu
1485*	≥ 40	*	Cancer
1485*	≥ 40	*	Cancer
1485*	≥ 40	*	Cancer

(b) 3-Anonymous Table

Figure 2.1. Example of k-Anonymity

2.2.2 Randomized Response

Randomized response is a technique developed by Warner [6] 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 4.3.1.

2.2.3 PRAM and FRAPP

PRAM [8] and FRAPP [7] 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 $a \in \mathcal{TUP}$ is 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. It is important to note that the matrix representation of $\mathfrak{M}_Q$ (as discussed in Section 1.5.2) *is not the same* as the transition matrix Q. As we will discuss in Section 4.3.2, the relationship between the two is that the matrix representation of $\mathfrak{M}_Q$ is equal to $\bigoplus_n Q$, where $\oplus$ is the Kronecker product.

FRAPP, with privacy parameter γ, imposes a restriction on these algorithms. This restriction, known as γ-amplification [54], 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 $\frac{P(b \to a)}{P(c \to a)} \leq \gamma$.

2.2.4 Differential Privacy

Differential privacy [27, 55] is defined as follows:

Definition 2.2.1. *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). Differential privacy is special in the sense that the constraints on probabilities in Definition 2.2.1 are exactly the same as the constraints that define its row cone. We discuss this in more detail in Section 4.2.3.1.

This work is an extension of our conference paper [56] which includes analysis of two new utility axioms (quasi-convexity/concavity), analysis of an additional utility measure (posterior sampling utility in Section 5.2.3) and the sufficing procedure (Section 5.4) that fixes a utility measure's violation of the sufficiency axiom.

2.3 Utility Axioms for Statistical Privacy

Kifer and Lin [57, 18] proposed three axioms for measures of information preservation for sanitizing algorithms to aid in the scientific analysis of utility in statistical privacy. We build on this foundation with two new axioms, systematic analyses of many common utility measures, new analyses of the axioms that show their connections to Bayesian theory, a new post-processing algorithm for the unattributed histogram that is based on these connections, and new results showing how to fix violations of the fundamental sufficiency axiom.

2.4 Bayesian Decision Theory

Axiomatic justifications for Bayesian decision theory are well-studied in statistics and economics. There are differences between our results and those from statistics and economics (e.g., [58, 59, 60, 61]). The goals differ (i.e., how to assign a number to an algorithm vs. how a user should make decisions in the face of uncertainty) and the subjects of the axioms differ (i.e., an algorithm vs. a data user's preference over actions after seeing sanitized data). In a sense, our results extend prior work in that the axioms about algorithms imply the existence of actions (that an end-user must choose from after seeing sanitized data), user preferences over actions, priors, and behaviors (i.e., Bayesian updating) that are treated in other work (each paper infers the existence of some but not all of these components).

The mathematical existence of subjective prior probabilities was derived by de Finetti through the concepts of exchangeability and the Dutch Book argument [58]. Given *objective* probabilities and a choices for actions, von Neumann and Morgenstern [59] used axioms to show that a decision maker should take an action that maximizes expected utility. Savage [60] extended this result

to show that a rational decision maker is mathematically equivalent to an individual who has *subjective* probabilities and chooses actions to maximize expected subjective utility. Myerson [61] proposed a set of axioms whose consequences are as follows: a user makes decisions by choosing a subjective prior, forms the posterior distribution upon seeing new evidence, and then takes an action to maximize expected utility (using the posterior to compute the expectation). The language of those axioms talks about prizes and lotteries. Prize X corresponds to an action (i.e. using X as a query answer or a model parameter) and lotteries refer to the possibility that a user may want to choose an action randomly instead of deterministically.

2.5 Utility Measures

In the literature, the word *utility* is very closely associated with concepts we refer to as *usability*. That is, proposed techniques often follow the principle that a user should not have to treat sanitized data any differently from ordinary data (although, for power users, it is important to account for the additional uncertainty in sanitized data).

Early work on privacy considered utility to be a property of the sanitized data rather than a property of the data sanitizer. As a result, utility was often measured as some notion of "closeness" between the original data and sanitized data [29] or "closeness" of a statistical analysis conducted on both datasets (e.g., [62, 63], which do not use detailed knowledge of how the data sanitizer works). Choosing data sanitizers according to such metrics often leads to privacy vulnerabilities such as minimality attacks [47].

In other cases, utility was recognized as a property of the data sanitizer but often was only defined for a restricted class of data sanitizers. For example, if we only consider data sanitizers that coarsen tuple attributes, one can measure utility based on the number of coarsening operations (e.g., [64]) or if we only consider data sanitizers that add noise to query answers, one can measure variance of the noise or probability that the noise exceeds a certain bound (see [65, 40, 21, 66, 42] for variations).

One of the most fundamental results about utility for data sanitizers is due to Ghosh et al. [11]. They considered scenarios having the following four characteristics: (1) the data owner wants to answer a single counting query in a private manner; (2) each end-user has a prior distribution over possible true query answers; (3) each end-user has a personal similarity measure between possible true query answers and possible sanitized query answers; (4) each end-user wants the data owner to choose a data sanitizer that maximizes the end-user's expected similarity between true and sanitized query answers. Ghosh et al. showed that in many cases, the data owner should first achieve privacy using a data sanitizer called the *Geometric mechanism* [11] and then customize the result for each end-user by performing a lossy postprocessing that will maximize the end-user's personal utility measure (to improve usability). Later, Gupte and Sundararajan [19] extended this result to a class of minimax utility measures. Kifer and Lin [57] then investigated useful properties for measures of information preservation. They proposed several axioms and measures [57, 18] that avoid problems illustrated in Example 5.1.1. However, it is not clear

whether the measures proposed in [57, 18] were meaningful for practical applications.

2.6 Using Sanitized Data

Once sanitized data has been produced, a data analyst often wishes to estimate some quantity – a query answer, a model parameter, etc. The research community has recognized that to build a reliable estimator, it is often necessary to process sanitized data in special ways – to account for constraints that must hold (e.g., [20, 21, 67]), to incorporate knowledge of how the data sanitizer works (e.g., [17, 10, 68, 69]), and to account for specialized output formats that can be used to represent uncertainty (e.g., [24, 70, 71, 26]). Bayesian methods (and Bayesian decision theory in particular) are not often used. This thesis provides a formal link between measures of information preservation and tools for working with sanitized data via this statistical methodology.

Chapter 3

Privacy and Utility Axioms

3.1 Review of Privacy Axioms

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 [28]. Such a set is often incomplete in the sense that it might not include all of the algorithms we should trust if we are prepared to accept $\mathfrak{Priv}$. 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, if we decide to trust algorithms satisfying k-anonymity to protect privacy, we should also trust the algorithm $\mathfrak{M}$. Indeed, it would be strange if releasing a k-anonymous table were acceptable but releasing a model built solely from that table (without any side information) were not acceptable. Therefore we should enlarge the set $\mathfrak{Priv}$ by adding $\mathfrak{M}$ into this set.

Having motivated the need for expanding the set of algorithms that defines a privacy definition, we review two axioms from [18] that allow us to enlarge a privacy definition without weakening it.

Axiom 3.1.1 (Post-processing [18])**.** *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 $\mathfrak{Priv}$.*[1]

Note that Axiom 3.1.1 prevents algorithm $\mathcal{A}$ from using side information since its only input is $\mathfrak{M}(D)$.

Axiom 3.1.2 (Convexity [18])**.** *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*

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

$\text{choice}^{p}_{\mathfrak{M}_1,\mathfrak{M}_2}$ *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 for the convexity axiom (Axiom 3.1.2) 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 [18].

3.2 Review of Utility Axioms

We review existing utility axioms in Section 3.2 and propose new axioms in Section 3.3. In this section we review three axioms of information preservation that were previously proposed in [57, 18]. These axioms will be used in new analyses of utility measures in Sections 5.2 and 5.3.

Axiom 3.2.1. (Sufficiency [18]). *An information preservation measure* $\mu_{\mathbb{I}}$ *(resp., loss measure* $\mathcal{L}_{\mathbb{I}}$*) should satisfy the relation* $\mu_{\mathbb{I}}(\mathfrak{M}_1) \geq \mu_{\mathbb{I}}(\mathfrak{M}_2)$ *(resp.,* $\mathcal{L}_{\mathbb{I}}(\mathfrak{M}_1) \leq \mathcal{L}_{\mathbb{I}}(\mathfrak{M}_2)$*) whenever* $\mathfrak{M}_2 = \mathcal{A} \circ \mathfrak{M}_1$ *for some (possibly randomized) algorithm* $\mathcal{A}$ *whose domain contains the range of* $\mathfrak{M}_1$.

The *axiom of sufficiency* essentially states that postprocessing by an algorithm $\mathcal{A}$ cannot increase the amount of information that is preserved. It is a fundamental axiom for any information measure.

However, note that postprocessing may improve *usability*. For example, $\mathcal{A}$ may take some of the information contained in the output of $\mathfrak{M}_1$ and put it into a usable form, such as a model parameter or query answer. As we show in Chapter 5.3, Bayesian decision theory plays a key role in usability and leads to a new processing algorithm for the sorted histogram problem (Chapter 7).

Another useful axiom is continuity [18]. It states that small changes in the probabilistic[2] behavior of $\mathfrak{M}$ result in small changes to the amount of information preserved. We rephrase it here using the following metric over the collection of data sanitizers with the same input domain.

Definition 3.2.2. (Metric over data sanitizers). *Define the metric* $\mathrm{d}^{*}_{\mathbb{I}}$ *over the set of data sanitizers with the same input domain* $\mathbb{I} = \{D_1, D_2, \dots\}$ *as follows:*

$$\mathrm{d}^{*}_{\mathbb{I}}(\mathfrak{M}_1, \mathfrak{M}_2) = \sup_{D \in \mathbb{I}} \sum_{\omega} \Big| P[\mathfrak{M}_1(D) = \omega] - P[\mathfrak{M}_2(D) = \omega] \Big|$$

where the summation is over $\omega \in \text{range}(\mathfrak{M}_1) \cup \text{range}(\mathfrak{M}_2)$.

Axiom 3.2.3. (Continuity [18]). *An information preservation measure* $\mu_{\mathbb{I}}$ *(resp., loss measure* $\mathcal{L}_{\mathbb{I}}$*) should be continuous with respect to the metric* $\mathrm{d}^{*}_{\mathbb{I}}$.

[2] Note $P(\mathfrak{M}(D_i) = \omega_j)$ is also defined for deterministic $\mathfrak{M}$.

The next axiom is called *branching* [18]. It quantifies how much information we expect to lose if we are only told that $\mathfrak{M}$ has output either ω_1 or ω_2 (but we don't know which one). The branching axiom states that this decrease depends on the probabilities with which ω_1 and ω_2 could be generated but does not depend on events that did not occur (i.e., it does not depend on $\omega_3, \omega_4, \dots$).

Axiom 3.2.4. (Branching [18]). *An information preservation measure $\mu_{\mathbb{I}}$ should satisfy the relation*

$$\mu_{\mathbb{I}}(\mathfrak{M}) = \mu_{\mathbb{I}}\left(\widetilde{\mathfrak{M}}\right) + G\left(\vec{P}[\mathfrak{M}(\cdot) = \omega_1],\ \vec{P}[\mathfrak{M}(\cdot) = \omega_2]\right)$$

for some function G, where

- *ω_1 and ω_2 are the first two elements in* $\mathrm{range}(\mathfrak{M})$ *(any arbitrary ordering will suffice).*
- *$\widetilde{\mathfrak{M}}$ is an algorithm with* $\mathrm{range}(\widetilde{\mathfrak{M}}) = \{\omega^*\} \cup \mathrm{range}(\mathfrak{M})$ *and which behaves exactly like $\mathfrak{M}$ except that $\widetilde{\mathfrak{M}}$ outputs ω^* whenever $\mathfrak{M}$ would have output ω_1 or ω_2.*
- *$\vec{P}[\mathfrak{M}(\cdot) = \omega_i]$ denotes the vector $\langle P[\mathfrak{M}(D_1) = \omega_i], P[\mathfrak{M}(D_2) = \omega_i], \dots \rangle$ corresponding to probabilities of generating ω_i.*

A loss measure $\mathcal{L}_{\mathbb{I}}$ should satisfy the same type of relation.

3.3 New Utility Axioms - Axioms of Quasiconvexity and Quasiconcavity

In this section we present two axioms concerning the utility of randomly choosing a data sanitizer. We first need the following definition.

Definition 3.3.1. (Operator $\oplus_p$). *For any two algorithms $\mathfrak{M}_1, \mathfrak{M}_2$ with the same input domain, let $\mathfrak{M}_1 \oplus_p \mathfrak{M}_2$ be the algorithm that runs $\mathfrak{M}_1$ with probability p and $\mathfrak{M}_2$ with probability $1-p$* ***and reveals which algorithm was run.***

Axiom 3.3.2. (Quasi-convexity of information). *Information preservation measures $\mu_{\mathbb{I}}$ and loss measures $\mathcal{L}_{\mathbb{I}}$ should satisfy the relations $\mu_{\mathbb{I}}(\mathfrak{M}_1 \oplus_p \mathfrak{M}_2) \leq \max\{\mu_{\mathbb{I}}(\mathfrak{M}_1), \mu_{\mathbb{I}}(\mathfrak{M}_2)\}$ and $\mathcal{L}_{\mathbb{I}}(\mathfrak{M}_1 \oplus_p \mathfrak{M}_2) \geq \min\{\mathcal{L}_{\mathbb{I}}(\mathfrak{M}_1), \mathcal{L}_{\mathbb{I}}(\mathfrak{M}_2)\}$ for all $\mathfrak{M}_1, \mathfrak{M}_2$ and $p \in [0,1]$.*

Axiom 3.3.3. (Quasi-concavity of information). *$\mu_{\mathbb{I}}$ and $\mathcal{L}_{\mathbb{I}}$ should satisfy $\mu_{\mathbb{I}}(\mathfrak{M}_1 \oplus_p \mathfrak{M}_2) \geq \min\{\mu_{\mathbb{I}}(\mathfrak{M}_1), \mu_{\mathbb{I}}(\mathfrak{M}_2)\}$ and $\mathcal{L}_{\mathbb{I}}(\mathfrak{M}_1 \oplus_p \mathfrak{M}_2) \leq \max\{\mathcal{L}_{\mathbb{I}}(\mathfrak{M}_1), \mathcal{L}_{\mathbb{I}}(\mathfrak{M}_2)\}$ for all $\mathfrak{M}_1, \mathfrak{M}_2$ and $p \in [0,1]$.*

Together, the two axioms state that the amount information preserved by the algorithm $\mathfrak{M}_3 \stackrel{\text{def}}{=} \mathfrak{M}_1 \oplus_p \mathfrak{M}_2$ is somewhere in between the information preserved by $\mathfrak{M}_1$ and $\mathfrak{M}_2$. That is, if we prefer $\mathfrak{M}_2$ over $\mathfrak{M}_1$, then $\mathfrak{M}_3$ is not better than $\mathfrak{M}_2$ (because sometimes it is the same as running the less preferred $\mathfrak{M}_1$), but it is not worse than $\mathfrak{M}_1$ (because sometimes it is the same as running the more preferred $\mathfrak{M}_2$).

Chapter 4

Systematic Analysis of Privacy Definitions

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 [29, 18]. 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., [38, 5, 2]) 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., [48, 52, 39, 38, 45, 47, 31, 46, 72, 73, 49, 74]) 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 [42, 40, 75, 76, 41, 77] 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 [35, 78, 79, 80, 81, 82, 66, 54, 83, 84, 85]

Of particular interest are Bayesian approaches such as [81, 82, 54, 80, 79, 85] that consider the safety of different types of sensitive information with respect to a wide variety of attackers. In this Chapter, 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?

Our framework works by first restating a privacy definition in the universal language of set theory, which provides a simple and convenient way to express any privacy definition. The next step is to find the *consistent normal form* of a privacy definition. This is a normalization step that removes some implicit assumptions. For example, many privacy definitions do not explicitly state that building a histogram from sanitized data and then releasing the histogram instead of the sanitized data is acceptable. However, the designers of privacy definitions often implicitly assume that it is indeed acceptable. From this consistent normal form we then extract a geometric structure called a row cone. Intuitively, the row cone captures all the ways in which an attacker's prior belief can be turned into a posterior belief after observing an output from an algorithm that satisfies the given privacy definition. The row cone can be defined as the solution set to a system of linear inequalities. We extract semantic guarantees by re-interpreting the coefficients of the linear inequalities as probabilities and the linear inequalities themselves as statements about probabilities.

Our contributions are:

- A novel framework that introduces the concepts of consistent normal form and row cone of a privacy definition and uses them for the purposes of understanding privacy definitions, extracting their semantic guarantees, and relaxing privacy definitions.
- Several applications of our framework, from which we extract previously unknown semantic guarantees for randomized response, FRAPP/PRAM, and random sampling.
- The framework provides guidelines for the design of privacy definitions (which influenced [79]) 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 chapter is organized as follows. We provide a detailed overview of our approach in Section 4.1. Using ideas based on an axiomatic foundation of privacy [57, 18], we show how to remove some implicit assumptions from privacy definitions in Section 4.2. Using the resulting privacy definition, we formally define the row cone (a fundamental geometric object we use for extracting semantic guarantees) in Section 4.2.2. We then proceed to discuss the relationship between our framework and the folklore concerning differential privacy [27] and syntactic privacy definitions such as k-anonymity [28] in Section 4.2.3. Finally, in Section 4.3, we apply our framework to extract new semantic guarantees for randomized response (Section 4.3.1), FRAPP/PRAM (Section 4.3.2), and random sampling (Section 4.3.3).

4.1 The Bird's-Eye View

We first present some basic concepts in Section 4.1.1 and then provide a high-level overview of our consistency methodology in Section 4.1.2.

4.1.1 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 in our methodology.

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 then 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. The attacker may also have side information – in the literature this is often expressed in terms of a prior distribution over possible datasets $D_i \in \mathbb{I}$. In this chapter, we are mostly interested in guarantees against attackers who reason probabilistically and so we also assume that an attacker's side information is encapsulated in a prior distribution.

The data curator will choose a privacy definition based on what it can guarantee about the privacy of sensitive information. If a privacy definition offer 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. In this thesis, when we discuss semantic guarantees, we are interested in the least common denominator: what does the privacy definition guarantee regardless of what sanitized output is produced by an algorithm satisfying that privacy definition. Another way to think about this is: given a computationally unbounded Bayesian attacker, what properties of the true dataset are being protected and what is the minimum level of uncertainty that the attacker will be left with. 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 [35, 78].

4.1.2 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. The steps of our framework then require us 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.

4.1.2.1 Consistent Normal Form of Privacy Definitions.

Recall from Section 1.5 that we take the unifying view that a privacy definition is a set of algorithms – those algorithms that are trusted to produce sanitized data for a given application, those algorithms that satisfy certain constraints, etc.

Not surprisingly, there are many sets of algorithms that do not meet common expectations of what a privacy definition is [18]. 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 [18]. 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 $\mathcal{A} \circ \mathfrak{M}$ should be trusted.

Thus, given a privacy definition $\mathfrak{Priv}$, we first must normalize by explicitly adding into it all of the algorithms we should trust. This normalization of a privacy definition $\mathfrak{Priv}$ is called the *consistent normal form* and denoted by $\mathrm{CNF}(\mathfrak{Priv})$. $\mathrm{CNF}\,\mathfrak{Priv}$ is again a set of algorithms and $\mathfrak{Priv} \subseteq \mathrm{CNF}(\mathfrak{Priv})$. We describe it in full technical detail in Section 4.2.1.

4.1.2.2 The Row Cone

Recall that we represent algorithms as matrices (Definition 1.5.1) and privacy definitions as sets of algorithms. Therefore $\mathrm{CNF}(\mathfrak{Priv})$, the set of algorithms we should trust if we accept the privacy definition $\mathfrak{Priv}$, is really a *set of matrices*. The row cone of $\mathfrak{Priv}$, denoted by $\mathrm{rowcone}(\mathfrak{Priv})$, is the set of vectors of the form $c\vec{x}$ where $c \geq 0$ and $\vec{x}$ is a row of one of those matrices (each matrix representing some algorithm $\mathfrak{M} \in \mathrm{CNF}(\mathfrak{Priv})$).

How does the row cone capture the semantics of $\mathfrak{Priv}$? Suppose $\mathfrak{M} \in \mathrm{CNF}(\mathfrak{Priv})$ is one of the algorithms that we trust. Let D be the true input dataset and let $\omega \equiv \mathfrak{M}(D)$ be the sanitized output that we publish. A Bayesian attacker who sees output ω and is trying to derive sensitive information will need to compute the posterior distribution $P(\mathrm{data} = D_i \mid \mathfrak{M}(\mathrm{data}) = \omega)$ for all datasets D_i. This posterior distribution is a function of the attacker's prior $P(\mathrm{data} = D_i)$ and

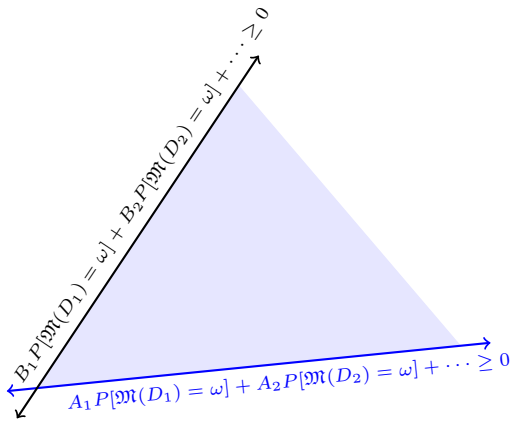

Figure 4.1. An example of a row cone (shaded) and its defining linear inequalities.

the vector of probabilities:

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

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 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).

Thus constraints satisfied by the row cone are also constraints on how prior probabilities could be turned into posterior probabilities. In Figure 4.1.2.2 we illustrate a row cone in 2 dimensions (i.e. the input domain consists of only 2 datasets). Each vector in the row cone is represented as a point in 2-d space. It turns out that the row cone is always a convex set and hence can be expressed as the intersection of halfspaces or, equivalently, as a solution to a system of linear inequalities as shown in Figure 4.1.2.2.

4.1.2.3 Extracting Semantic Guarantees From the Row Cone

The row cone is a convex set (in fact, a convex cone) and therefore can be expressed as the solution of a set of linear inequalities of the form [86]:

$$A_1 P(\mathfrak{M}(D_1) = \omega) + A_2 P(\mathfrak{M}(D_2) = \omega) + \cdots \geq 0$$

that must hold for all trusted algorithms $\mathfrak{M} \in$ CNF($\mathfrak{Priv}$) and sanitized outputs $\omega \in$ range($\mathfrak{M}$) they can produce. The key insight is that we can re-interpret the magnitude of the coefficients $|A_1|, |A_2|, \dots$ of these linear inequalities as probabilities (dividing by $|A_1| + |A_2| + \dots$ if necessary) and then re-interpret the linear inequalities as statements about prior and posterior probabilities

of an attacker. We give a detailed example in Section 4.3.1, where we apply our framework 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 [35, 78]).

4.2 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 4.3 to extract novel semantic guarantees for randomized response, FRAPP/PRAM, and random sampling. As a warmup to those applications, we include Section 4.2.3 to provide some simple introductory illustrations of our framework – we compute CNF($\mathfrak{Priv}$) and rowcone($\mathfrak{Priv}$) for differential privacy and for syntactic anonymization methods to re-derive some semantic guarantees that are known in the folklore.

4.2.1 The Consistent Normal Form

Using Axiom 3.1.1 and 3.1.2, we define the *consistent normal form* as follows:

Definition 4.2.1. (CNF). *Given a privacy definition $\mathfrak{Priv}$, its consistent normal form, denoted by* CNF($\mathfrak{Priv}$)*, is the smallest set of algorithms that contains $\mathfrak{Priv}$ and satisfies Axioms 3.1.1 and 3.1.2.*

If we accept Axioms 3.1.1 and 3.1.2, then the consistent normal form CNF($\mathfrak{Priv}$) has exactly the same privacy properties as $\mathfrak{Priv}$; that is, if we are prepared to trust any $\mathfrak{M} \in \mathfrak{Priv}$ we should also trust any $\mathfrak{M} \in$ CNF($\mathfrak{Priv}$). 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. Essentially, the consistent normal form uses Axioms 3.1.1 and 3.1.2 to turn implicit assumptions about which algorithms we trust into explicit statements.

The following theorem shows how to derive CNF($\mathfrak{Priv}$).

Theorem 4.2.2. *Given a privacy definition $\mathfrak{Priv}$, its consistent normal form* CNF($\mathfrak{Priv}$) *can be derived from the following process.*

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 $\text{CNF}(\mathfrak{Priv}) = \mathfrak{Priv}^{(2)}$.

Proof. See Section 4.5.1. □

Corollary 4.2.3. *If* $\mathfrak{Priv} = \{\mathfrak{M}\}$ *consists of just one algorithm,* $\text{CNF}(\mathfrak{Priv})$ *is the set of all algorithms of the form* $\mathcal{A} \circ \mathfrak{M}$*, where* $\text{range}(\mathfrak{M}) \subseteq \text{domain}(\mathcal{A})$ *and the random bits in* $\mathcal{A}$ *and* $\mathfrak{M}$ *are independent of each other.*

Proof. See Section 4.5.2. □

4.2.2 The Row Cone

Having motivated the row cone in Section 4.1.2.2, we now formally define it and derive its basic properties.

Definition 4.2.4 (Row Cone)**.** *Let* $\mathbb{I} = \{D_1, D_2, \dots\}$ *be the set of possible input datasets and let* $\mathfrak{Priv}$ *be a privacy definition. The* row cone *of* $\mathfrak{Priv}$*, denoted by* $\text{rowcone}(\mathfrak{Priv})$*, is defined as the set of vectors:*

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

Recalling the matrix representation of algorithms (as discussed in Section 1.5.2 and Figure 1.5.2), 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 $\text{rowcone}(\mathfrak{Priv})$ 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 4.2.5. $\text{rowcone}(\mathfrak{Priv})$ *is a convex cone.*

Proof. See Section 4.5.3. □

The fact that the row cone is a convex set means that it is expressible as the solution of a set of linear inequalities, as we previously saw in Figure 4.1.2.2. These linear inequalities can be interpreted as statements about possible joint distributions of input data and sanitized outputs. From those statements we can extract information about the relationships between possible prior and posterior distributions of the attacker. We present simple examples, in the way of motivation, in Section 4.2.3 before applying applying our framework to more complicated privacy definitions in Section 4.3.

4.2.3 Simple Examples from Folklore

In this section, as a warmup, we relate CNF($\mathfrak{Priv}$) and rowcone($\mathfrak{Priv}$) to known semantic guarantees from folklore.

4.2.3.1 Differential Privacy

Let ϵ-$\mathfrak{diffpriv}$ denote the set of algorithms satisfying ϵ-differential privacy (Definition 2.2.1). It is easy to see that ϵ-$\mathfrak{diffpriv}$ satisfies Axioms 3.1.1 and 3.1.2 and so it is already in consistent normal form: ϵ-$\mathfrak{diffpriv}$ = CNF(ϵ-$\mathfrak{diffpriv}$).

Furthermore, rowcone(ϵ-$\mathfrak{diffpriv}$) can be easily extracted. The vector $\vec{x} = (x_1, x_2, \dots) \in$ rowcone(ϵ-$\mathfrak{diffpriv}$) if and only if $x_i \leq e^\epsilon x_j$ whenever D_i and D_j differ in the value of one tuple. Alternatively, with $c > 0$, the vector

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

belongs to rowcone(ϵ-$\mathfrak{diffpriv}$) if and only if the linear inequality $cP(\mathfrak{M}(D_i) = \omega) \leq e^\epsilon cP(\mathfrak{M}(D_j) = \omega)$ is satisfied for all pairs of datasets D_i, D_j that differ in the value of one tuple.

Here is how these linear inequalities translate into semantic guarantees. A simple, well-known computation shows:

$$\begin{aligned} & P(\mathfrak{M}(D_i) = \omega) \leq e^\epsilon P(\mathfrak{M}(D_j) = \omega) \\ \Leftrightarrow \quad & \tfrac{P(\text{data}=D_i)P(\mathfrak{M}(D_i)=\omega)}{P(\text{data}=D_j)P(\mathfrak{M}(D_j)=\omega)} \leq e^\epsilon \tfrac{P(\text{data}=D_i)}{P(\text{data}=D_j)} \end{aligned} \tag{4.1}$$

According to folklore, Equation 4.1 is interpreted in terms of prior odds and posterior odds: if the attacker believes that table D_i (e.g., the table where Bob has cancer) is α times as likely as D_j (e.g., the table where Bob does not have cancer and all else is the same), then after seeing the sanitized output ω, the attacker will believe that D_i is only at most $e^\epsilon\alpha$ times as likely as D_j. We emphasize that these semantics of differential privacy are well known; we included this example because it helps illustrate the concepts of CNF($\mathfrak{Priv}$) and rowcone($\mathfrak{Priv}$), whose definition and use are contributions of this thesis.

4.2.3.2 Syntactic Methods

Syntactic privacy definitions are those that place restrictions on the format of the output that an algorithm is allowed to produce. As discussed in Section 2.2.1, k-anonymity [28] is a prototype of such privacy definitions. The original version of k-anonymity did not place any restrictions on the types of generalizations (coarsening) that can be performed on the input data. In the folklore, it is well-known that a k-anonymous algorithm can encode its entire input as a k-anonymous table. As a result, its consistent normal form is easy to compute.

Theorem 4.2.6. *Given a fixed schema with a quasi-identifier that contains an integer-valued attribute, the consistent normal form of k-anonymity consists of every algorithm whose input*

domain contains tables with this schema. The row cone consists of all vectors.

Proof. See Section 4.5.4. □

The essence of Theorem 4.2.6 is that without additional restrictions, k-anonymity cannot prevent a malicious anonymization algorithm from uniquely encoding its input into the format of a k-anonymous table that can be efficiently decoded to retrieve the input. Since the privacy definition cannot prevent such behavior, no worst-case semantic guarantees exist.

With restrictions on how the data is coarsened and on how the anonymization algorithms behave [87, 88], it is possible to exclude such malicious algorithms whose outputs uniquely determine their inputs. However, such restrictions do not necessarily produce privacy definitions that prevent *side-channel attacks* in which an algorithm uses the output format to encode some sensitive information about the input. Examples of side-channel attacks include: the minimality attack [47, 72] in which algorithms are forced to minimize a utility metric and end up accidentally leaking sensitive information[1]; an algorithm that outputs the table in Figure 2.1(b) (see Section 2.2.1) if the input is the table from Figure 2.1(a) and suppresses all attributes otherwise; an algorithm that suppresses the Age attribute only if Bob does not have cancer (hence unsuppressed Age values imply Bob has cancer).

For these reasons, we believe that when new syntactic privacy definitions are proposed, they should be accompanied by their row cones so that deficiencies such as possibilities of side-channel attacks can be evaluated.

4.2.3.3 Partitioning in Lieu of Syntactic Restrictions

Partitioning mechanisms such as [89] are alternatives to syntactic methods. Given a partitioning $\mathbb{P}$ of the input domain $\mathbb{I}$, let $\mathfrak{M}_{\mathbb{P}}$ be the algorithm that, on input D, returns the id of the partition containing D. Setting $\mathfrak{Priv} = \{\mathfrak{M}_{\mathbb{P}}\}$, then $\text{CNF}(\mathfrak{Priv}) \equiv \text{CNF}(\{\mathfrak{M}_{\mathbb{P}}\})$, the set of algorithms we should trust, is the set of algorithms that satisfy:

Definition 4.2.7 ($\mathbb{P}$-Partition Privacy)**.** *Given a partitioning $\mathbb{P}$ of the input space $\mathbb{I}$, a mechanism $\mathfrak{M}$ satisfies $\mathbb{P}$-partition privacy if $P[\mathfrak{M}(D_i) = \omega] = P[\mathfrak{M}(D_j) = \omega]$ whenever datasets D_i and D_j belong to the same partition in $\mathbb{P}$.*

As with differential privacy, the row cone can be easily read off of the definition: $\vec{x} = (x_1, x_2, \dots) \in \text{rowcone}(\{\mathfrak{M}_{\mathbb{P}}\})$ if and only if $x_i = x_j$ whenever D_i and D_j are in the same partition. Alternatively, for $c > 0$, the vector $(cP[\mathfrak{M}(D_1) = \omega],\ cP[\mathfrak{M}(D_2) = \omega],\ \dots) \in \text{rowcone}(\{\mathfrak{M}_{\mathbb{P}}\})$ if and only if $cP(\mathfrak{M}(D_i) = p) = cP(\mathfrak{M}(D_j) = \omega)$ for D_i and D_j in the same partition. The Bayesian guarantees are obvious: if D_i was believed to be α times as likely as D_j before seeing the sanitized output, then it is still α times as likely after seeing the sanitized output as long as D_i and D_j are in the same partition.

One may weaken the privacy definition by allowing a choice between different partitionings $\mathbb{P}_1, \mathbb{P}_2, \dots, \mathbb{P}_k$. The trusted set of algorithms would become $\text{CNF}(\{\mathfrak{M}_{\mathbb{P}_1}, \mathfrak{M}_{\mathbb{P}_2}, \dots, \mathfrak{M}_{\mathbb{P}_k}\})$. The

[1] The restrictions studied by [87, 88] were designed to thwart such attacks.

semantic guarantees then heavily depend on the different ways these partitions intersect each other.

4.3 Applications

In this section we present some of the main contributions of this thesis – applications of our framework to extract novel semantic guarantees provided by randomized response, FRAPP/PRAM, and random sampling. We show that they offer protections on different notions of parity of the input data. Since such protections are often unnecessary, we show, in Section 4.3.4, how to manipulate the row cone to relax privacy definitions.

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

Theorem 4.3.1. *Let $\mathbb{I} = \{D_1, \ldots, D_n\}$ be a finite set of possible datasets. Let $\mathfrak{M}^*$ be an algorithm with* domain($\mathfrak{M}^*$) $\subseteq \mathbb{I}$. *Let M^* be the matrix representation of $\mathfrak{M}^*$ (Definition 1.5.1)). If M^* is an invertible matrix then, denoting the i^{th} column of $(M^*)^{-1}$ as $m^{(i)}$,*

- *A vector $\vec{x} \in$* rowcone($\{\mathfrak{M}^*\}$) *if and only if $\vec{x} \cdot m^{(i)} \geq 0$ for every column $m^{(i)}$ of $(M^*)^{-1}$.*
- *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.*
- *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. See Section 4.5.5. □

Of our three examples (randomized response, random sampling, and FRAPP/PRAM), only randomized response directly satisfies the hypothesis of the theorem. Nevertheless, we will show that the theorem still helps us analyze random sampling and FRAPP/PRAM.

4.3.1 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 4.3.2 (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 4.3.3 (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 4.3.4. *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. See Section 4.5.6. □

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 4.3.5. *[CNF and rowcone] 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,*

$$\sum_{i=1}^{2^k} p^{\text{ham}(s,D_i)}(p-1)^{k-\text{ham}(s,D_i)} x_i \geq 0 \tag{4.2}$$

  where $\text{ham}(s, D_i)$ is 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. See Section 4.5.7 □

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

Example 4.3.6. (CNF of randomized response, $k = 2$). *Let $p > 1/2$. With 2 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:*

$$p^2 x_{00} + (1-p)^2 x_{11} \geq p(1-p)x_{01} + p(1-p)x_{10} \tag{4.3}$$

$$(1-p)^2 x_{00} + p^2 x_{11} \geq p(1-p)x_{01} + p(1-p)x_{10} \tag{4.4}$$

$$p^2 x_{01} + (1-p)^2 x_{10} \geq p(1-p)x_{00} + p(1-p)x_{11} \tag{4.5}$$

$$(1-p)^2 x_{01} + p^2 x_{10} \geq p(1-p)x_{00} + p(1-p)x_{11} \tag{4.6}$$

We use Example 4.3.6 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 4.3, 4.4, 4.5, and 4.6. Let us consider the following three attackers.

Attacker 1. This attacker has the prior beliefs that $P(\text{data} = 11) = p^2$, $P(\text{data} = 00) = (1-p)^2$, $(\text{data} = 01) = P(\text{data} = 10) = p(1-p)$, 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 4.3 on the attacker's inference. This constraint says that for all $\mathfrak{M}$ in the CNF of randomized response and for all $\omega \in \text{range}(\mathfrak{M})$,

$$\begin{aligned} & p^2 P[\mathfrak{M}(11) = \omega] + (1-p)^2 P[\mathfrak{M}(00) = \omega] \\ \geq \quad & p(1-p)P[\mathfrak{M}(01) = \omega] + p(1-p)P[\mathfrak{M}(10) = \omega] \end{aligned} \tag{4.7}$$

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 4.7, we get the constraint that for all $\omega \in \text{range}(\mathfrak{M})$:

$$\begin{aligned} & P(\text{data} = 11)P[\mathfrak{M}(11) = \omega] + P(\text{data} = 00)P[\mathfrak{M}(00) = \omega] \\ \geq\ & P(\text{data} = 01)P[\mathfrak{M}(01) = \omega] + P(\text{data} = 10)P[\mathfrak{M}(10) = \omega] \end{aligned}$$

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

$$\begin{aligned} & P[\text{parity}(\text{data}) = 0 \wedge \mathfrak{M}(\text{data}) = \omega] \\ \geq \quad & P[\text{parity}(\text{data}) = 1 \wedge \mathfrak{M}(\text{data}) = \omega] \end{aligned}$$

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:

$$\begin{aligned} & P[\text{parity}(\text{data}) = 0 \mid \mathfrak{M}(\text{data}) = \omega] \\ \geq \quad & P[\text{parity}(\text{data}) = 1 \mid \mathfrak{M}(\text{data}) = \omega] \end{aligned}$$

Thus $\mathfrak{M}$ guarantees that if and 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 the 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 indepen-

dent). Then, by similar calculations, Equation 4.5, implies that for this attacker

$$\begin{aligned} & P[\text{parity(data)} = 1 \mid \mathfrak{M}(\text{data}) = \omega] \\ \geq \quad & P[\text{parity(data)} = 0 \mid \mathfrak{M}(\text{data}) = \omega] \end{aligned}$$

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 4.3.7 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 CNF($\{\mathfrak{M}_{rr(p)}\}$). Also note that, by Theorem 4.3.5, an algorithm $\mathfrak{M}$ with matrix representation M belongs to CNF($\{\mathfrak{M}_{rr(p)}\}$) if and only if every row of M belongs to rowcone($\{\mathfrak{M}_{rr(p)}\}$). Thus the following theorem completely characterizes the privacy guarantees provided by randomized response.

Theorem 4.3.7. *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:*

- *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 Section 4.5.8. □

In many cases, protecting the parity of an entire dataset is not necessary in privacy preserving applications (in fact, some people find it odd).[2] 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 4.3.4.

Note that Kasiviswanathan et al. [90] proved a learning-theoretic separation result between randomized response and differential privacy which roughly states that randomized response cannot be used to efficiently learn a problem called MASKED-PARITY. That concept of parity involves solving a linear system of equations in a d-dimensional vector space over the integers modulo 2. While completely different from the notion of parity that we study, one direction of future work is to determine if our result about the semantic guarantees of randomized response can lead to a new proof of the result by Kasiviswanathan et al. [90]. One key step to generalize our framework is to interpret the semantic of a row vector r (the likelihood vector) along with the maximum probability (element) in this row vector. The maximum probability in r is the upper bound of the probability that this semantic (corresponding to r) can be learned. This is not possible with rowcone because rowcone ignores scaling factors. One future work is a different interpretation of CNF that does not ignore the scaling factors.

4.3.1.1 The relationship between randomized response and differential privacy

When setting $\epsilon = \log \frac{p}{1-p}$ then it is well known that randomized response satisfies ϵ-differential privacy. To be completed, we prove it using the consistent normal form in Lemma 4.3.8. 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 4.3.7 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.

Lemma 4.3.8. *Let* $\mathbb{I} = [0, 1]^N$ *be the input space of binary attributes of N people. Let* $\mathfrak{M}_{rr(p)}$ *be a randomized response algorithm as defined in Definition 4.3.3 for some* $p \in [0.5, 1]$. *Let* $C_2(\text{CNF}(\mathfrak{M}_{rr(p)}))$ *be set of constraints (convex cone) involving only two variables obtained by applying Fourier-Motzkin elimination on* $\text{CNF}(\mathfrak{M}_{rr(p)})$. *Then,* $C_2(\text{CNF}(\mathfrak{M}_{rr(p)}))$ *is exactly* ϵ*-differential privacy and* $e^\epsilon = \frac{p}{1-p}$. *That is,* $\forall i, j \in \mathbb{I}$ *and* $|i - j| = 1$, *we have*

$$x_i \geq e^{-\epsilon} x_j.$$

[2] 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.

Proof. See Section 4.5.9. □

4.3.1.2 Generalized randomized response

Now, we extend our result to more general cases where each person's data is not just a bit but a value from a finite domain. The following lemma characterizes the generalized randomized response. Informally, it says that an algorithm is generalized randomized response if and only if ,by looking at all but one person, it behaves as generalized randomized response. Intuitively, this is true because each person's data is perturbed independently.

Lemma 4.3.9. *Let $S = \{1, 2, ..., n\}$ be set of n people and $n \geq 3$. Let $V^{(i)} = \{1, ..., |V^{(i)}|\}$ be the domain (possible values) of person i. Let $\mathcal{A}^{(i)} : V^{(i)} \to V^{(i)}$ be a generalized randomized response of person i. Denote $\mathcal{P}_{b_i=v}$ be the projection on person i being v. Then, $\mathcal{A}$ is the generalized randomized response ($\prod_{i \in S} \bigotimes A^{(i)}$) if and only if*

1. *$\forall i \in S, A^{(i)}$ (after merging identical rows) is row linearly independent*
2. *$\forall i \in S, \forall v \in V^{(i)}, \mathcal{P}_{b_i=v}(A)$ is equal to $\prod_{j \in S, j \neq i} \bigotimes A^{(j)}$ or zero vector.*

Proof. See Section 4.5.10. □

Lemma 4.3.10. *Let $n \geq 3$. $\mathfrak{M}$ is a generalized randomized response of n people if and only if $\mathcal{P}_2(\mathfrak{M})$ is a randomized response of 2 people.*

Proof. It is easy to see that if $\mathfrak{M}$ is a randomized response of n people, then $\mathcal{P}_2(\mathfrak{M})$ is a randomized response of 2 people. Now, we prove the other direction. By Lemma 4.3.9, we know $\mathcal{P}_k(\mathfrak{M})$ is a randomized response of k people. Thus, $\mathfrak{M}$ is a randomized response of n people. □

4.3.2 FRAPP and PRAM

In some cases, computing the row cone of a privacy definition $\mathfrak{Priv}$ may not be possible. In those cases, even approximating the row cone can help us extract some of the semantic guarantees provided by the definition. If we can approximate rowcone($\mathfrak{Priv}$) with a strictly larger convex cone r', then we can derive some of the semantic guarantees provided by $\mathfrak{Priv}$ (we can think of these as lower bound on the true guarantees).

In this section, we apply these approximation ideas to FRAPP [7], which is a privacy definition based on the perturbation technique PRAM [8]. Recall from Section 2.2.3 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 4.3.11 (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 4.3.12 (γ-FRAPP [7])**.** *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 [54]: for all tuple values $a, b, c \in \mathcal{TUP}$, $\frac{P_Q(b \to a)}{P_Q(c \to a)} \leq \gamma$.*

We use the following terminology to discuss an approximation to the row cone.

Definition 4.3.13 (Approximation cone)**.** *Given a privacy definition $\mathfrak{Priv}$, an* approximation cone *of $\mathfrak{Priv}$ is a closed convex cone r' such that* rowcone($\mathfrak{Priv}$) $\subseteq r'$*.*

Any linear constraints satisfied by an approximation cone of $\mathfrak{Priv}$ are also satisfied by rowcone($\mathfrak{Priv}$) and thus the semantic guarantees extracted from those linear constraints are shared by $\mathfrak{Priv}$ as well.

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:

$$\begin{aligned} &\forall i_1, \ldots, i_k, j_1, \ldots, j_k \in \{1, \ldots, N\} : \\ &M_Q \left(\bigotimes_{\ell=1}^{k} (pe_{i_\ell} - (1-p)e_{j_\ell}) \right) \succeq \vec{0} \end{aligned} \tag{4.8}$$

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

Definition 4.3.14. (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 4.8 (where e_{j_ℓ} is the j_ℓ^{th} column vector of the $N \times N$ identity matrix).*

Lemma 4.3.15. *Let $p = \frac{\gamma}{\gamma+1}$. Then $\tilde{K}_p$ is an approximation cone for* γ-FRAPP*.*

Proof. See Section 4.5.11. □

The connection between the approximation cone $\tilde{K}_p$ of FRAPP and rowcone($\mathfrak{M}_{rr(p)}$), row cone of randomized response becomes, clear once we rephrase the linear constraints that define rowcone($\mathfrak{M}_{rr(p)}$) in Theorem 4.3.5 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 4.3.7, 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 $q_i \geq p = \frac{\gamma}{1+\gamma}$, then 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.

4.3.3 Random Sampling

Random sampling has often been studied as a method for protecting privacy [29]. We will examine the privacy provided by the following technique: an organization collects records from various individuals; each record is then deleted from the data with probability $1 - p$ (independently).

We view sampling as the combination of the following processes. To *collect* data, the data collector creates an ordered list of all of the individuals in the population. Each individual i has a record value r_i belonging to some domain $\mathcal{TUP}$. If individual i decides to participate in the survey, individual i gives r_i to the data collector. Thus the data collector ends up with an *ordered* dataset $D = [t_1, \ldots, t_W]$ where $t_i = r_i$ if individual i participated in the survey and $t_i =$"?" if individual i did not participate. To *anonymize* the data, for each individual i that participated in the survey, the data collector replaces the tuple t_i with the symbol "?" with

probability $1-p$. Finally, the data collector sorts all the tuples in the dataset to try to break the connection between individuals and the ordering of tuples in the data.

We will use our framework to analyze the privacy guarantees provided by this procedure. Since we need a matrix representation of the sampling algorithm, we need to order the set of all possible datasets. We will order them lexicographically based on an arbitrary ordering of the tuple domain.

Definition 4.3.16. (Domain of Random Sampling). *Define $\mathcal{TUP} = \{a_1, \ldots, a_N\}$ be the domain of tuples (ordered arbitrarily) and let "?" be a special symbol indicating a missing value. We extend the tuple ordering so that "?" is larger than any a_i. Let W be the size of the population. The input domain $\mathbb{I} = \{D_1, \ldots, D_n\}$ is the set of all databases containing W tuples where the tuple values belong to $\mathcal{TUP} \cup \{$"?"$\}$. The databases in $\mathbb{I}$ are ordered lexicographically.*

Definition 4.3.17. (p-Sampling). *Let $\mathfrak{M}_{drop(p)}$ be the algorithm that replaces each non-"?" tuple in its input with "?" independently and with probability $1-p$. Let $\mathfrak{M}_{sort}$ be the algorithm that sorts all the tuples in its input dataset. Define $\mathfrak{M}_{sample(p)}$, the sampling algorithm, to be the composition $\mathfrak{M}_{sort} \circ \mathfrak{M}_{drop(p)}$ (i.e. it replaces each non-"?" tuple in its input with "?" independently and with probability $1-p$ and then sorts the resulting tuples).*

To analyze the properties of sampling as a privacy definition, we need to analyze $\text{CNF}(\{\mathfrak{M}_{\text{sample}(p)}\})$ and the corresponding row cone $\text{rowcone}(\{\mathfrak{M}_{\text{sample}(p)}\})$. The algorithm $\mathfrak{M}_{\text{sample}(p)}$ can be seen as a two stage process that first randomly replaces tuples with "?" and then sorts the output. We can analyze each stage separate and combine their results using the following lemma.

Lemma 4.3.18. *Let $\mathfrak{M}_1$ and $\mathfrak{M}_2$ be two algorithms that commute ($\mathfrak{M}_1 \circ \mathfrak{M}_2 = \mathfrak{M}_2 \circ \mathfrak{M}_1$), let $\mathfrak{M}_1$ have a matrix representation that is invertible and let $\mathfrak{M}_2$ be idempotent. Then*

- $\text{CNF}(\{\mathfrak{M}_1 \circ \mathfrak{M}_2\}) = \text{CNF}(\{\mathfrak{M}_1\}) \cap \text{CNF}(\{\mathfrak{M}_2\})$
- $\text{rowcone}(\{\mathfrak{M}_1 \circ \mathfrak{M}_2\}) = \text{rowcone}(\{\mathfrak{M}_1\}) \cap \text{rowcone}(\{\mathfrak{M}_2\})$

Proof. See Section 4.5.12. □

The essence of the result in Lemma 4.3.18 is that the semantic guarantees of $\mathfrak{M}_1 \circ \mathfrak{M}_2$ are the conjunction of the guarantees of $\mathfrak{M}_1$ and $\mathfrak{M}_2$. If the two algorithms did not commute then the semantic guarantees of $\mathfrak{M}_1 \circ \mathfrak{M}_2$ could be strictly stronger than the conjunction of the guarantees of $\mathfrak{M}_1$ and $\mathfrak{M}_2$ (i.e. without commutativity, the whole can be greater than the sum of its parts). Using this lemma, we can show that the row cone of the sampling algorithm $\mathfrak{M}_{\text{sample}(p)}$ is:

Theorem 4.3.19. *Let $\mathfrak{M}$ be an algorithm and $o \in \text{range}(\mathfrak{M})$. The vector $(cP[\mathfrak{M}(D_1) = \omega], \ \ldots, \ cP[\mathfrak{M}(D_n) = \omega])$ belongs to $\text{rowcone}(\{\mathfrak{M}_{sample(p)}\})$ if and only if:*

- *$P(\mathfrak{M}(D_i) = \omega) = P(\mathfrak{M}(D_j) = \omega)$ whenever D_i and D_j are permutations of each other, and*
- $\forall i : \sum_{D_j \subseteq D_i} P(\mathfrak{M}(D_j) = \omega) \left(-(1-p)\right)^{blank(D_i, D_j)} \geq 0$

where the notation $D_j \subseteq D_i$ means that D_i can be converted to D_j by replacing tuples with "?", nonblank(D_i) is the number of tuples in D_i not equal to "?", and blank(D_i, D_j) is the number of tuples in D_j equal to "?" minus the number of tuples in D_i equal to "?".

Proof. See Section 4.5.13. □

The constraints on rowcone$(\{\mathfrak{M}_{\text{sample}(p)}\})$ imply the following semantic guarantees.

Theorem 4.3.20. *Suppose an attacker knows individuals $\{i_1, \ldots, i_k\}$ are a superset of those who participated in the survey. Furthermore, suppose the attacker knows that the individuals have record values $r_{j_1}, \ldots, r_{j_k}$ but is unsure about the true assignment σ of record value to individual (i.e. the attacker may know that exactly 25 of 138 individuals have cancer but does now know who are the cancer patients). If the attacker believes that each individual participated in the survey with probability $q \geq \frac{1}{2-p}$, then $\mathfrak{M}_{sample(p)}$ (and any other algorithm in* CNF$(\{\mathfrak{M}_{sample(p)}\})$*) guarantees that after seeing the sanitized data, the attacker learns nothing new about the true assignment. Furthermore, the attacker will believe that the parity of the subset of $\{i_1, \ldots, i_k\}$ who did not participate is more likely to be even than odd.*

Proof. See Section 4.5.14. □

Note that if the attacker knows about the participation in the survey of all individuals except for Bob, then Theorem 4.3.20 ensures that the attacker will believe that Bob is more likely to have participated (parity= 0) than not (parity= 1). The essence of Theorem 4.3.20 is *in the worst case*, one of the main protections offered by sampling is that it can prevent accusations that someone *did not* participate in the survey (and does not always protect individuals from accusations that they did participate). As this is a rather dubious piece of information to protect, this result suggests that the use of sampling to preserve privacy results in a loss of utility because it ends up protecting the pieces of information that users may not care about.

4.3.4 Relaxing Privacy Definitions

As we saw in Section 4.3.1, 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 [91], 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 4.3.6 (randomized response on databases with $k = 2$ tuples). Rewriting equations 4.3 and 4.6 to isolate x_{01} and setting $\alpha = p/(1-p)$, we get

$$\alpha x_{00} + x_{11}/\alpha - x_{10} \quad \geq \quad x_{01}$$

$$\geq \alpha x_{00} + \alpha x_{11} - \alpha^2 x_{10}$$
$$\Rightarrow x_{11} \leq \alpha x_{10}$$

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 4.3.6 yields the rest of the differential privacy constraints. Thus we see that differential privacy is a natural relaxation of randomized response.

4.4 Conclusions

In this chapter we presented the first (to the best of our knowledge) framework for extracting semantic guarantees from privacy definitions. The semantic guarantees target the least common denominator when facing a computationally unbounded Bayesian attacker – the guarantees need to hold no matter what sanitized output is produced by an algorithm satisfying the privacy definition. The framework depends on the concepts of consistent normal form CNF($\mathfrak{Priv}$) and on rowcone($\mathfrak{Priv}$). Both of these objects can be viewed as constraints on algorithms.

The importance of these two objects suggests that any new privacy definition $\mathfrak{Priv}$ should be presented using its normalized form CNF($\mathfrak{Priv}$) or rowcone($\mathfrak{Priv}$) so that its semantic guarantees can be better evaluated.

4.5 Proofs

4.5.1 Proof of Theorem 4.2.2

Theorem 4.2.2. *Given a privacy definition* $\mathfrak{Priv}$*, its consistent normal form* CNF($\mathfrak{Priv}$) *can be derived from the following process.*

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. We need to show that $\mathfrak{Priv}^{(2)}$ satisfies Axioms 3.1.1 and 3.1.2 consistent 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 3.1.2 (convexity). To show that $\mathfrak{Priv}^{(2)}$ satisfies Axiom 3.1.1 (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 $\mathfrak{M} = \text{choice}^p(\mathfrak{M}_1^{(1)}, \ldots, \mathfrak{M}_m^{(1)})$. It is easy to check that $\mathcal{A} \circ \mathfrak{M} = \text{choice}^p(\mathcal{A} \circ \mathfrak{M}_1^{(1)}, \ldots, \mathcal{A} \circ \mathfrak{M}_m^{(1)})$. 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 3.1.1 (post-processing).

Now let $\mathfrak{Priv}'$ be some privacy definition containing $\mathfrak{Priv}$ and satisfying both axioms. By Axiom 3.1.1 (post-processing), $\mathfrak{Priv}^{(1)} \subseteq \mathfrak{Priv}'$. By Axiom 3.1.2 (convexity) it follows that $\mathfrak{Priv}^{(2)} \subseteq \mathfrak{Priv}'$. Therefore $\text{CNF}\,\mathfrak{Priv} = \mathfrak{Priv}^{(2)} \subseteq \mathfrak{Priv}'$. □

4.5.2 Proof of Corollary 4.2.3

Corollary 4.2.3. *If* $\mathfrak{Priv} = \{\mathfrak{M}\}$ *consists of just one algorithm,* $\text{CNF}(\mathfrak{Priv})$ *is the set of all algorithms of the form* $\mathcal{A} \circ \mathfrak{M}$*, where* $\text{range}(\mathfrak{M}) \subseteq \text{domain}(\mathcal{A})$ *and the random bits in* $\mathcal{A}$ *and* $\mathfrak{M}$ *are independent of each other.*

Proof. We use the notation defined in Theorem 4.2.2. The corollary follows easily from process described in Theorem 4.2.2 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 $\text{CNF}(\mathfrak{Priv})$ has stopped after the first step. □

4.5.3 Proof of Theorem 4.2.5

Theorem 4.2.5. $\text{rowcone}(\mathfrak{Priv})$ *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 $\text{rowcone}(\mathfrak{Priv})$ 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 4.2.2 (and the post-processing Axiom 3.1.1), $\mathfrak{M}_1', \mathfrak{M}_2' \in \text{CNF}(\mathfrak{Priv})$ and

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

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 4.2.2, $\mathfrak{M}_* \in \text{CNF}(\mathfrak{Priv})$. Then for all $i = 1, 2, \dots,$

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

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. □

4.5.4 Proof of Theorem 4.2.6

Theorem 4.2.6. *Given a fixed schema with a quasi-identifier that contains an integer-valued attribute, the consistent normal form of k-anonymity consists of every algorithm whose input domain contains tables with this schema. The row cone consists of all vectors.*

Proof. Without loss of generality, assume Age is the integer-valued quasi-identifier attribute. Consider the algorithm $\mathfrak{M}_1$ that suppresses all attributes except for Age. It then sorts the tuples by Age (breaking ties arbitrarily). Using this sorted order, it puts the first k tuples into the first group, the second k-tuples into the second group, etc. For each group i, the age is coarsened into an age range $[a_i, b_i]$. The a_i and b_i are decimal numbers (e.g. 3.552) that encode all of the tuples in the group i. They have the following format. The number a_i is equal to the minimum age in group i minus 0.4. The number b_i has the form $\alpha_i + \beta_i$, where α_i is the maximum age in group i plus 0.5. β_i has the form $0.0\gamma_i$, where γ_i is a prefix-free encoding of the tuples in group i.

Clearly this algorithm $\mathfrak{M}_1$ satisfies k-anonymity and each input table is transformed into a unique "anonymized" table. Clearly there also exists a postprocessing algorithm $\mathcal{A}_2$ which can decode the "anonymized" table to recover the original table. By Axiom 3.1.1 (post-processing), the algorithm that first runs $\mathfrak{M}_1$ and then $\mathcal{A}_2$ (to recover the original table) belongs to the consistent normal form of k-anonymity. Since that algorithm is the identity, then by Axiom 3.1.1 (post-processing) all algorithms are in the consistent normal form of k-anonymity.

It easily follows that the row cone consists of all vectors. □

4.5.5 Proof of Theorem 4.3.1

Before proving this theorem, we need to introduce some definitions from convex analysis [86] and some intermediate results.

Definition 4.5.1. (Polyhedral Cone). *Let $v_1, ..., v_m \in \mathbb{R}^n$ be row-vectors. The* polyhedral cone *of $v_1, ..., v_m$, denoted by $C(v_1, ..., v_m)$, is the closed convex cone generated by $v_1, \ldots, v_n$: $C(v_1, \ldots, v_m) = \{\sum_{i=1}^{m} c_i v_i \mid c_i \geq 0\}$. If M is an $m \times n$ matrix with rows $v_1, \ldots, v_m$ then we define $C(M) \equiv C(v_1, \ldots, v_m)$.*

The next lemma says that the row cone of $\{\mathfrak{M}^*\}$ is a specific polyhedral cone that is related to the matrix representation M^* of $\mathfrak{M}^*$.

Lemma 4.5.2. *Let $\mathfrak{M}^*$ be an algorithm with finite domain $\mathbb{I}$ and let M^* be its matrix representation. Then* $\text{rowcone}(\{\mathfrak{M}^*\}) = C(M^*)$.

Proof. Using Corollary 4.2.3, it is easy to see that

$$\text{CNF}(\{\mathfrak{M}^*\}) \quad = \quad \{\mathcal{A} \circ \mathfrak{M}^* \quad : \quad \text{range}(\mathfrak{M}^*) \subseteq \text{domain}(\mathcal{A})\}$$

It is also easy to see that the matrix representation of $\mathcal{A} \circ \mathfrak{M}^*$ is equal to AM^*, where A is the matrix representation of $\mathcal{A}$ (thus composition of algorithms is equivalent to multiplication of the corresponding matrices). Now, each row of AM^* is equivalent to a nonnegative linear combination (with coefficients ≤ 1) of rows of M^*. Therefore each vector $\vec{x}$ is in $\text{rowcone}(\{\mathfrak{M}^*\})$ if and only if $\vec{x}$ is a nonnegative linear combination of rows of M^*, which is the same as saying $\text{rowcone}(\{\mathfrak{M}^*\}) = C(M^*)$. □

The next step is to find the linear constraints that determine $C(M^*)$ since these are the same constraints that determine $\text{rowcone}(\{\mathfrak{M}^*\})$. For this, we need the concept of a *dual cone*.

Definition 4.5.3. (Dual Cone). *Let $C \subseteq \mathbb{R}^n$ be a cone. The dual cone of C, denoted by C^*, is defined as $\{w \in \mathbb{R}^n | v \cdot w \geq 0, \ \forall v \in C\}$.*

We need the following result which applies to polyhedral cones such as $C(M^*)$ (note that $C(M^*)$ is a closed convex cone). The importance of this result is that it shows that $C(M^*)$, which is the equivalent to the row cone that we are interested in, is completely defined by the linear inequalities encapsulated by the dual cone $C^*(M^*)$. In other words, by definition of dual cone, $\vec{v} \in C(M^*)$ if and only if $\vec{v} \cdot \vec{w} \geq 0$ for all $\vec{w}$ in the dual cone $C^*(M^*)$.

Lemma 4.5.4. *[86]. Let C be a cone. Then C^* is a closed convex cone. If C is a closed and convex cone, then $C = C^{**}$ (C is the dual of its dual cone).*

The dual cone $C^*(M^*)$ contains infinitely many vectors and hence places infinitely many linear constraints that must be satisfied by $\vec{v}$ in order for v to be in $C(M*)$, the row cone we are interested in. The following result allows us to find a finite representative set of linear constraints.

Lemma 4.5.5. *[92]. Let M^* be an invertible matrix, then $C^*(M^*) = C(\ ((M^*)^{-1})^T\ )$. In other words, the dual cone is generated by the columns of the inverse of M^* – every vector in the dual cone is a positive linear combination of the (transpose of the) columns of $(M^*)^{-1}$*

From Lemmas 4.5.5 and 4.5.4, it easily follows that $\vec{v} \in C(M^*)$ if and only if $\vec{v} \cdot \vec{w} \geq 0$ for all $\vec{w}$ that are column vectors of $(M^*)^{-1}$.

We are now in position to prove Theorem 4.3.1.

Theorem 4.3.1. *Let $\mathbb{I} = \{D_1, \ldots, D_n\}$ be a finite set of possible datasets. Let $\mathfrak{M}^*$ be an algorithm with* domain$(\mathfrak{M}^*) \subseteq \mathbb{I}$*. Let M^* be the matrix representation of $\mathfrak{M}^*$ (Definition 1.5.1)). If M^* is an invertible matrix then, denoting the i^{th} column of $(M^*)^{-1}$ as $m^{(i)}$,*

- *A vector $\vec{x} \in$* rowcone$(\{\mathfrak{M}^*\})$ *if and only if $\vec{x} \cdot m^{(i)} \geq 0$ for every column $m^{(i)}$ of $(M^*)^{-1}$.*
- *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.*
- *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. The characterization of the row cone follows from Lemmas 4.5.2, 4.5.5, 4.5.4, and the discussion surrounding them. This proves the first part of the theorem.

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 proving the second part of the theorem automatically proves the third part.

To prove the second part of the theorem, let $A = M(M^*)^{-1}$. If we can show that the column sums of A are all 1 then, since A contains nonnegative 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 $\mathfrak{M} \in$ CNF$(\mathfrak{M}^*)$ by Theorem 4.2.2).

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 of length n whose components are all 1. Then since M is a matrix representation of an algorithm (Definition 1.5.1), 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.

□

4.5.6 Proof of Lemma 4.3.4

Lemma 4.3.4. *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 $\mathfrak{M}_{rr(1)}$ which always flips each bit in its input. It is easy to see that $\mathfrak{M}_{rr(1)} \circ \mathfrak{M}_{rr(p)} = \mathfrak{M}_{rr(1-p)}$ and $\mathfrak{M}_{rr(1)} \circ \mathfrak{M}_{rr(1-p)} = \mathfrak{M}_{rr(p)}$. From Theorem 4.2.2, 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 4.2.2, 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. □

4.5.7 Proof of Theorem 4.3.5

Theorem 4.3.5. *[CNF and rowcone] 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,*

$$\sum_{i=1}^{2^k} p^{\text{ham}(s,D_i)}(p-1)^{k-\text{ham}(s,D_i)} x_i \geq 0 \tag{4.2}$$

  where $\text{ham}(s, D_i)$ is 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 compute 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 4.3.1. Accordingly, we break the proof down into 3 steps.

Step 1: Compute $M_{rr(p)}$. 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 $\begin{pmatrix} c_{11}D & \dots & c_{1n}D \\ \vdots & \ddots & \vdots \\ c_{m1}D & \dots & c_{mn}D \end{pmatrix}$ 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

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

Step 2: Compute $(M_{rr(p)})^{-1}$. 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 compute 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

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

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

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

Step 3: Now we apply Theorem 4.3.1 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 $s = D_i$. □

4.5.8 Proof of Theorem 4.3.7

Theorem 4.3.7. *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:*

- *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 $\text{rowcone}(\mathfrak{M}_{rr(p)})$ 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 4.3.4 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:

$$P(\text{parity}(\text{data}) = 1) \quad = \quad \frac{1 - \prod_{j=1}^{k}(1 - 2q_j)}{2}$$

$$P(\text{parity(data)} = 0) \quad = \quad \frac{1 + \prod_{j=1}^{k} (1 - 2q_j)}{2}$$

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,

$$\begin{aligned} 0 \quad \leq \quad & \Big(P[\text{parity(data)} = 0] - P[\text{parity(data)} = 1]\Big) \\ \times \quad & \Big(P[\text{parity(data)} = 0 \mid \mathfrak{M}(\text{data})] \\ & -P[\text{parity(data)} = 1 \mid \mathfrak{M}(\text{data})]\Big) \end{aligned} \tag{4.9}$$

Now, it is easy to see that

$$\begin{aligned} & P(\text{parity(data)} = 0) - P(\text{parity(data)} = 1) \\ = \quad & \left[\bigotimes_{j=1}^{k}(-q_j,\ 1 - q_j)\right] \cdot \left[\bigotimes_{j=1}^{k}(1, 1)\right] \\ = \quad & \prod_{j=1}^{k}\Big[(-q_j,\ 1 - q_j) \cdot (1, 1)\Big] \end{aligned} \tag{4.10}$$

and

$$\begin{aligned} & P[\text{parity(data)} = 0 \mid \mathfrak{M}(\text{data})] - P[\text{parity(data)} = 1 \mid \mathfrak{M}(\text{data})] \\ & \qquad = \alpha \left[\bigotimes_{j=1}^{k}(-q_j, 1 - q_j)\right] \cdot \vec{x} \end{aligned} \tag{4.11}$$

where α is a positive normalizing constant and $\vec{x}$ is a vector of the matrix representation of $\mathfrak{M}$. So, by Equations 4.9, 4.10, and 4.11, the statement about protecting parity is equivalent to

$$\begin{aligned} & \forall \vec{x} \in \text{rowcone}(\{\mathfrak{M}_{rr(p)}\}) \\ 0 \quad \leq \quad & \left(\prod_{j=1}^{k}\Big[(-q_j,\ 1 - q_j) \cdot (1, 1)\Big]\right) \\ & * \left(\left[\bigotimes_{j=1}^{k}(-q_j,\ 1 - q_j)\right] \cdot \vec{x}\right) \end{aligned} \tag{4.12}$$

Step 2: Show that if for all j, $q_j \geq p \vee q_j \leq 1 - p$ then the constraints in Equation 4.12 hold

(i.e. the most likely parity *a priori* is the most likely parity *a posteriori*).

It follows from Corollary 4.2.3 that every $\mathfrak{M} \in \mathrm{CNF}(\{\mathfrak{M}_{rr(p)}\})$ has the form $\mathcal{A} \circ \mathfrak{M}_{rr(p)}$ and so, by Theorem 4.3.1, $\vec{x}$ is a row from the matrix representation of an $\mathfrak{M} \in \mathrm{CNF}(\{\mathfrak{M}_{rr(p)}\})$ if and only if $\vec{x} \in \mathrm{rowcone}(\{\mathfrak{M}_{rr(p)}\})$. This means that ever such $\vec{x}$ is a positive linear combination of rows of the randomized response algorithm $\mathfrak{M}_{rr(p)}$. Thus it suffices to show that

$$\begin{aligned} 0 \quad \leq \quad & \left(\prod_{j=1}^{k} \Big[(-q_j,\ 1-q_j) \cdot (1,1) \Big] \right) \\ & * \left(\left[\bigotimes_{j=1}^{k} (-q_j,\ 1-q_j) \right] \cdot \vec{m} \right) \end{aligned} \tag{4.13}$$

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 $\mathfrak{M}\, m$ 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 4.13 has the form:

$$\prod_{j=1}^{k} \Big[\Big((-q_j,\ 1-q_j) \cdot (1,1) \Big) * \Big((-q_j,\ 1-q_j) \cdot v_i \Big) \Big] \tag{4.14}$$

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 4.14 is nonnegative from which it follows that the conditions in Equation 4.13 and 4.12 are satisfied which implies Equation 4.9 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 4.12 in Step 2 (where we reformulated the problem into a statement that is more amenable to mathematical manipulation), we need to show that if

$$\begin{aligned} 0 \quad \leq \quad & \left(\prod_{j=1}^{k} \Big[(-q_j,\ 1-q_j) \cdot (1,1) \Big] \right) \\ & * \left(\left[\bigotimes_{j=1}^{k} (-q_j,\ 1-q_j) \right] \cdot \vec{x} \right) \end{aligned} \tag{4.15}$$

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

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

$$0 \quad \leq \quad \left(\left[\bigotimes_{j=1}^{k} (-q_j,\ 1-q_j) * \text{sign}(1-2q_j) \right] \cdot \vec{x} \right) \tag{4.16}$$

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

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

$$\vec{x}^T \bigoplus_{i=1}^{k} \frac{1}{2p-1} \begin{pmatrix} p & -(1-p) \\ -(1-p) & p \end{pmatrix} \tag{4.17}$$

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 computed in the proof of Theorem 4.3.5) 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 4.17 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 4.3.5 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
  $P(\text{parity}(J) = 0 \mid \mathfrak{M}(\text{data})) \geq P(\text{parity}(J) = 1 \mid \mathfrak{M}(\text{data}))$
- If $P(\text{parity}(J) = 1) \geq P(\text{parity}(J) = 0)$ then
  $P(\text{parity}(J) = 1 \ \wedge \ \mathfrak{M}(\text{data}) = \omega) \geq P(\text{parity}(J) = 0 \ \wedge \ \mathfrak{M}(\text{data}) = \omega)$

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

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

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. □

4.5.9 Proof of Lemma 4.3.8

Lemma 4.3.8. *Let $\mathbb{I} = [0,1]^N$ be the input space of binary attributes of N people. Let $\mathfrak{M}_{rr(p)}$ be a randomized response algorithm as defined in Definition 4.3.3 for some $p \in [0.5, 1]$. Let $C_2(\text{CNF}(\mathfrak{M}_{rr(p)}))$ be set of constraints (convex cone) involving only two variables obtained by applying Fourier-Motzkin elimination on $\text{CNF}(\mathfrak{M}_{rr(p)})$. Then, $C_2(\text{CNF}(\mathfrak{M}_{rr(p)}))$ is exactly ϵ-differential privacy and $e^\epsilon = \frac{p}{1-p}$. That is, $\forall i, j \in \mathbb{I}$ and $|i - j| = 1$, we have*

$$x_i \geq e^{-\epsilon} x_j.$$

Proof. Recall Theorem 4.3.5 proves that $\mathfrak{M} \in \text{CNF}(\mathfrak{M}_{rr(p)})$ if and only if for all row of the matrix representation M of $\mathfrak{M}$ is in rowcone($\{\mathfrak{M}_{rr(p)}\}$) as defined by Equation 4.2. Denote $S = \{x_1, ..., x_{2^N}\}$ be the set of variables(axises). Applying Fourier-Motzkin algorithm to eliminate variables is equivalently to projection. Thus, the linear constraints involving two variables, for example, x_i and x_j, is the projection of rowcone($\{\mathfrak{M}_{rr(p)}\}$) onto $S - \{x_i, x_j\}$ axises and denoted $C_{S-\{x_i,x_j\}}$. Because rowcone($\{\mathfrak{M}_{rr(p)}\}$), if viewed as a convex cone, is generated from column vectors of $M^T_{rr(p)}$, $C_{S-\{x_i,x_j\}}$ is generated from column vectors of $\mathcal{P}_{S-\{x_i,x_j\}}(M^T_{rr(p)})$. By Definition 4.3.3, one can prove $v \in \mathcal{P}_{S-\{x_i,x_j\}}(M^T_{rr(p)})$ if and only if there exists $c > 0$ such that $v = c(1, \frac{p}{1-p})^{\text{ham}(D_i,D_j)})$ or $v = c(1, \frac{p}{1-p})^{\text{ham}(D_i,D_j)})$. After summing up the identical (up to a positive scale factor) rows, one can obtain $C_{S-\{x_i,x_j\}}$ is generated by $r_1 = c(1, \frac{p}{1-p})^{\text{ham}(D_i,D_j)})$ and $r_2 = c(\frac{p}{1-p})^{\text{ham}(D_i,D_j)}, 1)$ where $c = \frac{1}{(1+\frac{p}{1-p})^{\text{ham}(D_i,D_j)}}$. Let $M = [r_1, r_2]$. By Lemma 4.5.5, the linear constraints of $C_{S-\{x_i,x_j\}}$ can be computed by M^{-1} and is algebraically manipulated below.

$$x_i \leq (\frac{p}{1-p})^{\text{ham}(D_i,D_j)} x_j \tag{4.18}$$

$$x_j \leq (\frac{p}{1-p})^{\text{ham}(D_i,D_j)} x_i \tag{4.19}$$

If D_i and D_j are neighbor and set $e^{-\epsilon} = \frac{p}{1-p}$, then we get ϵ-differential privacy constraints. In the case that D_i and D_j are not neighbors, the constraints are redundant. Let $\text{ham}(D_i, D_j) = k > 1$. One can obtain D_i from D_j by replacing k people's data. Then, applying ϵ-differential privacy k times, one can obtain Equation 4.18 and 4.19. □

Lemma 4.5.6. *Let $\mathfrak{M}^n_{rr(p)}$ be the randomized response of n people for some $p > 0.5$. Then, $\mathfrak{M}^n_{rr(p)}$ is $\ln \frac{p}{1-p}$-differentially private.*

Proof. One can easily check rows of $M^n_{rr(p)}$ satisfy $\ln \frac{p}{1-p}$-differentially private. □

Lemma 4.5.7. *Let $\mathfrak{M}^2_{rr(p)}$ be the randomized response of 2 people for some $p > 0.5$. Let $1 < k < \infty$ be a constant. There exists $p > 0.5$ such that $\frac{\ln \frac{p}{1-p}}{k}$-differential privacy is not in* $\mathrm{CNF}(\mathfrak{M}^2_{rr(p)})$.

Proof. To simplify the notation, let $\alpha = \frac{p}{1-p}$ and we can write the constraints of $\mathrm{CNF}(\mathfrak{M}^2_{rr(p)})$ as follows.

$$\alpha^2 x_{00} - \alpha x_{01} - \alpha x_{10} + x_{11} \geq 0$$
$$\alpha^2 x_{01} - \alpha x_{00} - \alpha x_{11} + x_{10} \geq 0$$
$$\alpha^2 x_{10} - \alpha x_{11} - \alpha x_{00} + x_{01} \geq 0$$
$$\alpha^2 x_{11} - \alpha x_{10} - \alpha x_{01} + x_{00} \geq 0$$

One can view that Equation 4.20, 4.20 and 4.20 are variation of Equation 4.20 by swapping bits. We pick

$$\alpha = (\frac{k+1}{k})^{\frac{1}{k-1}}.$$

First, note that $\alpha > 1$ and thus there exists $p > 0.5$ such that $\alpha = \frac{p}{1-p}$. □

4.5.10 Proof of Lemma 4.3.9

Lemma 4.3.9. *Let $S = \{1, 2, ..., n\}$ be set of n people and $n \geq 3$. Let $V^{(i)} = \{1, ..., |V^{(i)}|\}$ be the domain (possible values) of person i. Let $\mathcal{A}^{(i)} : V^{(i)} \to V^{(i)}$ be a generalized randomized response of person i. Denote $\mathcal{P}_{b_i=v}$ be the projection on person i being v. Then, $\mathcal{A}$ is the generalized randomized response ($\prod_{i\in S} \bigotimes A^{(i)}$) if and only if*

1. *$\forall i \in S, A^{(i)}$ (after merging identical rows) is row linearly independent*
2. *$\forall i \in S, \forall v \in V^{(i)}, \mathcal{P}_{b_i=v}(A)$ is equal to $\prod_{j\in S, j\neq i} \bigotimes A^{(j)}$ or zero vector.*

Proof. To begin with, we denote notations used in the proof. Let r be a row of A. Denote $r_{b_1,...,b_n}$ corresponds to the column where the value of the i-th person is b_i. Denote $r_{b_i,b_j,b_k,*}$ to be the row vector containing the elements where the value of the i-th person, the j-th person and the k-th person are b_i, b_j, and b_k respectively. For example, $r_{b_i,*}$ is equal to $\mathcal{P}_{b_i}(r)$. Now, we are ready to prove the lemma.

One can easily verify the only if direction by examining one person projections. We will prove the if direction. Let A be a matrix having $\prod_{i\in S} |V^{(i)}|$ columns. A satisfies the constraints of mechanisms (i.e., non-negative and each column sums up to 1). Without loss of generality, we assume A does not contain zero vector and A has no identical (up to a positive scale factor) row. Let r be a row of A. Because r is not $\vec{0}$ and $\forall v \in V_i, r_{b_1=v,*} \in \prod_{i\in S\setminus 1} \bigotimes A^{(i)}$ or $\vec{0}$, we must have for some $v_1 \in V^{(1)}$ such that $r_{b_1=v_1,*} \in \prod_{i\in S\setminus 1} \bigotimes A^{(i)}$. Thus, we can write, for some $v_1 \in V_1$, $r_{b_1=v_1,b_2,b_3,*}$ as Crock product of rows each from $A^(2), ..., A^{(n)}$. Note that we pick specific v_1 such

that $r_{b_1=v_1,*}$ is not zero vector. Thus, for some c, ω_2, ω_3 and X where $c > 0$, $a^{(k)}_{\omega_k,b_k}$ is the element of $A^{(k)}$ on ω_k-th row and b_k-th column, and $X \in \prod_{i=4}^{n} \bigotimes A^{(i)}$ (for the case of $n = 3$, $X = 1$),

$$r_{b_1=v_1,b_2,b_3,*} = c a^{(2)}_{\omega_2,b_2} a^{(3)}_{\omega_3,b_3} X.$$

Now, consider the projection of r onto the 2N person being, for example, v_2. Because of $r_{b_2=v_2,*} \in \prod_{i\in S\setminus 2} \bigotimes A^{(i)}$ or $r_{b_2=v_2,*} = \vec{0}$, we divided $V^{(2)}$ into two disjoint set $V^{(2)}_1$ and $V^{(2)}_0$ such that $v \in V^{(2)}_0 \iff r_{b_2=v,*} = \vec{0}$. Thus, for each $y_i \in V^{(2)}_1$, we can find an element of r, for example, $r_{b_1=x_1,b_2=y_i,b_3=x_3,\ldots,b_n=x_n} > 0$, where $x_i \in V^{(i)}$. Because $r_{b_2=y_i,*} \in \prod_{i\in S\setminus 2} \bigotimes A^{(i)}$, we know, for each $y_i \in V^{(2)}_1$, there exists a row $a^{(1)}_{z_i}$ of $A^{(1)}$ such that $a^{(1)}_{z_i,v_1} \neq 0$ (otherwise $r_{b_1=v_1,*} = \vec{0}$). Thus, for each $y_i V^{(2)}_1$, we can write

$$\begin{aligned} r_{b_1=x_1,b_2=y_i,b_3=x_3,*} &= r_{b_1=v_1,b_2=y_i,b_3=x_3,*} \frac{a^{(1)}_{z_i,x_1}}{a^{(1)}_{z_i,v_1}} \\ &= c a^{(2)}_{\omega_2,b_2} a^{(3)}_{\omega_3,b_3} \frac{a^{(1)}_{z_i,x_1}}{a^{(1)}_{z_i,v_1}} X \end{aligned}$$

Recall that $\forall v \in V^{(2)}_0, r_{b_2=v,*} = \vec{0}$. We now fully defined r in terms of variables. It is easy to see that if $z_1 = z_2 = \ldots = z_k$, then r is equal (up to a positive scale factor) to a row of $\prod_{i\in S} \bigotimes A^{(i)}$.

Now, consider the projection of r onto the 3rd person. We pick v_3 such that $r_{b_3=v_3,*} \neq \vec{0}$ and thus $r_{b_3=v_3,*} \in \prod_{i\in S\setminus 3} \bigotimes A^{(i)}$. Because r is not $\vec{0}$, we can pick $x_4, \ldots, x_n$ such that there exists x_1, x_2, x_3 and $r_{b_1=x_1,\ldots,b_n=x_n} > 0$. Moreover, we can set $x_1 = v_1$ and $x_2 = y_i \in V^{(2)}_1$ (because of $a^{(1)}_{z_i,v_1} > 0$ and $a^{(2)}_{\omega_2,y_i} > 0$). Because $r_{b_3=v_3,*} \in \prod_{i\in S\setminus 3} \bigotimes A^{(i)}$, we know, $\forall w_1 \in V^{(1)}$ and $\forall y_i, y_j \in V^{(2)}_1$,

$$\begin{aligned} \frac{r_{b_1=w_1,b_2=y_i,b_3=v_3,b_4=x_4,\ldots,b_n=x_n}}{b_1=v_1,b_2=y_i,b_3=v_3,b_4=x_4,\ldots,b_n=x_n} &= \frac{r_{b_1=w_1,b_2=y_j,b_3=v_3,b_4=x_4,\ldots,b_n=x_n}}{b_1=v_1,b_2=y_j,b_3=v_3,b_4=x_4,\ldots,b_n=x_n} \\ \iff \frac{a^{(1)}_{z_i,w_1}}{a^{(1)}_{z_i,v_1}} &= \frac{a^{(1)}_{z_j,w_1}}{a^{(1)}_{z_j,v_1}}. \end{aligned}$$

Thus, by varying w_1, we can conclude $z_i = z_j$. Thus, we write $\forall x_1 \in V^{(1)}, x_2 \in V^{(2)}_1, x_3 \in V^{(3)}$

$$r_{b_1=x_1,b_2=x_2,b_3=x_3,*} = c_2 a^{(1)}_{\omega_1,x_1} a^{(2)}_{\omega_2,x_2} a^{(3)}_{\omega_3,x_3} X \tag{4.20}$$

, for some ω_1 (a row index of $A^{(1)}$) and some $c_2 > 0$. Because of $r_{b_1=x_1,*} \in \prod_{i\in S\setminus 1} \bigotimes A^{(i)}$, we must also have $\forall y_i \in V^{(2)}_0, a^{(2)}_{\omega_2,y_i} = 0$. Thus, equation 4.20 also holds when $x_2 \in V^{(2)}_0$. We then conclude that r is identical (up to a positive scale factor) a row of $\prod_{i\in S} \bigotimes A^{(i)}$.

We conclude the proof by proving that r is exactly identical to a row of $\prod_{i\in S} \bigotimes A^{(i)}$ and due to mechanism constraint we must have A is identical to $\prod_{i\in S} \bigotimes A^{(i)}$ up to permutation. Let $r_1, \ldots, r_m$ be rows of $\prod_{i\in S} \bigotimes A^{(i)}$. Note that m must be smaller than or equal to $\prod_{i\in S} |V^{(i)}|$ (otherwise we must have for some j such that $A^{(j)}$ has more rows than columns which contradicts the assumption $A^{(j)}$ is row linearly independent). Then, there exists $c_1, \ldots, c_m \geq 0$ such that

$c_1 r_1, ..., c_m, r_m$ be rows of A. Note that because $\forall i, A^{(i)}$ is row linearly independent, we have $r_1, ..., r_m$ are linearly independent (property of Kronecker product). Because A is a mechanism, we must have sum of each column to be 1. There are m variables $c_1, ..., c_m$ and m linearly independent constraints. The solution of $c_1, ..., c_m$ must be unique and it is $\forall 1 \leq i \leq m, c_i = 1$. □

4.5.11 Proof of Lemma 4.3.15

Lemma 4.3.15. *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 Therefore 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

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

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

$$\begin{aligned} &\forall i_1, \ldots, i_k, j_1, \ldots, j_k \in \{1, \ldots, N\} : \\ &\quad M_Q \left(\bigotimes_{\ell=1}^{k} (pe_{i_\ell} - (1-p)e_{j_\ell}) \right) \succeq \vec{0} \end{aligned} \tag{4.21}$$

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

From Theorem 4.2.2, 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 choose 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 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 $\mathcal{A}^*$ is 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) $\subseteq \tilde{K}_p$. □

4.5.12 Proof of Lemma 4.3.18

Lemma 4.3.18. *Let $\mathfrak{M}_1$ and $\mathfrak{M}_2$ be two algorithms that commute ($\mathfrak{M}_1 \circ \mathfrak{M}_2 = \mathfrak{M}_2 \circ \mathfrak{M}_1$), let $\mathfrak{M}_1$ have a matrix representation that is invertible and let $\mathfrak{M}_2$ be idempotent. Then*

- $\text{CNF}(\{\mathfrak{M}_1 \circ \mathfrak{M}_2\}) = \text{CNF}(\{\mathfrak{M}_1\}) \cap \text{CNF}(\{\mathfrak{M}_2\})$
- $\text{rowcone}(\{\mathfrak{M}_1 \circ \mathfrak{M}_2\}) = \text{rowcone}(\{\mathfrak{M}_1\}) \cap \text{rowcone}(\{\mathfrak{M}_2\})$

Proof. Let $\mathbb{I}$ be the input domain. Since $\mathfrak{M}_1$ and $\mathfrak{M}_2$ commute, it is clear that $\text{range}(\mathfrak{M}_1) \subseteq \mathbb{I}$ and $\text{range}(\mathfrak{M}_2) \subseteq \mathbb{I}$ (i.e. the ranges must be subsets of the domain of the algorithms). Let M_1 and M_2 be the matrix representations of $\mathfrak{M}_1$ and $\mathfrak{M}_2$ respectively. Furthermore, since $\mathfrak{M}_1$ has a matrix representation that is invertible, $\text{range}(\mathfrak{M}_1) = \mathbb{I}$ and therefore $\text{range}(\mathfrak{M}_2) \subseteq \text{range}(\mathfrak{M}_1)$. We expand the range of $\mathfrak{M}_2$ so that it equals the range of $\mathfrak{M}_1$ (some outputs will just have 0 probability for all inputs). Then $\mathfrak{M}_1$ and $\mathfrak{M}_2$ have matrix representations M_1 and M_2 where the columns are indexed by datasets in $\mathbb{I}$ and the rows are also indexed by datasets in $\mathbb{I}$ (in the case of M_2, some of those rows may contain only 0 values). This ensures that the corresponding matrix representations M_1 and M_2 are square matrices. The commutativity of $\mathfrak{M}_1$ and $\mathfrak{M}_2$ imply that $M_1 M_2 = M_2 M_1$ and the fact that $\mathfrak{M}_2$ is idempotent (i.e. $\mathfrak{M}_2 \circ \mathfrak{M}_2 = \mathfrak{M}_2$) implies $M_2 M_2 = M_2$.

By Corollary 4.2.3, we see that

- $\text{CNF}(\{\mathfrak{M}_1\})$ is the set of algorithms of the form $\mathcal{A} \circ \mathfrak{M}_1$ where $\mathcal{A}$ is any postprocessing algorithm (whose domain contains the range of $\mathfrak{M}_1$.
- $\text{CNF}(\{\mathfrak{M}_2\})$ is the set of algorithms of the form $\mathcal{A} \circ \mathfrak{M}_2$
- $\text{CNF}(\{\mathfrak{M}_1 \circ \mathfrak{M}_2\})$ is the set of algorithms of the form $\mathcal{A} \circ \mathfrak{M}_1 \circ \mathfrak{M}_2$. By commutativity of $\mathfrak{M}_1$ and $\mathfrak{M}_2$ this is also the set of algorithms of the form $\mathcal{A} \circ \mathfrak{M}_2 \circ \mathfrak{M}_1$.

Now, let $\mathfrak{M}_3$ be an algorithm in $\text{CNF}(\{\mathfrak{M}_1 \circ \mathfrak{M}_2\})$. Then $\mathfrak{M}_3 = \mathcal{A}_1 \circ \mathfrak{M}_1 \circ \mathfrak{M}_2$ (for some $\mathcal{A}_1$) so that $\mathfrak{M}_3 \in \text{CNF}(\{\mathfrak{M}_2\})$. By commutativity, $\mathfrak{M}_3 = \mathcal{A}_2 \circ \mathfrak{M}_2 \circ \mathfrak{M}_1$ (for some $\mathcal{A}_2$) so that $\mathfrak{M}_3 \in \text{CNF}(\{\mathfrak{M}_1\})$ and therefore $\mathfrak{M}_3 \in \text{CNF}(\{\mathfrak{M}_2\}) \cap \text{CNF}(\{\mathfrak{M}_1\})$.

To prove the other direction, choose a $\mathfrak{M}_4 \in \text{CNF}(\{\mathfrak{M}_2\}) \cap \text{CNF}(\{\mathfrak{M}_1\})$. Then $\mathfrak{M}_4 = \mathcal{A}_4 \circ \mathfrak{M}_2$ for some $\mathcal{A}_4$. In terms of the corresponding matrix representations, we have:

$$\begin{aligned} M_4 &= A_4 M_2 \\ &= A_4 M_2 M_1^{-1} M_1 \\ &\quad (M_1^{-1} \text{ exists by hypothesis}) \\ &= A_4 M_1^{-1} M_2 M_1 \\ &\quad (\text{since } M_1 \text{ and } M_2 \text{ commute if and only if} \\ &\quad M_1^{-1} \text{ and } M_2 \text{ commute}) \end{aligned} \tag{4.22}$$

Now, note that a $A_4 M_1^{-1} M_2$ is the matrix representation of an algorithm because of the following facts: (1) $M_4 = A^* M_1$ where A^* is the matrix representation of some algorithm since $\mathfrak{M}_4 \in$

$\mathrm{CNF}(\{\mathfrak{M}_1\})$ (by assumption), (2) $A^* M_1 = M_4 = (A_4 M_1^{-1} M_2) M_1$ by Equation 4.22 and so $A^* = A_4 M_1^{-1} M_2$ because M_1 is invertible. Therefore

$$\begin{aligned} M_4 &= A_4 M_1^{-1} M_2 M_1 \\ &= A_4 M_1^{-1} M_2 M_2 M_1 \\ &\quad \text{(since } \mathfrak{M}_2\text{, and therefore } M_2\text{, is idempotent)} \\ &= (A_4 M_1^{-1} M_2) M_1 M_2 \\ &\quad \text{(by commutativity)} \end{aligned}$$

and therefore $M_4 \in \mathrm{CNF}(\{\mathfrak{M}_1 \circ \mathfrak{M}_2\})$ since we had shown that $(A_4 M_1^{-1} M_2)$ is the matrix representation of an algorithm.

We now prove the corresponding statement for row cones. The results for the consistent normal form immediately imply $\mathrm{rowcone}(\{\mathfrak{M}_1 \circ \mathfrak{M}_2\}) \subseteq \mathrm{rowcone}(\{\mathfrak{M}_1\}) \cap \mathrm{rowcone}(\{\mathfrak{M}_2\})$.

Now consider a vector $\vec{x} \in \mathrm{rowcone}(\{\mathfrak{M}_1\}) \cap \mathrm{rowcone}(\{\mathfrak{M}_2\})$. Then there is a an algorithm $\mathfrak{M}_1' \in \mathrm{CNF}(\{\mathfrak{M}_1\})$ such that $c_1\vec{x}$ is a row of the corresponding matrix representation M_1' (for some $c_2 > 0$). Similarly, there is a $\mathfrak{M}_2' \in \mathrm{CNF}(\{\mathfrak{M}_2\})$ such that $c_2\vec{x}$ is a row of the corresponding matrix representation M_2' (for some $c_2 > 0$). By appropriate postprocessing[3], we can assume that $\mathfrak{M}_1$ and $\mathfrak{M}_2$ have only two possible outputs. Without loss of generality assume $c_1 \geq c_2$. In this case, it is easy to construct and algorithm $\mathcal{A}$ so that $\mathcal{A} \circ \mathfrak{M}_1' = \mathfrak{M}_2'$. Hence $\mathfrak{M}_2' \in \mathrm{CNF}(\{\mathfrak{M}_2\}) \cap \mathrm{CNF}(\{\mathfrak{M}_1\})$ and therefore $\mathfrak{M}_2' \in \mathrm{CNF}(\{\mathfrak{M}_1 \circ \mathfrak{M}_2\})$. Since $c_2\vec{x}$ is a row of the matrix representation M_2', this means that $\vec{x} \in \mathrm{rowcone}(\{\mathfrak{M}_1 \circ \mathfrak{M}_2\})$. □

4.5.13 Proof of Theorem 4.3.19

In this section we prove Theorem 4.3.19. Before doing so, we first need to compute the matrix representation of the sampling algorithm $\mathfrak{M}_{\mathrm{sample}(p)}$ (defined in Definition 4.3.17). As discussed in Definition 4.3.16, a typical input to $\mathfrak{M}_{\mathrm{sample}(p)}$ consists of a sequence of tuples, one per individual in the population. If i^{th} individual did not provide any information for the survey, then the i^{th} tuple value is "?". If the i^{th} individual did provide information, then the i^{th} tuple is the record corresponding to that information.

The sampling algorithm $\mathfrak{M}_{\mathrm{sample}(p)}$ (Definition 4.3.17) can be expressed as a composition of two other algorithms:

$$\mathfrak{M}_{\mathrm{sample}(p)} = \mathfrak{M}_{\mathrm{sort}} \circ \mathfrak{M}_{\mathrm{drop}(p)}$$

where $\mathfrak{M}_{\mathrm{drop}(p)}$ replaces tuples with "?" independently and with probability $1 - p$, and $\mathfrak{M}_{\mathrm{sort}}$ sorts the tuples in its input dataset. Letting $M_{\mathrm{sample}(p)}$, $M_{\mathrm{drop}(p)}$, and $\mathrm{M}_{\mathrm{sort}}$ be the corresponding matrix representations, we see that

$$M_{\mathrm{sample}(p)} = \mathrm{M}_{\mathrm{sort}}\, M_{\mathrm{drop}(p)}$$

[3] i.e. using an algorithm that maps outputs not associated with a designated row to the same symbol

$$M_{\text{drop}(p)} = \bigoplus_{i=1}^{W} B_p = \begin{array}{c} \\ aa \\ ab \\ a? \\ ba \\ bb \\ b? \\ ?a \\ ?b \\ ?? \end{array} \begin{array}{c} \begin{array}{ccccccccc} aa & ab & a? & ba & bb & b? & ?a & ?b & ?? \end{array} \\ \begin{pmatrix} p^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & p^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ p(1-p) & p(1-p) & p & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & p^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & p^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & p(1-p) & p(1-p) & p & 0 & 0 & 0 \\ p(1-p) & 0 & 0 & p(1-p) & 0 & 0 & p & 0 & 0 \\ 0 & p(1-p) & 0 & 0 & p(1-p) & 0 & 0 & p & 0 \\ (1-p)^2 & (1-p)^2 & 1-p & (1-p)^2 & (1-p)^2 & 1-p & 1-p & 1-p & 1 \end{pmatrix} \end{array}$$

Figure 4.2. Matrix representation of $\mathfrak{M}_{\text{drop}(p)}$ with $\mathcal{TUP} = \{a, b\}$ and $W = 2$

The matrix representation M_{sort} is easy to compute. The rows are indexed by the set of databases of size W (the number of people in the population) and correspond to possible outputs. The columns are indexed by the set of databases of size W and correspond to possible inputs. A column corresponding to an input dataset D contains 0 entries everywhere except in the row corresponding to the sorted version of D (by convention "?" are considered larger than other tuple values). For example, when the domain of tuples $\mathcal{TUP} = \{a, b\}$ and the population size $W = 2$ then the matrix representation of $\mathfrak{M}_{\text{sort}}$ is:

$$\mathrm{M}_{\text{sort}} = \begin{array}{c} \\ aa \\ ab \\ a? \\ ba \\ bb \\ b? \\ ?a \\ ?b \\ ?? \end{array} \begin{array}{c} \begin{array}{ccccccccc} aa & ab & a? & ba & bb & b? & ?a & ?b & ?? \end{array} \\ \begin{pmatrix} \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & \mathbf{1} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} \end{pmatrix} \end{array}$$

The matrix representation of $\mathfrak{M}_{\text{drop}(p)}$ has the form

$$M_{\text{drop}(p)} = \bigoplus_{i=1}^{W} B_p \tag{4.23}$$

$$(i,j)\text{-entry of } B_p = \begin{cases} 1-p & \text{if } i = N+1, j < N+1 \\ p & \text{if } i = j, i \neq N+1 \\ 1 & \text{if } i = j = N+1 \\ 0 & \text{otherwise} \end{cases} \tag{4.24}$$

where $\bigoplus$ is the Kronecker product, B_p is an $N+1 \times N+1$ matrix (recall $N = |\,\mathcal{TUP}\,|$) whose first N diagonal entries are p, the first N entries of the last row are $1-p$, the last diagonal entry is 1 and all other entries are 0. When the duple domain is $\mathcal{TUP} = \{a, b\}$ and the population size $W = 2$ then B_p is:

$$B_p = \begin{pmatrix} p & 0 & 0 \\ 0 & p & 0 \\ 1-p & 1-p & 1 \end{pmatrix}$$

and the matrix representation of $\mathfrak{M}_{\text{drop}(p)}$ is shown in Figure 4.2.

Now, we are ready to prove Theorem 4.3.19.

Theorem 4.3.19. *Let $\mathfrak{M}$ be an algorithm and $o \in$ range$(\mathfrak{M})$. The vector $(cP[\mathfrak{M}(D_1) = \omega],\ \ldots,\ cP[\mathfrak{M}(D_n) = \omega])$ belongs to* rowcone$(\{\mathfrak{M}_{sample(p)}\})$ *if and only if:*

- $P(\mathfrak{M}(D_i) = \omega) = P(\mathfrak{M}(D_j) = \omega)$ *whenever D_i and D_j are permutations of each other, and*
- $\forall i : \sum_{D_j \subseteq D_i} P(\mathfrak{M}(D_j) = \omega)\,(-(1-p))^{blank(D_i, D_j)} \geq 0$

where the notation $D_j \subseteq D_i$ means that D_i can be converted to D_j by replacing tuples with "?", nonblank(D_i) is the number of tuples in D_i not equal to "?", and blank(D_i, D_j) is the number of tuples in D_j equal to "?" minus the number of tuples in D_i equal to "?".

Proof. Our strategy is to apply Lemma 4.3.18 to the algorithm $\mathfrak{M}_{\text{sample}(p)} = \mathfrak{M}_{\text{sort}} \circ \mathfrak{M}_{\text{drop}(p)}$.

Step 1: computing the row cone rowcone$(\{\mathfrak{M}_{\text{sort}}\})$. Consider a partition on the input datasets such that two datasets D_i and D_j are in the same partition if and only if sorting their tuples gives the same result. Note that this is equivalent to saying that D_i is a permutation of D_j. By Corollary 4.2.3, CNF$(\{\mathfrak{M}_{\text{sort}}\})$ is the set of algorithms of the form $\mathcal{A} \circ \mathfrak{M}_{\text{sort}}$ and therefore the row cone consists of all positive linear combinations of rows of the corresponding matrix representation M_{sort}. It is easy to see that a vector with nonnegative entries is a positive linear of rows of M_{sort} if and only if for all pairs of datasets D_i and D_j that are in the same partition, the components corresponding to D_i and D_j are the same. Therefore, given an algorithm $\mathfrak{M}$ and some output $\omega \in$ range$(\mathfrak{M})$, the vector

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

belongs to rowcone$(\{\mathfrak{M}_{\text{sort}}\})$ if and only if $P(\mathfrak{M}(D_i) = \omega) = P(\mathfrak{M}(D_j) = \omega)$ whenever D_i and D_j are permutations of each other.

Step 2: computing the row cone rowcone$(\{\mathfrak{M}_{\text{drop}(p)}\})$. The corresponding matrix representation

is $M_{\text{drop}(p)} = \bigoplus_{i=1}^{W} B_p$ where B_p is the $N+1 \times N+1$ matrix such that:

$$(i,j)\text{-entry of } B_p \quad = \quad \begin{cases} p & \text{if } i = j, i \neq N+1 \\ 1-p & \text{if } i = N+1, j < N+1 \\ 1 & \text{if } i = j = N+1 \\ 0 & \text{otherwise} \end{cases}$$

The inverse $(B_p)^{-1}$ is easy to compute and equals:

$$(i,j)\text{-entry of } (B_p)^{-1} \quad = \quad \begin{cases} \frac{1}{p} & \text{if } i = j, i \neq N+1 \\ -\frac{1-p}{p} & \text{if } i = N+1, j < N+1 \\ 1 & \text{if } i = j = N+1 \\ 0 & \text{otherwise} \end{cases}$$

and thus the inverse of $M_{\text{drop}(p)}$ exists and equals $(M_{\text{drop}(p)})^{-1} = \bigoplus_{i=1}^{W} (B_p)^{-1}$. Now, according to Theorem 4.3.1, a vector $\vec{x} \in \text{rowcone}(\{\mathfrak{M}_{\text{drop}(p)}\})$ if and only if $x \cdot m^{(i)} \geq 0$ for every column $m^{(i)}$ of $(M_{\text{drop}(p)})^{-1}$. Thus we need to enumerate the columns of $(M_{\text{drop}(p)})^{-1} = \bigoplus_{i=1}^{W} (B_p)^{-1}$. Now, define the functions:

1. $\text{super}(D_i, D_j)$ is 1 if D_i can be turned into D_j by replacing some tuples with "?", and it is 0 otherwise.

2. $\text{blank}(D_i, D_j)$ is the number of tuples in D_j whose value is "?" minus the number of tuples in D_i whose value is "?".

3. $\text{nonblank}(D_i)$ is the number of tuples in D_i whose value is not "?"

Note that if $\text{super}(D_i, D_j) = 1$ then $\text{blank}(D_i, D_j) \geq 0$ because it is the number of tuples that need to be changed into "?" in order to convert D_i into D_j. With these definitions, we can enumerate the columns of $(M_{\text{drop}(p)})^{-1}$ in the following way. We associate a dataset D_i to the i^{th} column $m^{(i)}$ of $(M_{\text{drop}(p)})^{-1}$. A simple proof by induction on W shows that[4]:

$$\begin{aligned} &j^{\text{th}} \text{ entry of } m^{(i)} \\ &\quad = \text{super}(D_i, D_j) \frac{(-(1-p))^{\text{blank}(D_i, D_j)}}{p^{\text{nonblank}(D_i)}} \end{aligned}$$

For example, Figure 4.3 shows $(M_{\text{drop}(p)})^{-1}$ for the case $W = 2$ and $\mathcal{TUP} = \{a, b\}$. The second column, $m^{(2)}$ is associated with the dataset $D_2 = ab$. Its 4^{th} entry is 0 because $D_4 = ba$ and

[4]Note in the induction, the base case is $(M_{\text{drop}(p)})^{-1} = (B_p)^{-1}$ and the general case is $(M_{\text{drop}(p)})^{-1} = \bigoplus_{i=1}^{W} (B_p)^{-1}$. For reference, Figure 4.3 shows $(M_{\text{drop}(p)})^{-1} = \bigoplus_{i=1}^{2} (B_p)^{-1}$ for the case $W = 2$.

$$(M_{\text{drop}(p)})^{-1} = \bigoplus_{i=1}^{2}(B_p)^{-1} = \bigoplus_{i=1}^{2}\begin{pmatrix} \frac{1}{p} & 0 & 0 \\ 0 & \frac{1}{p} & 0 \\ \frac{-(1-p)}{p} & \frac{-(1-p)}{p} & 1 \end{pmatrix}$$

$$= \begin{array}{c} \\ aa \\ ab \\ a? \\ ba \\ bb \\ b? \\ ?a \\ ?b \\ ?? \end{array} \begin{array}{c} \begin{array}{ccccccccc} aa & ab & a? & ba & bb & b? & ?a & ?b & ?? \end{array} \\ \begin{pmatrix} \frac{1}{p^2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{p^2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{-(1-p)}{p^2} & \frac{-(1-p)}{p^2} & \frac{1}{p} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{p^2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{p^2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{-(1-p)}{p^2} & \frac{-(1-p)}{p^2} & \frac{1}{p} & 0 & 0 & 0 \\ \frac{-(1-p)}{p^2} & 0 & 0 & \frac{-(1-p)}{p^2} & 0 & 0 & \frac{1}{p} & 0 & 0 \\ 0 & \frac{-(1-p)}{p^2} & 0 & 0 & \frac{-(1-p)}{p^2} & 0 & 0 & \frac{1}{p} & 0 \\ \frac{(1-p)^2}{p^2} & \frac{(1-p)^2}{p^2} & \frac{-(1-p)}{p} & \frac{(1-p)^2}{p^2} & \frac{(1-p)^2}{p^2} & \frac{-(1-p)}{p} & \frac{-(1-p)}{p} & \frac{-(1-p)}{p} & 1 \end{pmatrix} \end{array}$$

Figure 4.3. $(M_{\text{drop}(p)})^{-1}$ when $\mathcal{TUP} = \{a, b\}$ and $W = 2$

there is no way to convert ab to ba by replacing tuple values with "?". On the other hand, the 3rd entry is $\frac{-(1-p)}{p^2}$ because $D_3 = a?$ and $\text{super}(D_2, D_3) = 1$ (we can replace the b with "?"), $\text{blank}(D_2, D_3) = 1$, and $\text{nonblank}(D_2) = 2$.

Now, according to Theorem 4.3.1, the vector

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

(for some algorithm $\mathfrak{M}$ and output $\omega \in \text{range}(\mathfrak{M})$) belongs to $\text{rowcone}(\{\mathfrak{M}_{\text{drop}(p)}\})$ if and only if its dot product with every column $m^{(i)}$ of $(M_{\text{drop}(p)})^{-1}$ is nonnegative. Using our method of associating D_i with $m^{(i)}$, the condition becomes:

$$\forall i: \quad \sum_{D_j \subseteq D_i} P(\mathfrak{M}(D_j) = \omega) \frac{(-(1-p))^{\text{blank}(D_i, D_j)}}{p^{\text{nonblank}(D_i)}} \geq 0$$

where we use the notation $D_j \subseteq D_i$ to mean that D_i can be converted to D_j by changing some tuple values to "?". We then multiply by $p^{\text{nonblank}(D_i)}$ without affecting the inequalities to get:

$$\forall i: \quad \sum_{D_j \subseteq D_i} P(\mathfrak{M}(D_j) = \omega) \left(-(1-p)\right)^{\text{blank}(D_i, D_j)} \geq 0$$

Step 3: combining the results. Since the matrix representation of $\mathfrak{M}_{\text{drop}(p)}$ is invertible, $\mathfrak{M}_{\text{drop}(p)}$ and $\mathfrak{M}_{\text{sort}}$ commute, $\mathfrak{M}_{\text{sort}}$ is idempotent, and $\mathfrak{M}_{\text{sample}(p)} = \mathfrak{M}_{\text{sort}} \circ \mathfrak{M}_{\text{drop}(p)}$, we can use Lemma

4.3.18 to combine the results from the previous two steps. Thus a vector

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

belongs to rowcone($\{\mathfrak{M}_{\text{sample}(p)}\}$) if and only if:

$$\forall i: \sum_{D_j \subseteq D_i} P(\mathfrak{M}(D_j) = \omega)\,(-(1-p))^{\text{blank}(D_i, D_j)} \geq 0$$

and $P(\mathfrak{M}(D_i) = \omega) = P(\mathfrak{M}(D_j) = \omega)$ whenever D_i and D_j are permutations of each other. □

4.5.14 Proof of Theorem 4.3.20

Theorem 4.3.20. *Suppose an attacker knows individuals $\{i_1, \ldots, i_k\}$ are a superset of those who participated in the survey. Furthermore, suppose the attacker knows that the individuals have record values $r_{j_1}, \ldots, r_{j_k}$ but is unsure about the true assignment σ of record value to individual (i.e. the attacker may know that exactly 25 of 138 individuals have cancer but does now know who are the cancer patients). If the attacker believes that each individual participated in the survey with probability $q \geq \frac{1}{2-p}$, then $\mathfrak{M}_{sample(p)}$ (and any other algorithm in* CNF($\{\mathfrak{M}_{sample(p)}\}$)*) guarantees that after seeing the sanitized data, the attacker learns nothing new about the true assignment. Furthermore, the attacker will believe that the parity of the subset of $\{i_1, \ldots, i_k\}$ who did not participate is more likely to be even than odd.*

Proof. Let $\mathfrak{M}$ be an algorithm such that every row of its matrix representation M belongs to rowcone($\{\mathfrak{M}_{\text{sample}(p)}\}$).

Let $\{i_1, \ldots, i_k\}$ be a set of k individuals that is a superset of the individuals who participated in the survey. Suppose that an attacker knows that collectively their record values are $r_{j_1}, \ldots, r_{j_k}$ although the attacker may not know the specific assignment of record value to individual.

Let σ be any assignment of the record values $r_{j_1}, \ldots, r_{j_k}$ to the individuals $\{i_1, \ldots, i_k\}$. That is, $\sigma(i_1)$ is the record value assigned to individual i_1, etc. Let $D^\sigma \in \mathbb{I}$ be the possible input dataset that is determined by σ: the tuples corresponding to individuals $i_1, \ldots, i_k$ have values $\sigma(i_1), \ldots, \sigma(i_k)$, respectively, and all other tuples are set to "?".

Step 1: reduction to the case where $k = W$ (where W is the number of individuals in the population).

Without loss of generality we can assume $k = W$. To see why, if an individual j is not one of the $i_1, \ldots, i_k$, then the attacker knows for sure that individual j did not participate in the survey. In this case, it is impossible for $\mathfrak{M}_{\text{sample}(p)}$ to output any sanitized data in which the j^{th} tuple is different from "?" and it is impossible to have any input dataset where the j^{th} tuple is different from "?". We can therefore redefine the input space to consist of all individuals except j and $\mathfrak{M}_{\text{sample}(p)}$ operates as before (by dropping tuples independently and sorting the result). In other words, if individual j did not participate in the survey with probability 1, we can just

pretend individual j never existed. We can repeat this process of eliminating from consideration all individuals not in $\{i_1, \ldots, i_k\}$.

Therefore, without loss of generality, we set $k = W$ so that the attacker knows that individuals $1, \ldots, W$ and knows that their record values are $r_{i_1}, \ldots, r_{i_W}$ but may not know the specific assignment of records to individuals. Furthermore, the attacker believes that each individual participated in the survey with (independent) probability $q \geq \frac{1}{2-p}$.

Step 2: show that the relative preferences of assignments is the same.

If σ' is any other assignment and $D^{\sigma'}$ the corresponding database[5] then, by Theorem 4.3.19, $P(\mathfrak{M}(D^\sigma) = \omega) = P(\mathfrak{M}(D^{\sigma'} = \omega)$ and consequently

$$\frac{P(\text{data} = D^\sigma \mid \mathfrak{M}(\text{data}) = \omega)}{P(\text{data} = D^{\sigma'} \mid \mathfrak{M}(\text{data}) = \omega)} = \frac{P(\text{data} = D^\sigma)}{P(\text{data} = D^{\sigma'})}$$

Step 3: show that parity is protected when attacker knows the true assignment σ.

First, note that the condition:

$$\begin{aligned} &P\left(\text{parity}\left(\begin{smallmatrix}\text{Individuals}\\ \text{who did not}\\ \text{participate in survey}\end{smallmatrix}\right) = 0 \mid \mathfrak{M}_{\text{drop}(p)}(\text{data}) = \omega\right) \\ &\geq P\left(\text{parity}\left(\begin{smallmatrix}\text{Individuals}\\ \text{who did not}\\ \text{participate in survey}\end{smallmatrix}\right) = 1 \mid \mathfrak{M}_{\text{drop}(p)}(\text{data}) = \omega\right) \end{aligned} \tag{4.25}$$

is exactly equivalent to the following condition (after subtracting the right hand side from the left hand side and plugging in the relevant probabilities):

$$\begin{aligned} &\forall \omega \in \text{range}(\mathfrak{M}) : \\ &\sum_{D_j \subseteq D^\sigma} \frac{P(\mathfrak{M}(D_j) = \omega)\,(-(1-q))^{\text{miss}(D_j)}\, q^{\text{nonblank}(D_j)}}{P(\mathfrak{M}(\text{data}) = \omega)} \geq 0 \end{aligned} \tag{4.26}$$

where the notation $D_j \subseteq D^\sigma$ means that D^σ can be converted to D_j by replacing some tuples with "?", nonblank(D_j) is the number of tuples in D_j not equal to "?", and miss(D_j) is the number of tuples in D_j equal to "?". Note that the terms corresponding to D_j such that $D_j \not\subseteq D^\sigma$ have been dropped because they are multiplied by a 0 prior.

Multiplying by $P(\mathfrak{M}(\text{data}) = \omega)$, dividing by q^W (size of population), and noting that $q^{\text{nonblank}(D_j)}/q^W = q^{-miss(D_j)}$, our goal is now to prove:

$$\begin{aligned} &\forall \omega \in \text{range}(\mathfrak{M}) : \\ &\sum_{D_j \subseteq D^\sigma} P(\mathfrak{M}(D_j) = \omega) \left(-\frac{1-q}{q}\right)^{\text{miss}(D_j)} \geq 0 \end{aligned} \tag{4.27}$$

for any $\mathfrak{M}$ in the rowcone of $\mathfrak{M}_{\text{sample}(p)}$.

[5] i.e. the record belonging to individual i is $\sigma'(i)$.

By Lemma 4.3.18,

$$\begin{aligned}\mathrm{CNF}(\{\mathfrak{M}_{\mathrm{sample}(p)}\}) &= \mathrm{CNF}(\{\mathfrak{M}_{\mathrm{sort}} \circ \mathfrak{M}_{\mathrm{drop}(p)}\}) \\ &\subseteq \mathrm{CNF}(\{\mathfrak{M}_{\mathrm{drop}(p)}\})\end{aligned}$$

By Corollary 4.2.3, $\mathfrak{M}$ is therefore of the form $\mathcal{A} \circ \mathfrak{M}_{\mathrm{drop}(p)}$ for some postprocessing algorithm $\mathcal{A}$. This means that every row of M (the matrix representation of $\mathfrak{M}$) is nonnegative linear combination of the rows of $M_{\mathrm{drop}(p)}$ (the matrix representation of $\mathfrak{M}_{\mathrm{drop}(p)}$). Thus all we need to do is to show that:

$$\begin{gathered}\forall \omega \in \mathrm{range}(\mathfrak{M}_{\mathrm{drop}(p)}) : \\ \sum_{D_j \subseteq D^\sigma} P(\mathfrak{M}_{\mathrm{drop}(p)}(D_j) = \omega) \left(-\frac{1-q}{q}\right)^{\mathrm{miss}(D_j)} \geq 0\end{gathered} \tag{4.28}$$

because every inequality (one for each $\omega \in \mathrm{range}(\mathfrak{M})$) in Equation 4.27 is a nonnegative linear combination of the inequalities in Equation 4.28 since the rows in the matrix representation of $\mathfrak{M}$ are nonnegative linear combinations of the rows of the matrix representation of $\mathfrak{M}_{\mathrm{drop}(p)}$.

Now consider the vector:

$$(y_1, \ldots, y_n)^T$$

where $y_j = \left(-\frac{1-q}{q}\right)^{\mathrm{miss}(D_j)}$ if $D_j \subseteq D_\sigma$ and $y_j = 0$ if $D_j \not\subseteq D_\sigma$. Then the conditions in Equation 4.28 are equivalent to:

$$M_{\mathrm{drop}(p)} \cdot (y_1, \ldots, y_n)^T \succeq 0 \tag{4.29}$$

where $M_{\mathrm{drop}(p)}$ is the matrix representation of $\mathfrak{M}_{\mathrm{drop}(p)}$ and the notation $\vec{v} \succeq 0$ means that all of the components of $\vec{v}$ are nonnegative.

Now, it is easy to see that:

$$\begin{aligned} y &= (y_1, \ldots, y_n) \\ &= \bigoplus_{i=1}^{W} C_i \end{aligned}$$

where the C_i are column vectors defined as:

$$j^{\mathrm{th}} \text{ entry of } C_i = \begin{cases} 1 & \text{if } \sigma(i) = r_j \in \mathcal{TUP} \\ -\frac{1-q}{q} & \text{if } j = N+1 \end{cases}$$

Recall $\sigma(i)$ is the assignment of a record to individual i (in this step we assume that σ is known to the attacker and we remove this assumption in Step 4). Recall also that $\mathcal{TUP} = \{r_1, \ldots, r_N\}$ is the domain of possible record values and that tuples come from the set $\mathcal{TUP} \cup \{?\}$, where "?"

represents a missing value (and which follows all the r_i in the ordering of the domain). The values in $\mathcal{TUP}$ are ordered only for the purposes of expressing our algorithms as matrices[6].

Recalling the equation for $M_{\text{drop}(p)}$, the matrix representation of $\mathfrak{M}_{\text{drop}(p)}$ as the Kronecker product $M_{\text{drop}(p)} = \bigoplus_{i=1}^{W} B_p$ where B_p is defined in Equation 4.24, Equation 4.29 is equivalent to:

$$0 \preceq \left(\bigoplus_{i=1}^{W} B_p\right)\left(\bigoplus_{i=1}^{W} C_i\right) = \bigoplus_{i=1}^{W}(B_p C_i) \tag{4.30}$$

And thus our goal is to prove that the components of $B_p C_i$ are nonnegative. Considering the rows of B_p, we see that the dot product between row $j < N+1$ of B_p and the vector C_i is either p or 0. The dot product between row $N+1$ of B_p and the vector C_i is $(1-p) - \frac{1-q}{q}$. This quantity is nonnegative as long as $q \geq \frac{1}{2-p}$. Thus for this setting of q, Equation 4.30 is true, which implies Equation 4.29 is true, which implies Equation 4.28, which implies Equation 4.27, which implies Equation 4.25, which is what we needed to prove.

Step 4: show that parity is protected when attacker does not know the true assignment σ. When the attacker does not know the true assignment, there is a probability distribution over assignments. In this case, the difference in posterior probability of the parity is equivalent to a nonnegative linear combination of Equation 4.26 (where we vary $D^{\sigma'}$ through all possible assignments σ' of record values to individuals (as long as the assignments are consistent with the attacker's background knowledge) and the weights of the nonnegative combination are $P(\text{data} = D^{sigma'})$). Thus the difference in posterior probability of the parity is a nonnegative linear combination of nonnegative quantities (by Step 3) and therefore is nonnegative. □

[6]Thus we need an order on columns, which correspond to datasets, and the order on datasets is induced by the order on the tuples, for example see Figure 4.2.

Chapter 5

Information Measures in Statistical Privacy

5.1 Introduction

When a data owner wants to share data with third parties, the data owner must often perturb the data with the goal of filtering out sensitive information while minimizing the damage to useful statistical information.

The data owner would first choose a privacy definition, which specifies a set of algorithms[1] that are trusted to adequately filter out the sensitive information. These algorithms are known as *data sanitizers* (or *privacy mechanisms*). From this set, the data owner must then choose a data sanitizer that best preserves useful statistical information.

The role of a *utility measure* is to help the data owner make such a choice. It assigns a numerical score to each possible data sanitizer, which reflects the ability of the data sanitizer to preserve statistical information that is useful for a given application. The data owner then uses the chosen data sanitizer to perturb the data and then releases the resulting *sanitized data*, denoted by ω, to third parties.

In this chapter, we study measures of information preservation for sanitizing algorithms and methods for processing the output of these algorithms.

5.1.1 Problematic Measures

There have been many proposals for measuring utility (e.g., [11, 29, 13, 14, 66]). Which of these are suitable for measuring the amount of information preserved by a sanitizing algorithm? It is not always intuitive, and so this question must be answered using scientific criteria [57, 18]. Consider the following example.

[1] Note that this set is often specified implicitly through constraints that the algorithms must satisfy.

Example 5.1.1. *A dataset contains responses to a true/false survey question. The data owner must choose between the following two sanitizing algorithms (both of which satisfy her privacy requirements):*

Output \ Input	True	False
True	0.75	0.5
False	0.25	0.5

Algorithm $\mathfrak{M}_1$

Output \ Input	True	False
True	0.6	0.4
False	0.4	0.6

Algorithm $\mathfrak{M}_2$

Algorithm $\mathfrak{M}_1$ changes a "true" to "false" with probability 0.25 and changes "false" to "true" with probability 0.5. Algorithm $\mathfrak{M}_2$ changes "true" to "false" and vice versa with probability 0.4. One popular utility measure μ is the negative of the worst-case error of a sanitizing algorithm. The maximum probability of error of $\mathfrak{M}_1$ is 0.5 (which occurs when the input is "false") and so its utility is $\mu(\mathfrak{M}_1) = -0.5$. The maximum probability of error of $\mathfrak{M}_2$ is 0.4 and so $\mu(\mathfrak{M}_2) = -0.4$. Therefore $\mathfrak{M}_2$ has the higher utility score (-0.4 vs. -0.5) and would be chosen over $\mathfrak{M}_1$.

However, we claim that $\mathfrak{M}_2$ preserves less information than $\mathfrak{M}_1$. To see why, consider an algorithm $\mathcal{A}$ that changes a "true" to a "false" with probability 0.2 and never changes a "false" to "true". Then running $\mathfrak{M}_2$ on the data is equivalent to first running $\mathfrak{M}_1$ and then running $\mathcal{A}$ on the result (i.e. $\mathfrak{M}_2 = \mathcal{A} \circ \mathfrak{M}_1$). Therefore if the output of $\mathfrak{M}_2$ can be used for an application then so can the output of $\mathfrak{M}_1$ (because we can always simulate $\mathfrak{M}_2$ by running $\mathcal{A}$ on the output of $\mathfrak{M}_1$). Thus in the target applications of this thesis, $\mathfrak{M}_1$ should be preferred over $\mathfrak{M}_2$ and so worst case error is not suitable as a measure of information preservation.

Sufficiency, a fundamental property [18] for candidate measures μ of information preservation, states that $\mu(\mathfrak{M}_2) \leq \mu(\mathfrak{M}_1)$ whenever $\mathfrak{M}_2 = \mathcal{A} \circ \mathfrak{M}_1$ for some randomized algorithm $\mathcal{A}$. That is, if $\mathfrak{M}_2$ can be simulated by running $\mathcal{A}$ on the output of $\mathfrak{M}_1$, then $\mathfrak{M}_1$ provides more information than $\mathfrak{M}_2$. Kifer and Lin [18] gave *anecdotal* examples of utility measures that violate this property. We propose two new axioms, quasi-convexity and quasi-concavity, for information preservation and *systematically* analyze many utility measures with respect to sufficiency and quasi-convexity/concavity.

We also consider a natural way of converting utility measures that do not satisfy sufficiency into related measures that do. We perform this procedure on the worst-case error metrics to obtain utility measures that have interpretations in terms of minimizing the error of certain Bayesian decision makers. We also explore how this interpretation creates a tension between the intuitively appealing concepts of worst-case analysis and quasi-convexity.

We note that in other contexts and applications, it can be necessary for a utility measure ν to violate sufficiency or other axioms. For example, it may be the case that $\mathfrak{M}_2 = \mathcal{A} \circ \mathfrak{M}_1$ yet $\nu(\mathfrak{M}_2) > \nu(\mathfrak{M}_1)$. This can happen when $\mathfrak{M}_1$ or $\mathcal{A}$ are computationally expensive or difficult to implement (hence the data owner or user could prefer $\mathfrak{M}_2$); users might also prefer $\mathfrak{M}_2$ if its output is already in a format that they can directly use (or can be put into that format without too much effort), such as a model parameter. We refer to these concerns collectively as *usability*. More work needs to be done to quantify such tradeoffs between information preservation and

usability (it is outside the scope of this thesis).

5.1.2 Axiomatically Justified Measures

Can any information preservation measures for a sanitizing algorithm be justified from first principles? We analyzed three utility axioms presented by [18] and derived the following result: those axioms imply that the ability of a data sanitizer $\mathfrak{M}$ to preserve information should be measured as the average (over possible outputs of $\mathfrak{M}$) error of a Bayesian decision maker. This result is important because it provides support for a rarely used utility measure (we are only aware of its use by Alvim et al. [12, 13]).

Note that the axioms themselves are not Bayesian: they do not mention subjective probabilities nor Bayes' rule. They are also different from the types of axioms used in statistics and economics to justify Bayesian decision theory (e.g., [58, 59, 60, 61]). Thus, although such a link should be expected to exist, it was not obvious that these particular axioms would provide the formal link between sanitizing algorithms and decision theory.

5.1.3 Summary of Contributions

- We propose two new utility axioms called *quasi-convexity* and *quasi-concavity* that relate random choices of sanitizers to deterministic choices.
- We show that the average (over possible outputs of a sanitizer $\mathfrak{M}$) error of a Bayesian decision maker is an axiomatically justified measure of the ability of a data sanitizer $\mathfrak{M}$ to preserve information.
- We analyze the information-preserving properties of many existing utility measures. Many violate the sufficiency property (as in Example 5.1.1) and quasi-convexity, but do satisfy quasi-concavity.
- We study a natural procedure, called *sufficing*, for repairing a utility measure's violation of sufficiency. We apply sufficing to measures such as worst-case error. The results show additional connections to Bayesian decision theory. We discuss how these results raise open question about the relationship between information and risk-averseness (i.e., quantification of utility in terms of worst-case scenarios).

Open problems. We leave open the question of general methods for computing measures of information preservation and optimizing algorithms based on them. This is a long-standing open problem for almost all utility measures; currently almost all utility results in the literature depend on highly inefficient/intractable algorithms or customized hand-crafted analyses [9, 10, 11, 12, 13, 14]. Another open problem is quantifying the tradeoff between usability and information preservation.

5.2 Information-Preserving Properties of Existing Utility Measures

We view Axioms 3.2.1 of sufficiency, 3.3.2 of quasi-convexity, and 3.3.3 of quasi-concavity as more fundamental than Axiom 3.2.4 of branching because the latter is more complex. Thus in this section, we analyze several popular utility measures for their information-preserving properties with respect to Axioms 3.2.1, 3.3.2, and 3.3.3 (the other axioms are studied in Section 5.3).

5.2.1 Expected and Worst-case Discrepancy

The two most popular type of utility measures compute the expected discrepancy and worst-case discrepancy between the input and output of a data sanitizer $\mathfrak{M}$.

The components of these utility measures consist of a fixed set $\mathbb{O}^*$ of possible outputs, a probability distribution $P(D)$ over possible input datasets $D \in \mathbb{I}$ and either a quality function Q or an error function E. When using quality functions, $Q(D, \omega)$ is the benefit of releasing sanitized output $\omega \in \mathbb{O}^*$ when the true input dataset is D (the higher the better). When using error functions, $E(D, \omega)$ is the dissimilarity between D and ω (such as a squared distance between queries computed over D and queries computed over ω).

Expected discrepancy can be used as a loss measure by defining it as the expected value of E (with respect to the prior P and randomness in the sanitizing algorithm) and as a utility measure by defining it as the expected value of Q:

Definition 5.2.1. (Expected discrepancy). *Given an input domain $\mathbb{I}$, fixed output domain $\mathbb{O}^*$, a quality function $Q : \mathbb{I} \times \mathbb{O}^* \to \mathbb{R}$ (for utility) or error function $E : \mathbb{I} \times \mathbb{O}^* \to \mathbb{R}$ (for loss), and a prior distribution P over $\mathbb{I}$, define the following utility ${\mu_{\mathbb{I}}}^{SE}$ and loss ${\mathcal{L}_{\mathbb{I}}}^{SE}$ measures (defined for all $\mathfrak{M}$ with* $\text{range}(\mathfrak{M}) \subseteq \mathbb{O}^*$*) as follows:*

$$\begin{aligned} {\mu_{\mathbb{I}}}^{SE}(\mathfrak{M}) &= \sum_{D \in \mathbb{I}} \sum_{\omega \in \mathbb{O}^*} Q(D, \omega)\ P(D)\ P(\mathfrak{M}(D) = \omega) \\ {\mathcal{L}_{\mathbb{I}}}^{SE}(\mathfrak{M}) &= \sum_{D \in \mathbb{I}} \sum_{\omega \in \mathbb{O}^*} E(D, \omega)\ P(D)\ P(\mathfrak{M}(D) = \omega) \end{aligned}$$

On the other hand, instead of averaging over all inputs, one can simply drop the prior and consider the input that leads to the worst-case expected discrepancy (here the expectation is taken with respect to the randomness of a sanitizing algorithm, but not with respect to possible priors).

Definition 5.2.2. (Worst-case discrepancy). *Given an input domain $\mathbb{I}$, fixed output domain $\mathbb{O}^*$, a quality function $Q : \mathbb{I} \times \mathbb{O}^* \to \mathbb{R}$ (for utility) or error function $E : \mathbb{I} \times \mathbb{O}^* \to \mathbb{R}$ (for loss), define the utility ${\mu_{\mathbb{I}}}^{SM}$ and loss ${\mathcal{L}_{\mathbb{I}}}^{SM}$ measures (defined for all $\mathfrak{M}$ with* $\text{range}(\mathfrak{M}) \subseteq \mathbb{O}^*$*) as follows:*

$${\mu_{\mathbb{I}}}^{SM}(\mathfrak{M}) = \inf_{D \in \mathbb{I}} \sum_{\omega \in \mathbb{O}^*} Q(D, \omega)\ P(\mathfrak{M}(D) = \omega) \tag{5.1}$$

$$\mathcal{L}_{\mathbb{I}}{}^{SM}(\mathfrak{M}) \quad = \quad \sup_{D \in \mathbb{I}} \sum_{\omega \in \mathbb{O}^*} E(D, \omega) \; P(\mathfrak{M}(D) = \omega)$$

For example, if each $\omega \in \mathbb{O}^*$ is interpreted as a query answer, one can set $E(D, \omega)$ to be 0 if the noisy answer ω has absolute error at most δ (i.e. the distance between the ω and the query answer computed from D is at most δ, for some pre-specified δ); set $E(D, \omega) = 1$ otherwise. The worst-case discrepancy then measures the worst-case (over all datasets) probability that $\mathfrak{M}$ did not answer the query accurately (i.e. with error at most δ).

5.2.1.1 Conditions for Satisfying the Axioms

Theorems 5.2.3 and 5.2.4 show that no nontrivial instantiations of expected and worst-case discrepancy measures can satisfy the sufficiency axiom.

Theorem 5.2.3. *A utility $\mu_{\mathbb{I}}{}^{SE}$ or loss $\mathcal{L}_{\mathbb{I}}{}^{SE}$ measure (as described in Definition 5.2.1) satisfies Axiom 3.2.1 (sufficiency) only if it is a constant function.*

Proof. See Section 5.6.1. □

Theorem 5.2.4. *Let $\mathbb{I}$ be finite. A utility $\mu_{\mathbb{I}}{}^{SM}$ or loss $\mathcal{L}_{\mathbb{I}}{}^{SM}$ measure (as described in Definition 5.2.2) satisfies Axiom 3.2.1 (sufficiency) only if it is a constant function.*

Proof. See Section 5.6.2. □

Expected discrepancy always satisfies the axioms of quasi-convexity/concavity:

Theorem 5.2.5. *The expected discrepancy utility $\mu_{\mathbb{I}}{}^{SE}$ and loss $\mathcal{L}_{\mathbb{I}}{}^{SE}$ measures (as Definition 5.2.1) always satisfy Axioms 3.3.2 (quasi-convexity) and 3.3.3 (quasi-concavity).*

Proof. See Section 5.6.3. □

Worst-case discrepancy always satisfies quasi-concavity:

Theorem 5.2.6. *Let $\mathbb{I}$ be finite. The worst-case discrepancy utility $\mu_{\mathbb{I}}{}^{SM}$ and loss $\mathcal{L}_{\mathbb{I}}{}^{SM}$ measures (Definition 5.2.2) always satisfy Axiom 3.3.3 (quasi-concavity).*

Proof. See Section 5.6.4. □

However, worst-case discrepancy only satisfies quasi-convexity in very limited trivial situations.

Theorem 5.2.7. *Let $\mathbb{I}$ be finite. A worst-case discrepancy utility $\mu_{\mathbb{I}}{}^{SM}$ or loss $\mathcal{L}_{\mathbb{I}}{}^{SM}$ measure satisfies Axiom 3.3.2 (quasi-convexity) if and only if there exists a D^* such that $\mu_{\mathbb{I}}{}^{SM}(\mathfrak{M})$ only depends on the distribution $P(\mathfrak{M}(D^*))$ for all $\mathfrak{M}$ (and similarly, $\mathcal{L}_{\mathbb{I}}{}^{SM}$ only depends on $P(\mathfrak{M}(D^*))$). That is, it only depends on the behavior of $\mathfrak{M}$ on a fixed dataset D^*.*

Proof. See Section 5.6.5. □

Thus, overall, these are not strictly measures of information preservation; they trade some information loss for other factors (such as usability). Quantifying this tradeoff is an interesting direction for future work.

In Section 5.4 we discuss a generic strategy called *sufficing* for fixing violations of the sufficiency axiom for utility measures such as expected and worst-case discrepancy. Sufficing provides decision-theoretic semantics to both of these utility measures. Based on these semantics, we give a detailed explanation of why satisfying the axiom of quasi-convexity is especially important for worst-case discrepancy (Section 5.4.2)

5.2.2 Expected Error of the Most Likely Input

The next utility measure we analyze can satisfy Axiom 3.2.1 in some nontrivial situations. It was used by Askari et al. [14], who observed that a sanitized output ω can provide information about the original inputs via maximum likelihood inference. Askari et al. [14] originally used this observation to define privacy loss, but the same formula can be used to define a utility measure as well.

The components of this measure are a prior probability distribution $P(D)$ over $D \in \mathbb{I}$ and an error measure $E(D, \widehat{D})$ that expresses the dissimilarity between two datasets D and $\widehat{D}$. For example, $E(D, \widehat{D})$ could be 0 if $D = \widehat{D}$ and 1 otherwise. Another possibility is that $E(D, \widehat{D})$ could be the distance between a query answer computed from D and a query answer computed from $\widehat{D}$. Suppose a data sanitizer $\mathfrak{M}$ produces an output ω. The most likely input dataset is the one that maximizes $P(D)P(\mathfrak{M}(D) = \omega)$. Let us denote this most likely input as $\widehat{D}_{(\mathfrak{M},\omega)}$. The loss measure of a data sanitizer $\mathfrak{M}$, as defined by [14] is the expected error of $\widehat{D}_{(\mathfrak{M},\omega)}$:

Definition 5.2.8. (e.g., [14]). *Given an input domain $\mathbb{I}$ and error function $E : \mathbb{I} \times \mathbb{I} \to \mathbb{R}$. The loss ${\mathcal{L}_{\mathbb{I}}}^{MAP}(\mathfrak{M})$ is defined as*

$$\sum_{\omega \in \text{range}(\mathfrak{M})} \sum_{D \in \mathbb{I}} P(D)\ P(\mathfrak{M}(D) = \omega)\ E(D, \widehat{D}_{(\mathfrak{M},\omega)})$$

where $\widehat{D}_{(\mathfrak{M},\omega)} \equiv \arg\max_{D \in \mathbb{I}} P(D)\ P(\mathfrak{M}(D) = \omega)$ is the maximum a posteriori *dataset (i.e. has the highest posterior probability). For the corresponding utility measure ${\mu_{\mathbb{I}}}^{MAP}$ replace the error function E with a quality function Q.*

5.2.2.1 Conditions for Satisfying the Axioms

For some choices of error function E and quality function Q, the result is a nontrivial measure of information preservation.

Theorem 5.2.9. *The loss measure ${\mathcal{L}_{\mathbb{I}}}^{MAP}(\mathfrak{M})$ described in Definition 5.2.8 satisfies Axiom 3.2.1 if and only if the corresponding error function E has the form*

$$E(D, \widehat{D}) = \alpha H(D, \widehat{D}) + f(D)$$

where α is a nonnegative constant, f is an arbitrary function, and $H(D, \widehat{D}) = 0$ when $D = \widehat{D}$ and equals 1 otherwise. The condition for the utility measure ${\mu_{\mathbb{I}}}^{MAP}$ is that the corresponding quality function Q has the form

$$Q(D, \widehat{D}) = \alpha(1 - H(D, \widehat{D})) + f(D)$$

Proof. See Section 5.6.6. □

The "$+f(D)$" term in Theorem 5.2.9 only plays a trivial role. Its effect is to add a constant (i.e. the expected value $\sum_D f(D)P(D)$) to the utility/loss of every algorithm $\mathfrak{M}$. Similarly, the α is just a constant factor. Thus, without loss of generality, one can restrict attention to error functions E such that $E(D, \widehat{D}) = 1$ when $D \neq \widehat{D}$ and 0 otherwise. In these cases, ${\mathcal{L}_{\mathbb{I}}}^{\text{MAP}}(\mathfrak{M})$ is simply the probability of incorrectly guessing the true input.

Meanwhile, the axioms of quasi-convexity and quasi-concavity are always satisfied.

Theorem 5.2.10. *A utility ${\mu_{\mathbb{I}}}^{MAP}$ or loss measure ${\mathcal{L}_{\mathbb{I}}}^{MAP}$ (as described in Definition 5.2.8) always satisfies Axioms 3.3.2 (quasi-convexity) and 3.3.3 (quasi-concavity).*

Proof. See Section 5.6.7. □

5.2.3 Expected Error of Sampling the Posterior Distribution

Instead of using the expected error of the most likely dataset (as in Section 5.2.2), one obvious alternative is to consider the possible posterior distributions $P(\text{data} = D \mid \mathfrak{M}(\text{data}) = \omega)$ for each ω, and then measure the average expected error of a sample from the posterior distributions. This kind of measure can be useful for statistical procedures such as imputation and sampling-based inference [93] where samples from a posterior are used as surrogates for the data.

Thus, we need a prior $P(D)$ over $D \in \mathbb{I}$ and a quality function $Q(D, \widehat{D})$ (to obtain a utility measure) or an error function $E(D, \widehat{D})$ (to obtain a loss measure). After a data sanitizer $\mathfrak{M}$ produces an output ω, the posterior distribution of the data is:

$$P[\text{data} = D_i | \, \mathfrak{M}(\text{data}) = \omega] = \frac{P[D_i]P[\mathfrak{M}(D_i) = \omega]}{\sum\limits_{D^\star \in \mathbb{I}} P[D^\star]P[\mathfrak{M}(D^\star) = \omega]}$$

If D is the true input and ω is the output, the expected error of a sample from the posterior distribution is:

$$\frac{\sum_{\widehat{D} \in \mathbb{I}} \; E(D, \widehat{D}) \; P[\widehat{D}] \; P[\mathfrak{M}(\widehat{D}) = \omega]}{\sum_{D^\star \in \mathbb{I}} \; P[D^\star] \; P[\mathfrak{M}(D^\star) = \omega]}$$

Taking the expectation with respect to the true dataset D and output ω, we get the following utility measure:

Definition 5.2.11. *Given an input domain $\mathbb{I}$, a prior distribution $P(D)$ for $D \in \mathbb{I}$, and an error function $E : \mathbb{I} \times \mathbb{I} \to \mathbb{R}$, the loss measure ${\mathcal{L}_{\mathbb{I}}}^{BG}(\mathfrak{M})$ is defined as*

$$\sum_{\omega \in \text{range}(\mathfrak{M})} \frac{\sum_{D \in \mathbb{I}} \sum_{\widehat{D} \in \mathbb{I}} E(D, \widehat{D}) \times \begin{pmatrix} P[D] \;\; P[\mathfrak{M}(D)=\omega] \\ \times P[\widehat{D}] \;\; P[\mathfrak{M}(\widehat{D})=\omega] \end{pmatrix}}{\sum_{D^\star \in \mathbb{I}} \; P[D^\star] \; P[\mathfrak{M}(D^\star) = \omega]}$$

For the corresponding utility measure ${\mu_{\mathbb{I}}}^{BG}$ replace the error function E with a quality function Q.

5.2.3.1 Conditions for Satisfying the Axioms

Not all quality functions Q or error functions E result in utility ${\mu_{\mathbb{I}}}^{\text{BG}}$ and loss ${\mathcal{L}_{\mathbb{I}}}^{\text{BG}}$ measures that satisfy Axiom 3.2.1 of sufficiency. Theorem 5.2.12 precisely characterizes the appropriate E and Q. In conjunction with Lemma 5.2.13 and Theorem 5.2.14, the results show that E must be a squared error function (modulo an additive constant).

Theorem 5.2.12 makes the following assumption for convenience and clarity. It assumes that the given prior satisfies $P(D_i) > 0$ for all $D_i \in \mathbb{I}$. This assumption is not a severe limitation since any terms in the equation for ${\mathcal{L}_{\mathbb{I}}}^{\text{BG}}$ that contain a D (or $\widehat{D}$) that has prior probability $P(D) = 0$ (or $P(\widehat{D}) = 0$) will drop out of the summation since that term will be multiplied by 0. Thus the loss measure ${\mathcal{L}_{\mathbb{I}}}^{\text{BG}}$ depends only on the values of $E(D, \widehat{D})$ for D and $\widehat{D}$ that have nonzero probabilities. Without loss of generality, we can therefore assume that impossible datasets have already been excluded from $\mathbb{I}$.

Theorem 5.2.12. *Given a finite input domain $\mathbb{I}$, a prior with $P(D_i) > 0$ for all $D_i \in \mathbb{I}$ and an error function $E : \mathbb{I} \times \mathbb{I} \to \mathbb{R}$ the loss measure ${\mathcal{L}_{\mathbb{I}}}^{BG}$ (as described in Definition 5.2.11) satisfies Axiom 3.2.1 (sufficiency) if and only if*

$$E(D, \widehat{D}) = S(D, \widehat{D}) + f(D) + g(\widehat{D})$$

for some negative semidefinite[2] function S and arbitrary functions g and f.

Proof. See Section 5.6.8. □

The following result shows the functions f and g in Theorem 5.2.12 only change the loss measure by an additive constant.

Lemma 5.2.13. *Given a finite input domain $\mathbb{I}$ and a prior over $\mathbb{I}$, let ${\mathcal{L}_{\mathbb{I}}}^{(1)}$ be the expected error of the posterior distribution resulting from error function E and let ${\mathcal{L}_{\mathbb{I}}}^{(2)}$ be the corresponding loss measure resulting from error function $E(D, \widehat{D}) + f(D) + g(\widehat{D})$. Then there exists a constant c, depending only on f and g such that ${\mathcal{L}_{\mathbb{I}}}^{(2)}(\mathfrak{M}) = {\mathcal{L}_{\mathbb{I}}}^{(1)}(\mathfrak{M}) + c$ for all $\mathfrak{M}$.*

Proof. See Section 5.6.9. □

[2] A function S is negative semidefinite if for all functions h, we have $\sum_{D \in \mathbb{I}} \sum_{\widehat{D} \in \mathbb{I}} h(D) h(\widehat{D}) S(D, \widehat{D}) \leq 0$; we do not require S to be symmetric.

Theorem 5.2.12 and Lemma 5.2.13 now allow us to show the connection to least squares estimation. We say that a function $H : \mathbb{I} \times \mathbb{I} \to \mathbb{R}$ is a *Euclidean distance function* if there is some positive integer n and function $\phi : \mathbb{I} \to \mathbb{R}^n$ such that $H(D, \widehat{D}) = ||\phi(D) - \phi(\widehat{D})||_2^2$. Intuitively, H maps D and $\widehat{D}$ into points in a Euclidean space and returns the square of the distance between them. The next theorem shows that every loss measure ${\mathcal{L}_{\mathbb{I}}}^{\mathrm{BG}}$ (up to an additive constant) that satisfies Axiom 3.2.1 of sufficiency results from an error function E that is a Euclidean distance function.

Theorem 5.2.14. *Choose a finite input domain $\mathbb{I}$ and prior such that $P(D_i) > 0$ for all $D_i \in \mathbb{I}$. If E is a error function that is also a Euclidean distance function then the corresponding loss measure ${\mathcal{L}_{\mathbb{I}}}^{BG}$ satisfies Axiom 3.2.1. Conversely, let ${\mathcal{L}_{\mathbb{I}}}^{(1)}$ be the expected error of the posterior distribution resulting from some error function $E^{(1)}$. If ${\mathcal{L}_{\mathbb{I}}}^{(1)}$ satisfies Axiom 3.2.1, then there exists a constant c and a Euclidean distance function E such that ${\mathcal{L}_{\mathbb{I}}}^{(2)}$, defined as the expected error of the posterior distribution with respect to E, satisfies ${\mathcal{L}_{\mathbb{I}}}^{(1)}(\mathfrak{M}) = {\mathcal{L}_{\mathbb{I}}}^{(2)}(\mathfrak{M}) + c$ (for all $\mathfrak{M}$).*

Proof. See Section 5.6.10. □

Meanwhile, the axioms of quasi-convexity and quasi-concavity are always satisfied.

Theorem 5.2.15. *A measure ${\mathcal{L}_{\mathbb{I}}}^{BG}$ of the expected error of the posterior distribution (as described in Definition 5.2.11) always satisfies Axioms 3.3.2 (quasi-convexity) and 3.3.3 (quasi-concavity).*

Proof. See Section 5.6.11. □

5.3 Axiomatically Derived Measures

In this section, we present the next major contribution of this thesis. We show that any loss measure $\mathcal{L}_{\mathbb{I}}$ that satisfies Axiom 3.2.1 (sufficiency), Axiom 3.2.3 (continuity), and Axiom 3.2.4 (branching) is precisely the expected error (over all possible outputs of a data sanitizer $\mathfrak{M}$) of a Bayesian decision maker.

This class of loss measures is rarely used in statistical privacy (we are only aware of work by Alvim et al. [12, 13]). Our contribution is the result that these loss measures are precisely the ones that satisfy those 3 axioms. Without pre-supposing subjective probabilities it, strengthens the case for wider use of Bayesian methods in statistical privacy. In particular, this result motivated our algorithm for the sorted histogram problem (Chapter 7) since it implies that Bayesian decision theory is an appropriate methodology for dealing with the output produced by sanitizing algorithms (i.e. optimizing algorithms for information preservation becomes equivalent to optimizing the error of a Bayesian decision maker who will process the data).

We first review Bayesian decision theory and then we present our result.

5.3.1 Review of Bayesian Decision Theory, Specialized to Sanitized Data

Bayesian decision theory is usually presented in very general language because it can be applied to many different types of problems. Since the context of interest is sanitized data, we specialize our discussion to how a user would use Bayesian decision theory to process sanitized data. The components of Bayesian decision theory are thus:

- A set $\mathbb{I}$ of possible datasets.
- A prior probability distribution over $\mathbb{I}$.
- A set $\mathbb{A}$ of actions. This corresponds to the type of statistical analysis a user wants to perform. For example, if a user wants an estimate of an answer to a query, then $\mathbb{A}$ could be the set of possible query answers. The user's job is to select the best action $a \in \mathbb{A}$ based on uncertain/probabilistic information (e.g. the user's job is to estimate the answer to a query based on a noisy output from a data sanitizer).
- An error function $\mathcal{E}$. The value $\mathcal{E}(D, a)$ corresponds to the error suffered by a user who selects action $a \in \mathbb{A}$ when the true (unknown) data is D. In the cases of query answers, a would be a candidate answer to a query and $\mathcal{E}(D, a)$ could be the absolute difference between the query answer a and the query answer computed from D.
- An output ω produced by a data sanitizer $\mathfrak{M}$ that was run on the true data (which is unknown to the user).

The subjective expected error of selecting an action $a \in \mathbb{A}$ after observing sanitized data ω is

$$\begin{aligned} &\sum_{D \in \mathbb{I}} \mathcal{E}(D, a) P(\text{data} = D \mid \mathfrak{M}(\text{data}) = \omega) \\ &= \sum_{D \in \mathbb{I}} \mathcal{E}(D, a) \frac{P(\mathfrak{M}(D) = \omega) P(D)}{\sum_{D' \in \mathbb{I}} P(\mathfrak{M}(D') = \omega) P(D')} \end{aligned} \tag{5.2}$$

According to Bayesian decision theory, the user should choose an action that minimizes this expected error. Noting that the denominator is the same for all actions, this means that the optimal action a^* is:

$$a^* = \arg\min_{a \in \mathbb{A}} \sum_{D \in \mathbb{I}} \mathcal{E}(D, a) P(\mathfrak{M}(D) = \omega) P(D) \tag{5.3}$$

Other Bayesian and frequentist methodologies are also possible. However, our results in the next section indicate that choosing sanitizing algorithms to maximize information preservation is in some sense equivalent to choosing an algorithm whose outputs should be processed using Bayesian decision theory.

5.3.2 Expected Error of a Bayesian Decision Maker: an Axiomatically Justified Loss Measure

When a user choose an action $a \in \mathbb{A}$ (such as a specific estimate for query answers or model parameters) using Equation 5.3, the expected error of this action-selection strategy depends on a sanitizing algorithm $\mathfrak{M}$ and thus can be viewed as a loss measure ${\mathcal{L}_{\mathbb{I}}}^{\text{BDT}}$:

Definition 5.3.1. (Expected error of a Bayesian decision maker). *Given an input domain $\mathbb{I}$, a prior over $\mathbb{I}$, a set $\mathbb{A}$ of actions, and an error function $\mathcal{E} : \mathbb{I} \times \mathbb{A} \to \mathbb{R}$ that is bounded from below (i.e. for all D and all a, $\mathcal{E}(D, a) \geq \beta$ for some β) define the loss measure ${\mathcal{L}_{\mathbb{I}}}^{BDT}$ as:*

$$ {\mathcal{L}_{\mathbb{I}}}^{BDT}(\mathfrak{M}) = \sum_{\omega \in \text{range}(\mathfrak{M})} \inf_{a \in \mathbb{A}} \sum_{D \in \mathbb{I}} P(D) P(\mathfrak{M}(D) = \omega)\, \mathcal{E}(D, a) $$

The error function has to be bounded below so that ${\mathcal{L}_{\mathbb{I}}}^{\text{BDT}}$ is a real number (and not $-\infty$). Also note that we can get a utility measure ${\mu_{\mathbb{I}}}^{\text{BDT}}$ by taking the negative (i.e. $-{\mathcal{L}_{\mathbb{I}}}^{\text{BDT}}(\mathfrak{M})$).

The next theorem says that ${\mathcal{L}_{\mathbb{I}}}^{\text{BDT}}$ avoids information paradoxes (such as Example 5.1.1) by satisfying Axiom 3.2.1 for all priors and all $\mathcal{E}$ that are bounded from below. More importantly, it shows that any loss measure that satisfies Axioms 3.2.1, 3.2.3, and 3.2.4 is equivalent to ${\mathcal{L}_{\mathbb{I}}}^{\text{BDT}}$ for some set of actions $\mathbb{A}$, some error function $\mathcal{E}$ (that is bounded from below), and some prior distribution $P(D)$.

Theorem 5.3.2. *Let $\mathbb{I}$ be finite.*

(i) For every choice of $\mathbb{A}$, prior over $\mathbb{I}$, and error function $\mathcal{E}$ that is bounded from below, the resulting loss measure ${\mathcal{L}_{\mathbb{I}}}^{BDT}$ (Definition 5.3.1) satisfies Axioms 3.2.1, 3.2.3, and 3.2.4.

(ii) If a loss measure $\mathcal{L}_{\mathbb{I}}$ satisfies Axioms 3.2.1, 3.2.3, and 3.2.4 then there exists a prior over $\mathbb{I}$, a set of actions $\mathbb{A}$, and an error function $\mathcal{E}$ (that is bounded from below) such the corresponding loss measure ${\mathcal{L}_{\mathbb{I}}}^{BDT}$ equals $\mathcal{L}_{\mathbb{I}}$.

Proof. See Section 5.6.12. □

This class of utility/loss measures also satisfies the axioms of quasi-convexity and quasi-concavity.

Theorem 5.3.3. *A loss measure ${\mathcal{L}_{\mathbb{I}}}^{BDT}$ (as described in Definition 5.3.1) satisfies Axioms 3.3.2 (quasi-convexity) and 3.3.3 (quasi-concavity).*

Proof. See Section 5.6.13. □

The same results obviously hold for ${\mu_{\mathbb{I}}}^{\text{BDT}}$.

5.3.3 Discussion

Traditionally, after seeing a sanitized output ω, Bayesian decision theory defines the expected error as:

$$\inf_{a \in \mathbb{A}} \sum_{D \in \mathbb{I}} \mathcal{E}(D, a) \frac{P(\mathfrak{M}(D) = \omega) P(D)}{P(\omega)}$$

(where $P(\omega) = \sum_{D \in \mathbb{I}} P(D) P(\mathfrak{M}(D) = \omega)$). When we discuss expected error as a loss measure, we further must take the expectation with respect to ω as well (yielding the equation in Definition 5.3.1).

Note that measuring these quantities for sanitizing algorithms is generally a difficult problem as is optimizing algorithms based on these utility/loss measures. However, the same is true for almost all other types of utility and loss measures (the most common result is a hand-crafted asymptotic upper bound). Thus identifying tractable situations is an interesting area for future work.

We also expect that Bayesian methods for analyzing sanitized data will pose additional research challenges due to the type of noise that is added by the sanitization process [17]. We provide two pieces of evidence that research effort in this direction is worthwhile. The first is this theoretical argument that (according to several axioms) information preservation is the expected error of an analyst who uses Bayesian decision theory (we strengthen this argument in Section 5.4 by showing another connection to Bayesian decision theory that arises from fixing violations of the axiom of sufficiency for the worst-case/expected utility measures discussed in Section 5.2.1).

5.4 Fixing Violations of Sufficiency

In Section 5.2, we examined utility measures that satisfy the sufficiency axiom only in limited circumstances. We now discuss an operation, called *sufficing*, that can be used to fix violations of this axiom.

Definition 5.4.1. *The sufficing operation converts a utility measure $\mu_{\mathbb{I}}$ into a utility measure $\mu_{\mathbb{I}}^{\diamond}$ defined as $\mu_{\mathbb{I}}^{\diamond}(\mathfrak{M}) = \sup_{\mathcal{A}} \mu_{\mathbb{I}}(\mathcal{A} \circ \mathfrak{M})$, where the supremum is over all (possibly randomized) algorithms whose domain contains* $\text{range}(\mathfrak{M})$.

Intuitively, it looks for the best method to handle the output of $\mathfrak{M}$. Its usefulness comes from the following result, whose proof is trivial.

Lemma 5.4.2. *For any utility measure $\mu_{\mathbb{I}}$, sufficing produces a utility measure $\mu_{\mathbb{I}}^{\diamond}$ that always satisfies Axiom 3.2.1 (sufficiency). It is also the "smallest repair" of $\mu_{\mathbb{I}}$ in the following sense: if some utility measure $\mu_{\mathbb{I}}'$ satisfies Axiom 3.2.1 and $\mu_{\mathbb{I}}'(\mathfrak{M}) \geq \mu_{\mathbb{I}}(\mathfrak{M})$ for all $\mathfrak{M}$ then $\mu_{\mathbb{I}}'(\mathfrak{M}) \geq \mu_{\mathbb{I}}^{\diamond}(\mathfrak{M})$ for all $\mathfrak{M}$. As a result, $\mu_{\mathbb{I}} \equiv \mu_{\mathbb{I}}^{\diamond}$ if and only if $\mu_{\mathbb{I}}$ satisfies Axiom 3.2.1.*

Proof. See Section 5.6.14. □

In Sections 5.4.1 and 5.4.2 we examine how the sufficing transformation affects the expected and worst-case discrepancy measures from Section 5.2. For both utility measures, sufficing adds semantics related to the axiomatic utility measure that was derived in Section 5.3.2 (i.e. expected error of a Bayesian decision maker).

5.4.1 Sufficing Expected Discrepancy

Our first result connects the expected discrepancy (Definition 5.2.1) to the expected error of a Bayesian decision maker (Definition 5.3.1) – after sufficing, they become the same. One way to interpret this result is that the optimal post-processing algorithm $\mathcal{A}$ in Definition 5.4.1 (sufficing) re-assigns the outputs of $\mathfrak{M}$ based on decision theory; the outputs are re-assigned in a way that minimizes expected (with respect to the posterior distribution) error.

Theorem 5.4.3. *Given a prior P over $\mathbb{I}$, a fixed output domain $\mathbb{O}^*$, and a quality function $Q : \mathbb{I} \times \mathbb{O}^* \to \mathbb{R}$, let ${\mu_{\mathbb{I}}}^{SE}$ be the corresponding expected discrepancy utility function (Definition 5.2.1). Let ${\mu_{\mathbb{I}}}^{SE\diamond}$ be the utility function that results from sufficing. Then ${\mu_{\mathbb{I}}}^{SE\diamond}$ is the negative of the expected error of a Bayesian decision maker (Definition 5.3.1) with the same prior and quality function Q (or error function $\mathcal{E} = -Q$)*[3].

Proof. Recall that

$$ {\mu_{\mathbb{I}}}^{\mathrm{SE}}(\mathfrak{M}) \quad = \quad \sum_{D \in \mathbb{I}} \sum_{\omega \in \mathbb{O}^*} Q(D, \omega) P(D) P(\mathfrak{M}(D) = \omega) $$

and so

$$ \begin{aligned} {\mu_{\mathbb{I}}}^{\mathrm{SE}\diamond}(\mathfrak{M}) &= \sup_{\mathcal{A}} {\mu_{\mathbb{I}}}^{\mathrm{SE}}(\mathcal{A} \circ \mathfrak{M}) \\ &= \sup_{\mathcal{A}} \sum_{D \in \mathbb{I}} \sum_{\omega \in \mathbb{O}^*} Q(D, \omega) P(D) P(\mathcal{A}(\mathfrak{M}(D)) = \omega) \\ &= \sup_{\mathcal{A}} \sum_{D \in \mathbb{I}} \sum_{\nu \in \mathrm{range}(\mathfrak{M})} \sum_{\omega \in \mathbb{O}^*} Q(D, \omega)\ P(D) P(\mathfrak{M}(D) = \nu) P(\mathcal{A}(\nu) = \omega) \\ &= \sup_{\mathcal{A}} \sum_{\nu \in \mathrm{range}(\mathfrak{M})} \sum_{\omega \in \mathbb{O}^*} \left(\sum_{D \in \mathbb{I}} P(D) P(\mathfrak{M}(D) = \nu) Q(D, \omega) \right) P(\mathcal{A}(\nu) = \omega) \end{aligned} $$

It is easy to see that this results in a series of independent optimization problems, one for each $\nu_i \in \mathrm{range}(\mathfrak{M})$:

$$ \sup_{\substack{P(\mathcal{A}(\nu_i)=\omega_1) \\ P(\mathcal{A}(\nu_i)=\omega_2) \\ P(\mathcal{A}(\nu_i)=\omega_3) \\ \cdots}} \sum_{\omega \in \mathbb{O}^*} \left(\sum_{D \in \mathbb{I}} P(D) P(\mathfrak{M}(D) = \nu_i) Q(D, \omega) \right) P(\mathcal{A}(\nu_i) = \omega) $$

(for ν_i we need to choose values of $P(\mathcal{A}(\nu_i) = \omega_1), P(\mathcal{A}(\nu_i) = \omega_2), \ldots$ and the values we choose

[3]Note that the set $\mathbb{A}$ of actions is now the same as $\mathbb{O}^*$.

for ν_i are unrelated to the value we choose for ν_j when $j \neq i$). This subproblem has the optimal value $\sup_{\omega \in \mathbb{O}^*} \left(\sum_{D \in \mathbb{I}} P(D) P(\mathfrak{M}(D) = \nu_i) Q(D, \omega) \right)$. Thus

$$\begin{aligned} \mu_{\mathbb{I}}{}^{\mathrm{SE}\diamond}(\mathfrak{M}) &= \sup_{\mathcal{A}} \sum_{\nu \in \mathrm{range}(\mathfrak{M})} \sum_{\omega \in \mathbb{O}^*} \left(\sum_{D \in \mathbb{I}} P(D) P(\mathfrak{M}(D) = \nu) Q(D, \omega) \right) P(\mathcal{A}(\nu) = \omega) \\ &= \sum_{\nu \in \mathrm{range}(\mathfrak{M})} \sup_{\omega \in \mathbb{O}^*} \left(\sum_{D \in \mathbb{I}} P(D) P(\mathfrak{M}(D) = \nu) Q(D, \omega) \right) \end{aligned}$$

which is the negative of the expected error of a Bayesian decision maker with action set $\mathbb{O}^*$. □

The corresponding Bayesian decision maker from Theorem 5.4.3 has the same prior P and the same quality function Q with the best action that reassigns each output $\nu \in \mathbb{O}^*$ to some output $\omega \in \mathbb{O}^*$ such that ω maximizes the expected utility given the observation ν (with the likelihood vector that generates ν). Thus, we can use the expected error of a Bayesian decision maker as a replacement for expected discrepancy in applications.

5.4.2 Sufficing Worst-case Discrepancy

Worst-case discrepancy $\mu_{\mathbb{I}}{}^{\mathrm{SM}}$ (Definition 5.2.2) considers the input $D \in \mathbb{I}$ for which the outputs of $\mathfrak{M}$ are syntactically[4] most dis-similar from the input. As a result it does not satisfy the sufficiency axiom. After the sufficing transformation, the resulting utility measure $\mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}$ has the following form:

Theorem 5.4.4. *Let $|\mathbb{I}|$ be finite. Given a fixed output space $\mathbb{O}^*$ and a bounded[5] quality function $Q : \mathbb{I} \times \mathbb{O}^* \to \mathbb{R}$, let $\mu_{\mathbb{I}}{}^{SM}$ be the corresponding worst-case discrepancy utility measure. The sufficing operation produces the following utility measure:*

$$\mu_{\mathbb{I}}{}^{SM\diamond}(\mathfrak{M}) = \inf_{P} \mu_{\mathbb{I}}{}^{BDT}(\mathfrak{M}; P, Q)$$

where the infimum is over the set of all prior probability distributions over $\mathbb{I}$ and $\mu_{\mathbb{I}}{}^{BDT}(\mathfrak{M}; P, Q)$ is the utility measure corresponding to the negative expected error of a Bayesian decision maker (Definition 5.3.1) with prior P and quality function Q (i.e. error function $\mathcal{E} = -Q$).

Proof. Recall that

$$\mu_{\mathbb{I}}{}^{\mathrm{SM}}(\mathfrak{M}) = \inf_{D \in \mathbb{I}} \sum_{\omega \in \mathbb{O}*} Q(D, \omega) P(\mathfrak{M}(D) = \omega)$$

[4] That is, if $\mathfrak{M}$ outputs the value 1, the quality function treats it as a number rather than an uncertain quantity (where the uncertainty depends on the probabilities $P(\mathfrak{M}(D_i) = 1)$ for different D_i).

[5] So that $\mu_{\mathbb{I}}{}^{\mathrm{SM}}$ is finite for all $\mathfrak{M}$

Thus,

$$\begin{aligned}
\mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}) &= \sup_{\mathcal{A}} \mu_{\mathbb{I}}{}^{\mathrm{SM}}(\mathcal{A} \circ \mathfrak{M}) \\
&= \sup_{\mathcal{A}} \inf_{D \in \mathbb{I}} \sum_{\omega \in \mathbb{O}^*} Q(D, \omega) P(\mathcal{A}(\mathfrak{M}(D)) = \omega) \\
&= \sup_{\mathcal{A}} \inf_{P_\theta \in \mathfrak{P}} \sum_{D \in \mathbb{I}} \sum_{\omega \in \mathbb{O}^*} Q(D, \omega) P_\theta(D) P(\mathcal{A}(\mathfrak{M}(D)) = \omega) \\
&\quad \text{(where } \mathfrak{P} \text{ is the set of all priors over } \mathbb{I}\text{; the equality results from} \\
&\quad \text{the fact that this minimum occurs at a prior that puts probability} \\
&\quad \text{1 on some dataset)} \\
&= \sup_{\mathcal{A}} \inf_{P_\theta \in \mathfrak{P}} \sum_{D \in \mathbb{I}} \sum_{\omega \in \mathbb{O}^*} \sum_{\nu \in \mathrm{range}(\mathfrak{M})} Q(D, \omega) P_\theta(D) P(\mathfrak{M}(D) = \nu) P(\mathcal{A}(\nu) = \omega) \\
&= \inf_{P_\theta \in \mathfrak{P}} \sup_{\mathcal{A}} \sum_{D \in \mathbb{I}} \sum_{\omega \in \mathbb{O}^*} \sum_{\nu \in \mathrm{range}(\mathfrak{M})} Q(D, \omega) P_\theta(D) P(\mathfrak{M}(D) = \nu) P(\mathcal{A}(\nu) = \omega)
\end{aligned}$$

The interchange of inf and sup is allowed by Sion's theorem[6] since the function is linear in P_θ and $\mathcal{A}$ (due to the boundedness of Q), because the set of all priors is convex and compact (due to finiteness of $|\,\mathbb{I}\,|$) and due to the convexity of the set of randomized algorithms with output space $\mathbb{O}^*$ and domain containing range($\mathfrak{M}$).

Thus we have

$$\begin{aligned}
\mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}) &= \inf_{P_\theta \in \mathfrak{P}} \sup_{\mathcal{A}} \sum_{D \in \mathbb{I}} \sum_{\omega \in \mathbb{O}^*} \sum_{\nu \in \mathrm{range}(\mathfrak{M})} Q(D, \omega) P_\theta(D) P(\mathfrak{M}(D) = \nu) P(\mathcal{A}(\nu) = \omega) \\
&= \inf_{P_\theta \in \mathfrak{P}} \sup_{\mathcal{A}} \sum_{D \in \mathbb{I}} \sum_{\omega \in \mathbb{O}^*} Q(D, \omega) P_\theta(D) P(\mathcal{A}(\mathfrak{M}(D)) = \omega)
\end{aligned}$$

The inner supremum equals $\mu_{\mathbb{I}}{}^{\mathrm{BDT}}(\mathfrak{M}; P_\theta, Q)$ (as shown in the proof of Theorem 5.4.3).

□

Thus $\mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}$ is a utility measure that considers a set of Bayesian decision makers, all sharing the same quality function Q but having different priors. They all process the output of $\mathfrak{M}$ according to decision theory, and $\mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}$ measures how satisfied the least one of them is. These are nice semantics to have, and $\mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}$ satisfies Axiom 3.3.3 of quasi-concavity. However, it does not satisfy Axiom 3.3.2 of quasi-convexity.

Theorem 5.4.5. *Let $\mathbb{I}$ be finite and let the quality function Q be bounded. Then $\mu_{\mathbb{I}}{}^{SM\diamond}$ satisfies Axiom 3.3.3 (quasi-concavity).*

Proof. See Section 5.6.15. □

[6] Maurice Sion, "On general minimax theorems," *Pacific Journal of Mathematics*, 8:171–176, 1958.

Theorem 5.4.6. *Let $\mathbb{I}$ and $\mathbb{O}^*$ be finite and let the quality function $Q : \mathbb{I} \times \mathbb{O}^* \to \mathbb{R}$ be bounded. Then $\mu_{\mathbb{I}}^{SM\diamond}$ satisfies Axiom 3.3.2 (quasi-convexity) if and only if it is constant.*

Proof. See Section 5.6.16. □

Recall that worst-case discrepancy $\mu_{\mathbb{I}}^{\mathrm{SM}}$ (Definition 5.2.2) violated the fundamental axiom of sufficiency, and that sufficing was used to fix this violation by creating a related utility measure $\mu_{\mathbb{I}}^{\mathrm{SM}\diamond}$. Theorem 5.4.4 gave semantics to $\mu_{\mathbb{I}}^{\mathrm{SM}\diamond}$ in terms of the least-satisfied Bayesian decision maker. Using these semantics, we now explain why the failure of $\mu_{\mathbb{I}}^{\mathrm{SM}\diamond}$ to satisfy the axiom of quasi-convexity is significant. Consider the following two algorithms $\mathfrak{M}_1$ and $\mathfrak{M}_2$ along with the quality function Q.

	D_1	D_2
ω_1	.8	.6
ω_2	.2	0
ω_3	0	.4

Alg. $\mathfrak{M}_1$

	D_1	D_2
ω_1	.6	.8
ω_2	0	.2
ω_3	.4	0

Alg. $\mathfrak{M}_2$

	D_1	D_2
D_1	1	0
D_2	0	1

Q

By definition, $\mu_{\mathbb{I}}^{\mathrm{SM}\diamond}(\mathfrak{M}) = \sup_{\mathcal{A}} \mu_{\mathbb{I}}^{\mathrm{SM}}(\mathcal{A}_1 \circ \mathfrak{M}_1)$ and a simple linear program shows that $\mu_{\mathbb{I}}^{\mathrm{SM}\diamond}(\mathfrak{M}) = \mu_{\mathbb{I}}^{\mathrm{SM}}(\mathcal{A}_1 \circ \mathfrak{M}_1)$ for the following algorithm $\mathcal{A}_1$ and resulting algorithm $\mathcal{A}_1 \circ \mathfrak{M}_1$.

	ω_1	ω_2	ω_3
D_1	4/7	1	0
D_2	3/7	0	1

Alg. $\mathcal{A}_1$

	D_1	D_2
D_1	46/70	24/70
D_2	24/70	46/70

Alg. $\mathcal{A}_1 \circ \mathfrak{M}_1$

Using the quality function Q, when the input is D_1 the expected quality of $\mathcal{A}_1 \circ \mathfrak{M}_1$ is 23/35 and when the input is D_2, the expected quality is also 23/35. Therefore $\mu_{\mathbb{I}}^{\mathrm{SM}}(\mathcal{A}_1 \circ \mathfrak{M}_1) = 23/35$ and thus $\mu_{\mathbb{I}}^{\mathrm{SM}\diamond}(\mathfrak{M}_1) = 23/35$. By symmetry, $\mu_{\mathbb{I}}^{\mathrm{SM}\diamond}(\mathfrak{M}_2) = 23/35$. Next we consider the algorithm $\mathfrak{M}_1 \oplus_{0.5} \mathfrak{M}_2$ with the $\mathcal{A}^*$ for which $\mu_{\mathbb{I}}^{\mathrm{SM}\diamond}(\mathfrak{M}_1 \oplus_{0.5} \mathfrak{M}_2) = \mu_{\mathbb{I}}^{\mathrm{SM}}(\mathcal{A}^* \circ (\mathfrak{M}_1 \oplus_{0.5} \mathfrak{M}_2))$ ($\mathcal{A}^*$ is also obtained via a simple linear program).

	D_1	D_2
ω_1 from $\mathfrak{M}_1$	.4	.3
ω_2 from $\mathfrak{M}_1$	.1	0
ω_3 from $\mathfrak{M}_1$	0	.2
ω_1 from $\mathfrak{M}_2$	.3	.4
ω_2 from $\mathfrak{M}_2$	0	.1
ω_3 from $\mathfrak{M}_2$	.2	0

Alg. $\mathfrak{M}_1 \oplus_{0.5} \mathfrak{M}_2$

	ω_1, $\mathfrak{M}_1$	ω_2, $\mathfrak{M}_1$	ω_3, $\mathfrak{M}_1$	ω_1, $\mathfrak{M}_2$	ω_2, $\mathfrak{M}_2$	ω_3, $\mathfrak{M}_2$
D_1	1	1	0	0	0	1
D_2	0	0	1	1	1	0

Alg. $\mathcal{A}^*$

	D_1	D_2
D_1	.7	.3
D_2	.3	.7

Alg. $\mathcal{A}^* \circ (\mathfrak{M}_1 \oplus_{0.5} \mathfrak{M}_2)$

The expected quality of $\mathcal{A}^* \circ (\mathfrak{M}_1 \oplus_{0.5} \mathfrak{M}_2)$ is 0.7 when the input is D_1 or D_2 and so $\mu_{\mathbb{I}}^{\mathrm{SM}\diamond}(\mathfrak{M}_1 \oplus_{0.5} \mathfrak{M}_2) = 0.7 > 23/35 = \mu_{\mathbb{I}}^{\mathrm{SM}\diamond}(\mathfrak{M}_1) = \mu_{\mathbb{I}}^{\mathrm{SM}\diamond}(\mathfrak{M}_2)$. Thus we see that quasi-convexity is violated.

Now let us examine this from a Bayesian perspective with the help of Theorem 5.4.4.

Consider an analyst $\mathfrak{B}_1$ who believes $P(\text{data} = D_1) = 3/7$ and $P(\text{data} = D_2) = 4/7$. The set

$\mathbb{A}$ of actions is $\{D_1, D_2\}$ (e.g., run analysis software on D_1 or run it on D_2). Upon seeing output ω_1 from $\mathfrak{M}_1$, $\mathfrak{B}_1$ must choose between action D_1 (with average quality $P(\text{data} = D_1)P(\mathfrak{M}(D_1) = \omega_1)Q(D_1, D_1) + P(\text{data} = D_2)P(\mathfrak{M}(D_2) = \omega_1)Q(D_2, D_1) = 12/35$) and D_2 (with average quality also $12/35$). Since the average quality is the same, analyst $\mathfrak{B}_1$ can choose either one (say D_1). Upon seeing output ω_2 from $\mathfrak{M}_1$, the analyst will choose action D_1 with average quality $3/35$ and upon seeing output ω_3 from $\mathfrak{M}_1$, the analyst will choose action D_2 with average quality $8/35$. Overall, the expected quality of analyst $\mathfrak{B}_1$ for algorithm $\mathfrak{M}_1$ is $12/35 + 3/35 + 8/35 = 23/35$; in fact, this analyst's prior will yield the lowest expected quality for $\mathfrak{M}_1$ among all priors. For analyst $\mathfrak{B}_1$, the expected quality of $\mathfrak{M}_2$ is $26/35$.

Now consider analyst $\mathfrak{B}_2$ whose believes $P(\text{data} = D_1) = 4/7$ and $P(\text{data} = D_2) = 3/7$. For analyst $\mathfrak{B}_2$, the expected quality of $\mathfrak{M}_1$ is $26/35$ and the expected quality of $\mathfrak{M}_2 = 23/35$; in fact $\mathfrak{B}_2$ has the prior that obtains the lowest expected quality for $\mathfrak{M}_2$ among all priors.

To summarize, if we run $\mathfrak{M}_1$ on the data, $\mathfrak{B}_1$ is the least happy among all decision makers and is the reason why $\mu_{\mathbb{I}}{}^{\text{SM}\diamond}(\mathfrak{M}_1) = 23/35$; if we run $\mathfrak{M}_2$ on the data, $\mathfrak{B}_2$ is the least happy analyst and is the reason why $\mu_{\mathbb{I}}{}^{\text{SM}\diamond}(\mathfrak{M}_2) = 23/35$. What happens if we run $\mathfrak{M}_1 \oplus_{0.5} \mathfrak{M}_2$? This is the same as randomly choosing between $\mathfrak{M}_1$ and $\mathfrak{M}_2$, announcing the choice, and running the chosen algorithm on the data. Clearly, one of the analysts $\mathfrak{B}_1$ or $\mathfrak{B}_2$ will always be unhappy with the chosen algorithm (although we can't predict ahead of time which one it will be), yet this randomized choice has higher utility $\mu_{\mathbb{I}}{}^{\text{SM}\diamond}(\mathfrak{M}_1 \oplus_{0.5} \mathfrak{M}_2) = 0.7 > 23/35$.

This example of tension between the axiom of quasi-convexity and utility measures based on worst-case scenarios raises (possibly philosophical) open questions about the relative importance of worst-case analysis, quasi-convexity, and how the two intuitively appealing ideas should be reconciled.

5.5 Conclusions

In this chapter we examined generalizations of many existing approaches for measuring utility and identified conditions under which they satisfy necessary utility axioms. Our principle contributions are this analysis and the identification of an equivalence between branching measures and ER guessing measures, thus identifying usable utility measures, providing a statistical interpretation to a set of utility axioms, and providing a principle for processing sanitized data. In Chapter 6, empirical results show the benefit of this approach. In additions, we study a natural procedure, called sufficing, to fix violation of sufficiency. Applying sufficing to expected discrepancy results in the expected error of Bayesian decision maker. Applying sufficing to worst-case discrepancy results in expected error of least-satisfied Bayesian decision maker (i.e. worst-case (among priors) of expected error of Bayesian decision makers). This, however, does not satisfy Axiom 3.3.2 (quasi-convexity).

5.6 Proofs

5.6.1 Proof of Theorem 5.2.3

Theorem 5.2.3. *A utility ${\mu_{\mathbb{I}}}^{SE}$ or loss ${\mathcal{L}_{\mathbb{I}}}^{SE}$ measure (as described in Definition 5.2.1) satisfies Axiom 3.2.1 (sufficiency) only if it is a constant function.*

Proof. We will show that ${\mu_{\mathbb{I}}}^{\mathrm{SE}}$ satisfies Axiom 3.2.1 (sufficiency) if and only if:

$$\begin{aligned} Q(D,\omega_1) &= Q(D,\omega_2),\ \forall \omega_1,\omega_2 \in \mathrm{range}(\mathfrak{M}), \\ &\quad \forall D \in \mathbb{I} \text{ with } P(D) > 0 \end{aligned} \tag{5.4}$$

It is clear that such a Q implies a constant ${\mu_{\mathbb{I}}}^{\mathrm{SE}}$.

By way of contradiction, suppose that there exist $\omega_1, \omega_2 \in \mathrm{range}(\mathfrak{M})$ and a $D \in \mathbb{I}$ such that $Q(D,\omega_1) < Q(D,\omega_2)$ and $P(D) > 0$. Choose any p such that $1 > p > 0$. Let $\mathfrak{M}_1$ and $\mathfrak{M}_2$ behave as follows.

$$\begin{aligned} P(\mathfrak{M}_1(D) = \omega_1) &= p \\ P(\mathfrak{M}_1(D) = \omega_2) &= 1-p \\ P(\mathfrak{M}_1(D') = \omega_2) &= 1, \forall D' \in \mathbb{I}, D' \neq D \\ P(\mathfrak{M}_2(D') = \omega_2) &= 1, \forall D' \in \mathbb{I} \end{aligned}$$

Let $\mathcal{A}$ be an algorithm that always outputs ω_2. Then $\mathfrak{M}_2 = \mathcal{A} \circ \mathfrak{M}_1$. Using the assumption of $Q(D,\omega_1) < Q(D,\omega_2)$:

$$\begin{aligned} & {\mu_{\mathbb{I}}}^{\mathrm{SE}}(\mathfrak{M}_1) - {\mu_{\mathbb{I}}}^{\mathrm{SE}}(\mathfrak{M}_2) \\ = \;& P(D)pQ(D,\omega_1) + P(D)(1-p)Q(D,\omega_2) \\ & -P(D)Q(D,\omega_2) \\ = \;& P(D)p\Big(Q(D,\omega_1) - Q(D,\omega_2)\Big) < 0 \end{aligned}$$

since $P(D) > 0$ and $p > 0$. This contradicts the axiom of sufficiency. □

5.6.2 Proof of Theorem 5.2.4

Theorem 5.2.4. *Let $\mathbb{I}$ be finite. A utility ${\mu_{\mathbb{I}}}^{SM}$ or loss ${\mathcal{L}_{\mathbb{I}}}^{SM}$ measure (as described in Definition 5.2.2) satisfies Axiom 3.2.1 (sufficiency) only if it is a constant function.*

Proof. Suppose ${\mu_{\mathbb{I}}}^{\mathrm{SM}}$ satisfies Axiom 3.2.1. Choose an ordering $\omega_1, \omega_2, \ldots$ for the elements of $\mathbb{O}^*$.

Part 1: Showing $\min_{D \in \mathbb{I}} Q(D,\omega)$ is a constant γ independent of ω. For each ω, define the

data sanitizer $\mathfrak{M}_\omega$ that always outputs ω. Clearly, for any data sanitizer $\mathfrak{M}$, there exists a $\mathcal{A}$ such that $\mathfrak{M}_\omega = \mathcal{A} \circ \mathfrak{M}$ and so by Axiom 3.2.1 of sufficiency, ${\mu_{\mathbb{I}}}^{\mathrm{SM}}(\mathfrak{M}_\omega)$ is the same for all ω and corresponds to the minimum value ${\mu_{\mathbb{I}}}^{\mathrm{SM}}$ can take. Call this value γ.

By definition, ${\mu_{\mathbb{I}}}^{\mathrm{SM}}(\mathfrak{M}_\omega) = \gamma = \min_{D \in \mathbb{I}} Q(D, \omega)$ no matter which ω we choose.

Part 2: Showing that for every k, there exists a $D \in \mathbb{I}$ such that $Q(D, \omega_1) = Q(D, \omega_2) = \cdots = Q(D, \omega_k) = \gamma$

For each $k = 1, 2, \dots$ define the data sanitizer $\mathfrak{M}_k$ as follows:

$$P(\mathfrak{M}_k(D) = \omega_i) \quad = \quad \begin{cases} 1/k & \text{if } i \leq k \\ 0 & \text{otherwise} \end{cases}$$

Thus $\mathfrak{M}_k$ ignores its input and returns one of $\omega_1, \dots, \omega_k$ uniformly at random. Since the $\mathfrak{M}_k$ $(k = 1, 2, \dots)$ ignore their inputs then clearly for any data sanitizer $\mathfrak{M}$, there exists an algorithm $\mathcal{A}$ with $\mathfrak{M}_k = \mathcal{A} \circ \mathfrak{M}$ and so, if the axiom of sufficiency is to hold, ${\mu_{\mathbb{I}}}^{\mathrm{SM}}(\mathfrak{M}_k) = \gamma$ (the minimum value of ${\mu_{\mathbb{I}}}^{\mathrm{SM}}$) for all k. Since γ is the minimum value $Q(D, \omega)$ can take (for any D and ω), then by definition of ${\mu_{\mathbb{I}}}^{\mathrm{SM}}$, the dataset D that achieves the minimum in Equation 5.1 (in Definition 5.2.2) must have $Q(D, \omega_1) = Q(D, \omega_2) = \cdots = Q(D, \omega_k) = \gamma$

Part 3: Showing that there exists a $D^* \in \mathbb{I}$ such that $Q(D^*, \omega) = \gamma$ for all ω. If $\{\omega_1, \omega_2, \dots\}$ is finite, this follows from Part 2. If it is countably infinite, then suppose on the contrary that for every D there was an ω_{i_D} such that $Q(D, \omega_{i_D}) > \gamma$. Then since $\mathbb{I}$ is finite, we have a contradiction with Part 2 by setting $k = 1 + \max\{i_D \ : \ D \in \mathbb{I}\}$.

This D^* is the minimizer of Equation 5.1 in Definition 5.2.2. Hence ${\mu_{\mathbb{I}}}^{\mathrm{SM}}(\mathfrak{M}) = \gamma$ for all $\mathfrak{M}$. □

5.6.3 Proof of Theorem 5.2.5

Theorem 5.2.5. *The expected discrepancy utility ${\mu_{\mathbb{I}}}^{SE}$ and loss ${\mathcal{L}_{\mathbb{I}}}^{SE}$ measures (as Definition 5.2.1) always satisfy Axioms 3.3.2 (quasi-convexity) and 3.3.3 (quasi-concavity).*

Proof. Let $\mathfrak{M}_1$ and $\mathfrak{M}_2$ be two algorithms. We will show that

$${\mu_{\mathbb{I}}}^{\mathrm{SE}}(\mathfrak{M}_1 \oplus_p \mathfrak{M}_2) = p\, {\mu_{\mathbb{I}}}^{\mathrm{SE}}(\mathfrak{M}_1) + (1-p)\, {\mu_{\mathbb{I}}}^{\mathrm{SE}}(\mathfrak{M}_2)$$

from which quasi-convexity and quasi-concavity obviously follow.

Let $\mathbb{O}_1$ be the range of $\mathfrak{M}_1$ and $\mathbb{O}_2$ be the range of $\mathfrak{M}_2$ (note if there is an $\omega \in \mathbb{O}_1$ and $\omega \notin \mathbb{O}_2$, then by definition $P(\mathfrak{M}_2(D) = \omega) = 0$). Let $\mathfrak{M} = \mathfrak{M}_1 \oplus_p \mathfrak{M}_2$. Then

$$\begin{aligned} {\mu_{\mathbb{I}}}^{\mathrm{SE}}(\mathfrak{M}_1 \oplus_p \mathfrak{M}_2) &= \sum_{D \in \mathbb{I}} \sum_{\omega \in \mathbb{O}_1 \cup \mathbb{O}_2} Q(D, \omega)\ P(D)\ P(\mathfrak{M}(D) = \omega) \\ &= \sum_{D \in \mathbb{I}} \sum_{\omega \in \mathbb{O}_1 \cup \mathbb{O}_2} Q(D, \omega)\ P(D) \Big(pP(\mathfrak{M}_1(D) = \omega) + (1-p)P(\mathfrak{M}_2(D) = \omega) \Big) \\ &= p\, {\mu_{\mathbb{I}}}^{\mathrm{SE}}(\mathfrak{M}_1) + (1-p)\, {\mu_{\mathbb{I}}}^{\mathrm{SE}}(\mathfrak{M}_2) \end{aligned}$$

□

5.6.4 Proof of Theorem 5.2.6

Theorem 5.2.6. *Let $\mathbb{I}$ be finite. The worst-case discrepancy utility ${\mu_{\mathbb{I}}}^{SM}$ and loss ${\mathcal{L}_{\mathbb{I}}}^{SM}$ measures (Definition 5.2.2) always satisfy Axiom 3.3.3 (quasi-concavity).*

Proof. Let $\mathfrak{M}_1$ and $\mathfrak{M}_2$ be two algorithms. In the following, we show

$$\mu_{\mathbb{I}}{}^{\mathrm{SM}}(\mathfrak{M}_1 \oplus_p \mathfrak{M}_2) \geq p\,\mu_{\mathbb{I}}{}^{\mathrm{SM}}(\mathfrak{M}_1) + (1-p)\,\mu_{\mathbb{I}}{}^{\mathrm{SM}}(\mathfrak{M}_2)$$

from which quasi-concavity easily follows. Let $\mathbb{O}_1$ be the range of $\mathfrak{M}_1$ and $\mathbb{O}_2$ be the range of $\mathfrak{M}_2$ (note if there is an $\omega \in \mathbb{O}_1$ and $\omega \notin \mathbb{O}_2$, then by definition $P(\mathfrak{M}_2(D) = \omega) = 0$). Let $\mathfrak{M} = \mathfrak{M}_1 \oplus_p \mathfrak{M}_2$. Let

$$D^* = \arg\min_{D \in \mathbb{I}} \sum_{\omega \in \mathbb{O}_1 \cup \mathbb{O}_2} Q(D, \omega)\ P(\mathfrak{M}(D) = \omega)$$

so that

$$\mu_{\mathbb{I}}{}^{\mathrm{SM}}(\mathfrak{M}) \quad = \quad \sum_{\omega \in \mathbb{O}_1 \cup \mathbb{O}_2} Q(D^*, \omega)\ P(\mathfrak{M}(D^*) = \omega) = \sum_{\omega \in \mathbb{O}_1 \cup \mathbb{O}_2} Q(D^*, \omega)\ \Big(pP(\mathfrak{M}_1(D^*) = \omega) + (1-p)P(\mathfrak{M}_2(D^*) =$$

By definition of worst-case discrepancy, it follows that

$$\begin{aligned}
\mu_{\mathbb{I}}{}^{\mathrm{SM}}(\mathfrak{M}_1) &\leq \sum_{\omega \in \mathbb{O}_1 \cup \mathbb{O}_2} Q(D^*, \omega)\ P(\mathfrak{M}_1(D^*) = \omega) \text{ and} \\
\mu_{\mathbb{I}}{}^{\mathrm{SM}}(\mathfrak{M}_2) &\leq \sum_{\omega \in \mathbb{O}_1 \cup \mathbb{O}_2} Q(D^*, \omega)\ P(\mathfrak{M}_2(D^*) = \omega).
\end{aligned}$$

Thus, we get

$$\mu_{\mathbb{I}}{}^{\mathrm{SM}}(\mathfrak{M}_1 \oplus_p \mathfrak{M}_2) \geq p\,\mu_{\mathbb{I}}{}^{\mathrm{SM}}(\mathfrak{M}_1) + (1-p)\,\mu_{\mathbb{I}}{}^{\mathrm{SM}}(\mathfrak{M}_2)$$

□

5.6.5 Proof of Theorem 5.2.7

Theorem 5.2.7. *Let $\mathbb{I}$ be finite. A worst-case discrepancy utility ${\mu_{\mathbb{I}}}^{SM}$ or loss ${\mathcal{L}_{\mathbb{I}}}^{SM}$ measure satisfies Axiom 3.3.2 (quasi-convexity) if and only if there exists a D^* such that ${\mu_{\mathbb{I}}}^{SM}(\mathfrak{M})$ only depends on the distribution $P(\mathfrak{M}(D^*))$ for all $\mathfrak{M}$ (and similarly, ${\mathcal{L}_{\mathbb{I}}}^{SM}$ only depends on $P(\mathfrak{M}(D^*))$). That is, it only depends on the behavior of $\mathfrak{M}$ on a fixed dataset D^*.*

Proof. We first consider the "if" direction. For a given mechanism $\mathfrak{M}$, if D^* is not a minimizer in Equation 5.1 (in Definition 5.2.2), then $\mu_{\mathbb{I}}{}^{\mathrm{SM}}$ must be some fixed constant c (since it cannot

depend on the behavior of $\mathfrak{M}$ on datasets $D \neq D^*$). This means that for some $D' \neq D^*$, $Q(D', \omega) = c$ for all ω. For all other datasets D (i.e. other than D^* and D'), $Q(D, \omega) \geq c$ for all ω.

Thus, if a worst-case discrepancy measure $\mu_{\mathbb{I}}^{\text{SM}}$ only depends on the distribution of $\mathfrak{M}(D^*)$, then for some fixed c, $\mu_{\mathbb{I}}^{\text{SM}}(\mathfrak{M}) \equiv \min\{c, \sum_{\omega \in \mathbb{O}^*} Q(D^*, \omega) P(\mathfrak{M}(D^*) = \omega)\}$. Clearly this function satisfies quasi-convexity.

We now prove the "only if" direction here. Fix a dataset D^*. Consider the following two conditions on Q and D^*. If either one of them is satisfied, then clearly $\mu_{\mathbb{I}}^{\text{SM}}$ only depends on the distribution of $\mathfrak{M}(D^*)$.

1. $\forall D \neq D^*$, $\sup_{\omega \in \mathbb{O}^*} Q(D^*, \omega) \leq \inf_{\omega \in \mathbb{O}^*} Q(D, \omega)$
2. $\exists D' \in \mathbb{I}$ and a constant c such that $Q(D', \omega) = c$ for all $\omega \in \mathbb{O}^*$ and $\forall D \notin \{D^*, D'\}$, we have $Q(D, \omega) \geq c$ for all $\omega \in \mathbb{O}^*$.

We now choose $D^* = \arg\min_{D \in \mathbb{I}} \inf_\omega Q(D, \omega)$ and will prove that if both conditions are violated for this D^*, then $\mu_{\mathbb{I}}^{\text{SM}}$ cannot satisfy quasi-convexity.

Now choose $D^\dagger = \arg\min_{D \neq D^*} \inf_\omega Q(D, \omega)$ (i.e., the next "minimal" dataset after D^*). Now, for D^* there exist $\omega^*_{\text{hi}}, \omega^*_{\text{lo}} \in \mathbb{O}^*$, for $D^\dagger$ there exist $\omega^\dagger_{\text{lo}}, \omega^\dagger_{\text{hi}} \in \mathbb{O}^*$, and for all other $D \in \mathbb{I}$ there exist corresponding $\omega_D \in \mathbb{O}^*$ with the following properties:

- $Q(D^*, \omega^*_{\text{lo}}) \leq Q(D^\dagger, \omega^\dagger_{\text{lo}}) < Q(D^*, \omega^*_{\text{hi}})$ (this is due to the failure of condition (1))
- $Q(D^\dagger, \omega^\dagger_{\text{lo}}) < Q(D^\dagger, \omega^\dagger_{\text{hi}})$ (this is due to the definition of $D^\dagger$ and the failure of condition (2) which serve to prevent $Q(D^\dagger, \cdot)$ from being constant in the second argument).
- $Q(D^\dagger, \omega^\dagger_{\text{lo}}) < Q(D, \omega_D)$ for all $D \notin \{D^*, D^\dagger\}$ (this is due to the definition of $D^\dagger$ and the failure of condition (2) which prevents the following statement from being true: $\forall \omega'$: $Q(D, \omega') = \inf_\omega Q(D^\dagger, \omega)$).

We now construct two mechanisms $\mathfrak{M}_1$ and $\mathfrak{M}_2$ such that $\mu_{\mathbb{I}}^{\text{SM}}(\mathfrak{M}_1) = \mu_{\mathbb{I}}^{\text{SM}}(\mathfrak{M}_2)$ and $\mu_{\mathbb{I}}^{\text{SM}}(\mathfrak{M}_1 \oplus_{0.5} \mathfrak{M}_2) > \max\{\mu_{\mathbb{I}}^{\text{SM}}(\mathfrak{M}_1), \mu_{\mathbb{I}}^{\text{SM}}(\mathfrak{M}_2)\}$.

$$\begin{aligned} \mathfrak{M}_1(D^*) &= \omega^*_{\text{hi}} \\ \mathfrak{M}_1(D^\dagger) &= \omega^\dagger_{\text{lo}} \\ \mathfrak{M}_1(D) &= \omega_D \text{ for all other } D \in \mathbb{I} \end{aligned}$$

By construction, it is clear that $\mu_{\mathbb{I}}^{\text{SM}}(\mathfrak{M}_1) = Q(D^\dagger, \omega^\dagger)$ because $D^\dagger$ is the worst-case dataset for $\mathfrak{M}_1$ (i.e., it minimizes Equation 5.1 (Definition 5.2.2). We then define

$$\mathfrak{M}_2(D^*) = \omega^*_{\text{hi}} \text{ with probability } \frac{Q(D^\dagger, \omega^\dagger_{\text{lo}}) - Q(D^*, \omega^*_{\text{lo}})}{Q(D^*, \omega^*_{\text{hi}}) - Q(D^*, \omega^*_{\text{lo}})}$$

$$\begin{aligned}\mathfrak{M}_2(D^*) &= \omega^*_{\text{lo}} \text{ with probability } \frac{Q(D^*,\omega^*_{\text{hi}}) - Q(D^\dagger,\omega^\dagger_{\text{lo}})}{Q(D^*,\omega^*_{\text{hi}}) - Q(D^*,\omega^*_{\text{lo}})}\\ \mathfrak{M}_2(D^\dagger) &= \omega^\dagger_{\text{hi}}\\ \mathfrak{M}_2(D) &= \omega_D \text{ for all other } D \in \mathbb{I}\end{aligned}$$

By construction, the worst-case dataset for $\mathfrak{M}_2$ is D^* and a simple calculation shows that ${\mu_{\mathbb{I}}}^{\text{SM}}(\mathfrak{M}_1) = Q(D^\dagger, \omega^\dagger)$.

Now consider $\mathfrak{M} \stackrel{\text{def}}{=} \mathfrak{M}_1 \oplus_{0.5} \mathfrak{M}_2$. If the worst-case dataset for it is D^* (i.e., in Equation 5.1 in Definition 5.2.2), then clearly the corresponding utility is strictly greater than that of $\mathfrak{M}_2$ (for which D^* was the worst-case dataset) and hence the utility is also strictly greater than that of $\mathfrak{M}_1$ (since they both have the same utility). If the worst-case dataset was $D^\dagger$, then clearly the corresponding utility is strictly greater than that of $\mathfrak{M}_1$ (for which $D^\dagger$ was the worst-case dataset) and so it is also strictly greater than the utility for $\mathfrak{M}_2$. If the worst-case dataset is some other $D \in \mathbb{I}$, then clearly its utility is strictly greater that of both $\mathfrak{M}_1$ and $\mathfrak{M}_2$ since such a D is never the worst-case dataset for either of those two mechanisms.

□

5.6.6 Proof of Theorem 5.2.9

To prove theorem 5.2.9, we need the following technical result from [18]:

Lemma 5.6.1 ([18])**.** *A utility measure $\mu_{\mathbb{I}}$ having the form:*

$$\mu_{\mathbb{I}}(\mathfrak{M}) = \sum_{\omega \in \text{range}(\mathfrak{M})} F(P(\mathfrak{M}(D_1) = \omega), \ldots, P(\mathfrak{M}(D_n) = \omega))$$

for some continuous function F satisfies Axiom 3.2.1 of sufficiency if F is convex over vectors with no negative components and $F(c\vec{x}) = cF(\vec{x})$ for all nonnegative c.

Theorem 5.2.9. *The loss measure ${\mathcal{L}_{\mathbb{I}}}^{MAP}(\mathfrak{M})$ described in Definition 5.2.8 satisfies Axiom 3.2.1 if and only if the corresponding error function E has the form*

$$E(D, \widehat{D}) = \alpha H(D, \widehat{D}) + f(D)$$

where α is a nonnegative constant, f is an arbitrary function, and $H(D, \widehat{D}) = 0$ when $D = \widehat{D}$ and equals 1 otherwise. The condition for the utility measure ${\mu_{\mathbb{I}}}^{MAP}$ is that the corresponding quality function Q has the form

$$Q(D, \widehat{D}) = \alpha(1 - H(D, \widehat{D})) + f(D)$$

Proof. We prove the utility version of this theorem. To simplify the notation, $1 - H(D, \widehat{D})$ is replaced with $G(D, \widehat{D})$ and hence $G(D, \widehat{D}) = 1$ when $D = \widehat{D}$ and is 0 otherwise.

Part 1: Showing if direction It is clear from the definition of MAP guessing that removing

the "$+f(D_i)$" term only changes the utility measure by a constant amount, so without loss of generality, we may assume that $Q = \alpha G$(i.e., $Q(D, \widehat{D}) = \alpha$ when $D = \widehat{D}$ and 0 otherwise).

Define $\widehat{D}_{(\mathfrak{M},\omega)}$ to be $\arg\max_{D\in\mathbb{I}} P(D)P(\mathfrak{M}(D) = \omega)$ (which is a maximizer of the posterior probability).

Then by Lemma 5.6.1, we need to show that the function $F(\vec{P}[\mathfrak{M}(\cdot) = \omega]) \stackrel{\text{def}}{=} \sum_{D\in\mathbb{I}} P[D]\ P[\mathfrak{M}(D) = \omega]\ \alpha H(D, \widehat{D}_{(\mathfrak{M},\omega)})$ is convex (over vectors with no negative components) and $F(c\vec{x}) = cF(\vec{x})$. The latter is obvious, so we just focus on convexity. Now rewriting F,

$$\begin{aligned} F(\vec{P}[\mathfrak{M}(\cdot) = \omega]) &= \sum_{D\in\mathbb{I}} P[D]\ P[\mathfrak{M}(D) = \omega]\ \alpha H(D, \widehat{D}_{(\mathfrak{M},\omega)}) \\ &= P[\widehat{D}_{(\mathfrak{M},\omega)}]\ P[\mathfrak{M}(\widehat{D}_{(\mathfrak{M},\omega)}) = \omega]\ \alpha \\ &= \alpha \max_{D\in\mathbb{I}} P(D)P(\mathfrak{M}(D) = \omega) \end{aligned}$$

which is clearly a convex function of the vector $\vec{P}[\mathfrak{M}(\cdot) = \omega]$.

Part 2: Showing only if direction Now suppose $\mu_{\mathbb{I}}{}^{\text{MAP}}$, with associated quality function Q, satisfies Axiom 3.2.1 of sufficiency. Pick any two $D_a, D_b \in \mathbb{I}$ and some ω and consider the class of all data sanitizers $\mathfrak{M}$ such that $P(D_a)P(\mathfrak{M}(D_a) = \omega) = P(D_b)(\mathfrak{M}(D_b) = \omega) > P(D^*)P(\mathfrak{M}(D^*) = \omega)$ for all $D^* \notin \{D_a, D_b\}$. Thus, when the output is ω, both D_a and D_b achieve the maximum of the posterior probability. Therefore in order for $\mu_{\mathbb{I}}{}^{\text{MAP}}$ to be well-defined, we need

$$\begin{aligned} &\sum_{D\in\mathbb{I}} P[D]\ P[\mathfrak{M}(D) = \omega]\ \alpha Q(D, D_a) \\ &= \sum_{D\in\mathbb{I}} P[D]\ P[\mathfrak{M}(D) = \omega]\ \alpha Q(D, D_b) \\ &\Rightarrow \sum_{D\in\mathbb{I}} P[D]\ P[\mathfrak{M}(D) = \omega]\ \alpha\Big(Q(D, D_a) - Q(D, D_b)\Big) = 0 \end{aligned}$$

Writing $y_D = P[D]P[\mathfrak{M}(D) = \omega]$ and $q_D = Q(D, D_a) - Q(D, D_b)$ this becomes the linear equation

$$\sum_{D\in\mathbb{I}} y_D q_D = 0 \tag{5.5}$$

where the y_D are free variables that satisfy the constraints $y_{D_a} = y_{D_b}$ and $0 \le y_D < y_{D_a}$ for $D \notin \{D_a, D_b\}$ (by construction).

Part 2a: showing that $\exists\gamma$, s.t. $Q(D, D) - Q(D, D^*) = \gamma$ for all D, D^*. Setting $y_D = 0$ for $D \notin \{D_a, D_b\}$ shows that $q_{D_a} = -q_{D_b}$ (since $y_{D_a} = y_{D_b}$). This implies $Q(D_a, D_a) - Q(D_a, D_b) = -Q(D_b, D_a) + Q(D_b, D_b)$ since D_a and D_b were arbitrarily picked at the beginning of Part 2, this shows that there is a constant γ such that $Q(D, D) - Q(D, D^*) = \gamma$ for all D, D^*.

Part 2b: showing $Q(D, D^*) = \gamma H(D, D^*) + f(D)$. Note that

$$Q(D, D) - Q(D, D^*) = \gamma$$

implies that

$$Q(D, D^*) = Q(D, D')$$

whenever $D^* \neq D'$. Therefore we can safely define the function $f(D) \stackrel{\text{def}}{=} Q(D, D^*)$ for any choice of $D^* \neq D$ (f is well-defined since any such D^* gives the same result). Define $H(D, D^*) = 1$ if D and D^* are the same and 0 otherwise. Then $Q(D, D^*) = \gamma H(D, D^*) + f(D)$ (since $Q(D, D) = \gamma + f(D) = \gamma + Q(D, D^*)$ which is true by Part 2a and the well-definedness of f, and clearly $Q(D, D^*) = f(D)$).

Part 2c: showing $\gamma \geq 0$. We know that $Q(D, D^*) = \gamma H(D, D^*) + f(D)$ where $H(D, D^*) = 1$ if $D = D^*$ and 0 otherwise. Removing the "$+f(D)$" term only changes ${\mu_{\mathbb{I}}}^{\text{MAP}}$ by a constant and therefore removing it has no effect on whether or not Axiom 3.2.1 is satisfied. Once the $f(D)$ terms has been removed, multiplying γ by a positive constant only has the effect of multiplying the utility measure by that constant and therefore also has no effect on whether or not Axiom 3.2.1 is satisfied.

If $\gamma = 0$ then we are done. Thus without loss of generality (based on the discussion of the preceding paragraph), we can assume $f(D) \equiv 0$ and $|\gamma| = 1$. Our goal is then to prove that $\gamma \neq -1$.

By way of contradiction, assume $\gamma = -1$ so that $Q(D, D^*) = -1$ if $D = D^*$ and 0 otherwise. Then consider the data sanitizer $\mathfrak{M}_{\text{id}}$ that always outputs its input. Then $\text{range}(\mathfrak{M}_{\text{id}}) = \mathbb{I}$ and upon observing the output D_i, the posterior distribution is maximized at D_i as well (so $\widehat{D}_{(\mathfrak{M}_{\text{id}}, D_i)} = D_i$)

$$\begin{aligned}
&{\mu_{\mathbb{I}}}^{\text{MAP}}(\mathfrak{M}_{\text{id}}) \\
&= \sum_{\omega \in \text{range}(\mathfrak{M}_{\text{id}})} \sum_{D \in \mathbb{I}} P[D]\ P[\mathfrak{M}_{\text{id}}(D) = \omega]\ Q(D, \widehat{D}_{(\mathfrak{M}_{\text{id}}, \omega)}) \\
&= \sum_{D^* \in \mathbb{I}} \sum_{D \in \mathbb{I}} P[D]\ P[\mathfrak{M}_{\text{id}}(D) = D^*]\ Q(D, D^*) \\
&= \sum_{D^* \in \mathbb{I}} P[D^*]\ Q(D^*, D^*) \\
&= \sum_{D^* \in \mathbb{I}} -P(D^*) \\
&= -1
\end{aligned}$$

On the other hand, consider the data sanitizer $\mathfrak{M}_0$ that always outputs 0.

$$\begin{aligned}
&{\mu_{\mathbb{I}}}^{\text{MAP}}(\mathfrak{M}_0) \\
&= \sum_{\omega \in \text{range}(\mathfrak{M}_0)} \sum_{D \in \mathbb{I}} P[D]\ P[\mathfrak{M}_0(D) = \omega]\ Q(D, \widehat{D}_{(\mathfrak{M}_0, \omega)})
\end{aligned}$$

$$= \sum_{D \in \mathbb{I}} P[D]\ Q(D, \widehat{D}_{(\mathfrak{M}_0,0)}) \tag{5.6}$$

Then after seeing the output 0, the prior and posterior distributions are still the same and $\widehat{D}_{(\mathfrak{M}_0,0)}$, a dataset with largest prior/posterior probability may not be unique, but any choice (randomized or deterministic) of a dataset that maximizes $P(D)$ will give the same utility score in Equation 5.6, and so ${\mu_{\mathbb{I}}}^{\text{MAP}}(\mathfrak{M}_0) = -\max_{D \in \mathbb{I}} P(D)$. Since our proof statement had the condition[7] that $P(D) > 0$ for all D, this implies that ${\mu_{\mathbb{I}}}^{\text{MAP}}(\mathfrak{M}_0) > -1 = {\mu_{\mathbb{I}}}^{\text{MAP}}(\mathfrak{M}_{\text{id}})$. However, $\mathfrak{M}_0 = \mathfrak{M}_0 \circ \mathfrak{M}_{\text{id}}$ and so according to Axiom 3.2.1 we should have ${\mu_{\mathbb{I}}}^{\text{MAP}}(\mathfrak{M}_{\text{id}}) \geq {\mu_{\mathbb{I}}}^{\text{MAP}}(\mathfrak{M}_0)$. This is a contradiction that arose from the assumption that $\gamma = -1$ (which we showed, without loss of generality, was equal to the case of $\gamma < 1$). Thus we must have $\gamma \geq 0$ and the proof is done.

□

5.6.7 Proof of Theorem 5.2.10

Theorem 5.2.10. *A utility ${\mu_{\mathbb{I}}}^{MAP}$ or loss measure ${\mathcal{L}_{\mathbb{I}}}^{MAP}$ (as described in Definition 5.2.8) always satisfies Axioms 3.3.2 (quasi-convexity) and 3.3.3 (quasi-concavity).*

Proof. Let $\mathfrak{M}_1$ be an algorithm with range $\mathbb{O}_1$ and $\mathfrak{M}_2$ be an algorithm with range $\mathbb{O}_2$. Let $\mathfrak{M} = \mathfrak{M}_1 \oplus_p \mathfrak{M}_2$. Note, if there is any $\omega \in \mathbb{O}_1 \cap \mathbb{O}_2$ then $\mathfrak{M}$ provides disambiguation because it reveals which algorithm ($\mathfrak{M}_1$ or $\mathfrak{M}_2$) was actually used to produce that ω. Hence

$$\begin{aligned}
{\mu_{\mathbb{I}}}^{\text{MAP}}(\mathfrak{M}) &= \sum_{\omega \in \text{range}(\mathfrak{M})} \sum_{D \in \mathbb{I}} P(D)\ P(\mathfrak{M}(D) = \omega)\ Q(D, \widehat{D}_{(\mathfrak{M},\omega)}) \\
&= \sum_{\omega \in \mathbb{O}_1} \sum_{D \in \mathbb{I}} P(D)\ p\ P(\mathfrak{M}_1(D) = \omega)\ Q(D, \widehat{D}_{(\mathfrak{M}_1,\omega)}) \\
&\quad + \sum_{\omega \in \mathbb{O}_1} \sum_{D \in \mathbb{I}} P(D)\ (1-p)\ P(\mathfrak{M}_2(D) = \omega)\ Q(D, \widehat{D}_{(\mathfrak{M}_2,\omega)}) \\
&= p\, {\mu_{\mathbb{I}}}^{\text{MAP}}(\mathfrak{M}_1) + (1-p)\, {\mu_{\mathbb{I}}}^{\text{MAP}}(\mathfrak{M}_2).
\end{aligned}$$

and both quasi-convexity and quasi-concavity follow. □

5.6.8 Proof of Theorem 5.2.12

Theorem 5.2.12. *Given a finite input domain $\mathbb{I}$, a prior with $P(D_i) > 0$ for all $D_i \in \mathbb{I}$ and an error function $E : \mathbb{I} \times \mathbb{I} \to \mathbb{R}$ the loss measure ${\mathcal{L}_{\mathbb{I}}}^{BG}$ (as described in Definition 5.2.11) satisfies Axiom 3.2.1 (sufficiency) if and only if*

$$E(D, \widehat{D}) = S(D, \widehat{D}) + f(D) + g(\widehat{D})$$

[7] This assumption was to avoid unnecessary technicalities. Without it, the theorem statement needs to only state properties of $Q(D_i, D_)$ for datasets that do not have 0 prior probability.

for some negative semidefinite[8] *function S and arbitrary functions g and f.*

Proof. We will prove the utility version of this theorem and results for loss measures follow by setting $E \stackrel{\text{def}}{=} -Q$. Therefore, for the utility version, we must show that $Q(D, \widehat{D}) = S(D, \widehat{D}) + f(D) + g(\widehat{D})$ where S is positive semidefinite.

Let $\mu_{\mathbb{I}}{}^{\text{BG}}$ be the utility measure with quality function Q. Then

$$\mu_{\mathbb{I}}{}^{\text{BG}}(\mathfrak{M}) \quad = \quad \sum_{\omega \in \mathfrak{M}} F(\vec{P}[\mathfrak{M}(\cdot) = \omega])$$

where

$$F(\vec{P}[\mathfrak{M}(\cdot) = \omega]) \quad \stackrel{\text{def}}{=} \quad \frac{\sum_{D \in \mathbb{I}} \sum_{\widehat{D} \in \mathbb{I}} \; Q(D, \widehat{D}) \;\; P[D] \; P[\mathfrak{M}(D) = \omega] \;\; P[\widehat{D}] \; P[\mathfrak{M}(\widehat{D}) = \omega]}{\sum_{D^\star \in \mathbb{I}} \; P[D^\star] \; P[\mathfrak{M}(D^\star) = \omega]}$$

Note that F is continuous (and note $F(\vec{0}) = 0$ by continuity) and therefore by Lemma 5.6.2, $\mu_{\mathbb{I}}{}^{\text{BG}}$ satisfies Axiom 3.2.1 of sufficiency if and only if F is convex over vectors with no negative components and for all such vectors $F(c\vec{x}) = cF(\vec{x})$ whenever $c > 0$. Clearly this latter property is true of F so we just need to prove convexity of F.

We will show that F is convex if and only if $Q(D_i, D_j) = S^*(D_i, D_j) + f^*(D_i) + g^*(D_j)$ where S^* is positive semidefinite (since we are proving the utility version) but not necessarily symmetric. **To simplify notation, define M to be the $|\mathbb{I}| \times |\mathbb{I}|$ matrix whose rows and columns are indexed by $D \in \mathbb{I}$ such that the (D_i, D_j) entry of M is $Q(D_i, D_j)$.** In matrix form, this condition is $M = S + \vec{1}\vec{u}^T + \vec{v}\vec{1}^T$ (where S is the matrix representation of S^*, $\vec{u}$ is a vector whose i^{th} component is $g(D_i)$ and $\vec{v}$ is a vector whose i^{th} component is $f(D_i)$).

Part 1: Showing F is convex if $M = S + \vec{1}\vec{u}^T + \vec{v}\vec{1}^T$ and S is positive semidefinite. Note that F is representable as

$$F(\vec{x}) \quad = \quad \frac{\vec{x}^T M \vec{x}}{\vec{x}^T \vec{1}} = \frac{\vec{x}^T S \vec{x}}{\vec{x}^T \vec{1}} + \vec{u}^T \vec{x} + \vec{x}^T \vec{v}$$

for vectors $\vec{x} \neq \vec{0}$ that have no negative components (once we show convexity of F on this set, convexity at 0 follows by continuity). Now, the linear terms have no effect on convexity, so we just need to show convexity for the function:

$$F^*(\vec{x}) \quad = \quad \frac{\vec{x}^T S \vec{x}}{\vec{x}^T \vec{1}}$$

Define the functions $\phi_0(\vec{x}) = \vec{x}^T S \vec{x}$, $\phi_1(\vec{x}, t) = t\phi_0(\vec{x}/t)$, and $\psi(\vec{x}) = (\vec{x}, \vec{x}^T \vec{1})$. Then

- ϕ_0 is convex because S is positive semidefinite

[8] A function S is negative semidefinite if for all functions h, we have $\sum_{D \in \mathbb{I}} \sum_{\widehat{D} \in \mathbb{I}} h(D) h(\widehat{D}) S(D, \widehat{D}) \leq 0$; we do not require S to be symmetric.

- $\phi_1(\vec{x}, t)$ is convex when $t > 0$ because it is the *perspective* of a convex function.[9]
- $\phi_1 \circ \psi = F^*$ (over vectors with no negative components)
- F^* is convex (over vectors with no negative components) because because ψ is linear and ϕ_1 is convex.
- F is convex (over vectors with no negative components) because it differs from F^* by a linear function.

Part 2: Showing F is convex implies $M = S + \vec{1}\vec{u}^T + \vec{v}\vec{1}^T$ and S is positive semidefinite.

Suppose F is convex over vectors with no negative components. Over such vectors, by definition of F, it follows that $cF(\vec{x}) = F(c\vec{x})$ for all $c > 0$ and in the presence of this condition, convexity is equivalent to $F(\vec{x} + \vec{y}) \leq F(\vec{x}) + F(\vec{y})$. Recalling that $F(\vec{x}) = \frac{\vec{x}^T M \vec{x}}{\vec{x}^T \vec{1}}$,

$$
\begin{aligned}
& F(\vec{x}) + F(\vec{y}) - F(\vec{x} + \vec{y}) \geq 0 \\
\Rightarrow \quad & \frac{\vec{x}^T M \vec{x}}{\vec{x}^T \vec{1}} + \frac{\vec{y}^T M \vec{y}}{\vec{y}^T \vec{1}} - \frac{(\vec{x} + \vec{y})^T M (\vec{x} + \vec{y})}{(\vec{x} + \vec{y})^T \vec{1}} \geq 0 \\
\Rightarrow \quad & \left[\vec{y}^T \vec{1}(\vec{x}^T \vec{1} + \vec{y}^T \vec{1})\right] \ \vec{x}^T M \vec{x} + \left[\vec{x}^T \vec{1}(\vec{x}^T \vec{1} + \vec{y}^T \vec{1})\right] \ \vec{y}^T M \vec{y} - \left[\vec{x}^T \vec{1} \vec{y}^T \vec{1}\right] \ (\vec{x} + \vec{y})^T M (\vec{x} + \vec{y}) \geq 0 \\
\Rightarrow \quad & \frac{\vec{x}^T}{\vec{x}^T \vec{1}} M \frac{\vec{x}}{\vec{x}^T \vec{1}} + \frac{\vec{y}^T}{\vec{y}^T \vec{1}} M \frac{\vec{y}}{\vec{y}^T \vec{1}} - 2 \frac{\vec{x}^T}{\vec{x}^T \vec{1}} M \frac{\vec{y}}{\vec{y}^T \vec{1}} \geq 0 \\
\Rightarrow \quad & (\frac{\vec{x}}{\vec{x}^T \vec{1}} - \frac{\vec{y}}{\vec{y}^T \vec{1}})^T M (\frac{\vec{x}}{\vec{x}^T \vec{1}} - \frac{\vec{y}}{\vec{y}^T \vec{1}}) \geq 0 \\
\Rightarrow \quad & \vec{z}^T M \vec{z} \geq 0 \text{ for all } \vec{z} \text{ where } \vec{z}^T \vec{1} = 0
\end{aligned}
$$

Now consider the matrix $V = I - \vec{1}\ \vec{1}^T / |\mathbb{I}|$ (where I is the $|\mathbb{I}| \times |\mathbb{I}|$ identity matrix). Then $V\vec{x}$ is a vector whose components sum up to 0 and $V\vec{z} = \vec{z}$ whenever $\vec{z}$ is a vector whose components add up to 0. Thus

$$
\begin{aligned}
& \vec{z}^T M \vec{z} \geq 0 \text{ for all } \vec{z} \text{ where } \vec{z}^T \vec{1} = 0 \\
\Leftrightarrow \quad & \vec{z}^T V^T M V \vec{z} \geq 0 \text{ for all } \vec{z} \text{ where } \vec{z}^T \vec{1} = 0 \\
\Leftrightarrow \quad & \vec{z}^T V^T M V \vec{z} \geq 0 \text{ for all } \vec{z} \in \mathbb{R}^{|\mathbb{I}|}
\end{aligned}
$$

and so F is convex implies $V^T M V$ is positive semidefinite.

Now consider

$$
M - V^T M V \quad = \quad M - (I - \frac{1}{|\mathbb{I}|} \vec{1}\vec{1}^T)^T M (I - \frac{1}{|\mathbb{I}|} \vec{1}\vec{1}^T)
$$

[9] Stephen Boyd and Lieven Vandenberghe, *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004.

$$
\begin{aligned}
&= \frac{1}{|\mathbb{I}|}(M\vec{1}\vec{1}^T + \vec{1}\vec{1}^T M) - \frac{1}{|\mathbb{I}|^2}\vec{1}\vec{1}^T M\vec{1}\vec{1}^T \\
&= \vec{1}(\frac{1}{|\mathbb{I}|}\vec{1}^T M) + (\frac{1}{|\mathbb{I}|}M\vec{1} - \frac{1}{|\mathbb{I}|^2}\vec{1}\vec{1}^T M\vec{1})\vec{1}^T
\end{aligned}
$$

Set $\vec{u} = (\frac{1}{|\mathbb{I}|}\vec{1}^T M)^T$, set $\vec{v} = (\frac{1}{|\mathbb{I}|}M\vec{1} + \frac{1}{|\mathbb{I}|^2}\vec{1}\vec{1}^T M\vec{1})$, and set $S = V^T MV$ (and note that we proved S is positive semidefinite). Thus

$$M = S + \vec{1}\vec{u}^T + \vec{v}\vec{1}^T$$

□

5.6.9 Proof of Lemma 5.2.13

Lemma 5.2.13. *Given a finite input domain $\mathbb{I}$ and a prior over $\mathbb{I}$, let ${\mathcal{L}_{\mathbb{I}}}^{(1)}$ be the expected error of the posterior distribution resulting from error function E and let ${\mathcal{L}_{\mathbb{I}}}^{(2)}$ be the corresponding loss measure resulting from error function $E(D, \widehat{D}) + f(D) + g(\widehat{D})$. Then there exists a constant c, depending only on f and g such that ${\mathcal{L}_{\mathbb{I}}}^{(2)}(\mathfrak{M}) = {\mathcal{L}_{\mathbb{I}}}^{(1)}(\mathfrak{M}) + c$ for all $\mathfrak{M}$.*

Proof. Note that according to the prior, the probability of seeing output ω is the result of marginalizing the joint distribution of ω and D as follows:

$$P[\omega] = \sum_{D\in\mathbb{I}} P(\omega, D) = \sum_{D\in\mathbb{I}} P[D]\ P[\mathfrak{M}(D) = \omega]$$

Also note that:

$$
\begin{aligned}
&\sum_{\omega\in\text{range}(\mathfrak{M})} \frac{\sum_{D\in\mathbb{I}}\sum_{\widehat{D}\in\mathbb{I}} f(D)\ \times\ P[D]\ P[\mathfrak{M}(D) = \omega]\ \times P[\widehat{D}]\ P[\mathfrak{M}(\widehat{D}) = \omega]}{\sum_{D^\star\in\mathbb{I}}\ P[D^\star]\ P[\mathfrak{M}(D^\star) = \omega]} \\
&= \sum_{\omega\in\text{range}(\mathfrak{M})} \frac{\sum_{D\in\mathbb{I}}\sum_{\widehat{D}\in\mathbb{I}} f(D)\ \times\ P[D]\ P[\mathfrak{M}(D) = \omega]\ \times P[\widehat{D}]\ P[\mathfrak{M}(\widehat{D}) = \omega]}{P(\omega)} \\
&\quad \text{(Marginalize out the } \widehat{D} \text{ to get a } P(\omega)) \\
&= \sum_{\omega\in\text{range}(\mathfrak{M})} \frac{\sum_{D\in\mathbb{I}} f(D)\ P[D]\ P[\mathfrak{M}(D) = \omega]\ P[\omega]}{P(\omega)} \\
&= \sum_{\omega\in\text{range}(\mathfrak{M})} \sum_{D\in\mathbb{I}} f(D)\ P[D]\ P[\mathfrak{M}(D) = \omega] \\
&= \sum_{D\in\mathbb{I}} f(D)\ P[D] \sum_{\omega\in\text{range}(\mathfrak{M})} P[\mathfrak{M}(D) = \omega] \\
&= \sum_{D\in\mathbb{I}} f(D)\ P[D] \\
&= \mathbb{E}[f]
\end{aligned}
$$

This calculation (used once more with $g(\widehat{D})$ replacing $f(D)$), and Definition 5.2.11 (expected

error of the posterior distribution) imply that:

$$\mathcal{L}_{\mathbb{I}}{}^{(2)} = \mathcal{L}_{\mathbb{I}}{}^{(1)} + \mathbb{E}[f] + \mathbb{E}[g]$$

and so the desired constant c is the expected value of f (using the given prior over $\mathbb{I}$) plus the expected value of g. □

5.6.10 Proof of Theorem 5.2.14

Theorem 5.2.14. *Choose a finite input domain $\mathbb{I}$ and prior such that $P(D_i) > 0$ for all $D_i \in \mathbb{I}$. If E is a error function that is also a Euclidean distance function then the corresponding loss measure $\mathcal{L}_{\mathbb{I}}{}^{BG}$ satisfies Axiom 3.2.1. Conversely, let $\mathcal{L}_{\mathbb{I}}{}^{(1)}$ be the expected error of the posterior distribution resulting from some error function $E^{(1)}$. If $\mathcal{L}_{\mathbb{I}}{}^{(1)}$ satisfies Axiom 3.2.1, then there exists a constant c and a Euclidean distance function E such that $\mathcal{L}_{\mathbb{I}}{}^{(2)}$, defined as the expected error of the posterior distribution with respect to E, satisfies $\mathcal{L}_{\mathbb{I}}{}^{(1)}(\mathfrak{M}) = \mathcal{L}_{\mathbb{I}}{}^{(2)}(\mathfrak{M}) + c$ (for all $\mathfrak{M}$).*

Proof.
Part 1: First, let E be a Euclidean distance function. Then there exists a space $\mathbb{R}^k$ and a mapping $\phi : \mathbb{I} \to \mathbb{R}^k$ such that $E(D_i, D_j) = ||\phi(D_i) - \phi(D_j)||_2^2$. So

$$\begin{aligned} E(D_i, D_j) &= ||\phi(D_i) - \phi(D_j)||_2^2 \\ &= -\phi(D_i) \cdot \phi(D_j) + ||\phi(D_i)||_2^2 + ||\phi(D_j)||_2^2 \\ &= S(D_i, D_j) + f(D_i) + g(D_j) \end{aligned}$$

where $S(D_i, D_j) \stackrel{\text{def}}{=} -\phi(D_i) \cdot \phi(D_j)$ is known to be negative semi-definite (since it is the negative of dot products) and $f(D) = g(D) = ||\phi(D)||_2^2$. Thus by Theorem 5.2.12, using E as the error function results in a loss measure that satisfies Axiom 3.2.1 of sufficiency.

Part 2: Conversely, let $\mathcal{L}_{\mathbb{I}}{}^{\text{BG}}$ be a loss measure (of expected posterior error) that satisfies Axiom 3.2.1 and uses an error function E. Then there exists a negative semidefinite function S and functions f and g such that $E(D_i, D_j) = S(D_i, D_j) + f(D_i) + g(D_j)$. Note that we can assume without loss of generality that S is symmetric (if it isn't, replacing S with $S'(D_i, D_j) = \frac{1}{2}[S(D_i, D_j) + S(D_j, D_i)]$ does not change the value of the loss measure). Now, using S as an error function in place of E (keeping the same prior) results in a loss measure $\mathcal{L}_{\mathbb{I}}{}^{(a)}$ that differs from $\mathcal{L}_{\mathbb{I}}{}^{\text{BG}}$ by a constant.

We can represent S as an $|\mathbb{I}| \times |\mathbb{I}|$ matrix M whose rows and columns are indexed by $D \in \mathbb{I}$ such that the (D_i, D_j) entry of M is $S(D_i, D_j)$. This matrix M is then symmetric negative semidefinite and therefore can be represented as $M = -X^T X$ where X is some $|\mathbb{I}| \times |\mathbb{I}|$ matrix. The (D_i, D_j) entry of M is then the dot product of the column of X indexed by D_i and the column indexed by D_j. Let $\phi(D)$ be the column of X indexed by D. Then $S(D_i, D_j) = -\phi(D_i) \cdot \phi(D_j)$.

Define $H(D_i, D_j) = ||\phi(D_i) - \phi(D_j)||_2^2 = S(D_i, D_j) + ||\phi(D_i)||_2^2 + ||\phi(D_j)||_2^2$. Then clearly H

is a Euclidean distance function. Recall that ${\mathcal{L}_{\mathbb{I}}}^{(a)}$ is a loss measure (of the expected posterior error) that uses S as its error function and differs from ${\mathcal{L}_{\mathbb{I}}}^{\mathrm{BG}}$ by a constant. Using H as an error function instead of S (keeping the same prior) results in a loss measure ${\mathcal{L}_{\mathbb{I}}}^{(b)}$ that differs from ${\mathcal{L}_{\mathbb{I}}}^{(a)}$ by a constant (by Lemma 5.2.13) and therefore ${\mathcal{L}_{\mathbb{I}}}^{(b)}$ differs from ${\mathcal{L}_{\mathbb{I}}}^{\mathrm{BG}}$ by a constant. □

5.6.11 Proof of Theorem 5.2.15

Theorem 5.2.15. *A measure ${\mathcal{L}_{\mathbb{I}}}^{BG}$ of the expected error of the posterior distribution (as described in Definition 5.2.11) always satisfies Axioms 3.3.2 (quasi-convexity) and 3.3.3 (quasi-concavity).*

Proof. Let $\mathfrak{M}_1$ be an algorithm with range $\mathbb{O}_1$ and $\mathfrak{M}_2$ be an algorithm with range $\mathbb{O}_2$. Let $\mathfrak{M} = \mathfrak{M}_1 \oplus_p \mathfrak{M}_2$. By definition,

$$\begin{aligned}
{\mathcal{L}_{\mathbb{I}}}^{\mathrm{BG}}(\mathfrak{M}) &= \sum_{\omega \in \mathbb{O}_1} \frac{\sum_{D \in \mathbb{I}} \sum_{\widehat{D} \in \mathbb{I}} E(D, \widehat{D}) \;\; P(D) p P[\mathfrak{M}_1(D) = \omega] \;\; P(\widehat{D}) p P(\mathfrak{M}_1(\widehat{D}) = \omega)}{\sum_{D^\star \in \mathbb{I}} P(D^\star) p P(\mathfrak{M}_1(D^\star) = \omega)} \\
&\quad + \sum_{\omega \in \mathbb{O}_2} \frac{\sum_{D \in \mathbb{I}} \sum_{\widehat{D} \in \mathbb{I}} E(D, \widehat{D}) \;\; P(D)(1-p) P(\mathfrak{M}_2(D) = \omega) \;\; P(\widehat{D})(1-p) P(\mathfrak{M}_2(\widehat{D}) = \omega)}{\sum_{D^\star \in \mathbb{I}} P(D^\star)(1-p) P(\mathfrak{M}_2(D^\star) = \omega)} \\
&= p\, {\mathcal{L}_{\mathbb{I}}}^{\mathrm{BG}}(\mathfrak{M}_1) + (1-p)\, {\mathcal{L}_{\mathbb{I}}}^{\mathrm{BG}}(\mathfrak{M}_2)
\end{aligned}$$

□

5.6.12 Proof of Theorem 5.3.2

To prove theorem 5.3.2, we need the following technical result from [18]:

Lemma 5.6.2 ([18])**.** *A loss measure $\mathcal{L}_{\mathbb{I}}$ satisfies Axioms 3.2.1, 3.2.3, and 3.2.4 if and only if $\mathcal{L}_{\mathbb{I}}$ has the form:*

$$\mathcal{L}_{\mathbb{I}}(\mathfrak{M}) = \sum_{\omega \in \mathrm{range}(\mathfrak{M})} f(P(\mathfrak{M}(D_1) = \omega), \ldots, P(\mathfrak{M}(D_n) = \omega))$$

where f is a continuous function that is concave over vectors with no negative components and $f(c\vec{x}) = cf(\vec{x})$ for all nonnegative c.

Now we prove the theorem.

Theorem 5.3.2. *Let $\mathbb{I}$ be finite.*

(i) For every choice of $\mathbb{A}$, prior over $\mathbb{I}$, and error function $\mathcal{E}$ that is bounded from below, the resulting loss measure ${\mathcal{L}_{\mathbb{I}}}^{BDT}$ (Definition 5.3.1) satisfies Axioms 3.2.1, 3.2.3, and 3.2.4.

(ii) If a loss measure $\mathcal{L}_{\mathbb{I}}$ satisfies Axioms 3.2.1, 3.2.3, and 3.2.4 then there exists a prior over $\mathbb{I}$, a set of actions $\mathbb{A}$, and an error function $\mathcal{E}$ (that is bounded from below) such the corresponding loss measure ${\mathcal{L}_{\mathbb{I}}}^{BDT}$ equals $\mathcal{L}_{\mathbb{I}}$.

Proof of (i)**:**
We must first show that ${\mathcal{L}_{\mathbb{I}}}^{\mathrm{BDT}}$ is always finite and then we must show that Lemma 5.6.2 applies. Let a' be any action.

$$ {\mathcal{L}_{\mathbb{I}}}^{\mathrm{BDT}}(\mathfrak{M}) \leq \sum_{\omega \in \mathrm{range}(\mathfrak{M})} \sum_{D \in \mathbb{I}} P(D) P(\mathfrak{M}(D) = \omega)\, \mathcal{E}(D, a') < \infty $$

because $\mathbb{I}$ is a finite set. It is also clear that ${\mathcal{L}_{\mathbb{I}}}^{\mathrm{BDT}}(\mathfrak{M}) \geq \beta > -\infty$, where β is the lower bound on $\mathcal{E}$, so ${\mathcal{L}_{\mathbb{I}}}^{\mathrm{BDT}}$ is finite. To show that Lemma 5.6.2 applies, notice that

$$ {\mathcal{L}_{\mathbb{I}}}^{\mathrm{BDT}}(\mathfrak{M}) = \sum_{\omega \in \mathrm{range}(\mathfrak{M})} \inf_{a \in \mathbb{A}} \sum_{D \in \mathbb{I}} P(D) P(\mathfrak{M}(D) = \omega)\, \mathcal{E}(D, a) $$

and so we must show that the function f defined as

$$ \begin{aligned} & f(P(\mathfrak{M}(D_1) = \omega, \ldots, P(\mathfrak{M}(D_n) = \omega)) \\ = \; & \inf_{a \in \mathbb{A}} \sum_{D \in \mathbb{I}} P(D) P(\mathfrak{M}(D) = \omega)\, \mathcal{E}(D, a) \end{aligned} $$

is continuous, concave over vectors with no negative components and $f(c\vec{x}) = cf(\vec{x})$ for all nonnegative c. The function f is concave because the infimum of linear functions is concave. The other two properties are obvious and therefore, by Lemma 5.6.2, ${\mathcal{L}_{\mathbb{I}}}^{\mathrm{BDT}}$ satisfies Axiom 3.2.3, 3.2.4 and 3.2.1.

Proof of (ii)**:**
Let $\mathcal{L}_{\mathbb{I}}$ be a loss measure satisfying the Axiom 3.2.3, 3.2.4 and 3.2.1. By Lemma 5.6.2 (replacing the f in the lemma with $-F$),

$$ \mathcal{L}_{\mathbb{I}}(\mathfrak{M}) \quad = \quad \sum_{\omega \in \mathrm{range}(\mathfrak{M})} -F(\vec{P}[\mathfrak{M}(\cdot) = \omega]) \tag{5.7} $$

where F is continuous, *convex* over vectors with no negative components, and $cF(\vec{x}) = F(c\vec{x})$ for $c \geq 0$.

Consider the **epigraph** of F, defined as the following subset of $\mathbb{R}^{|\,\mathbb{I}\,|} \times \mathbb{R}$:

$$ \mathrm{epi}(F) = \{(\vec{x}, z) \;:\; z \geq F(\vec{x}) \;\wedge\; \vec{x}[i] \geq 0, i = 1, \ldots, |\,\mathbb{I}\,|\} $$

This is a convex set. Note that for any $\vec{x^*}$ with no negative components, the vector $(\vec{x^*}, F(\vec{x^*}))$ lies the boundary of the epigraph). By the supporting hyperplane theorem,[10] for any given $\vec{x^*}$ there exists a supporting hyperplane defined by the triple $(\vec{a}_{\vec{x^*}}, b_{\vec{x^*}}, h_{\vec{x^*}})$, with the following properties:

$\langle 1 \rangle$ $(\vec{a}_{\vec{x^*}}, b_{\vec{x^*}}) \neq (\vec{0}, 0)$

[10] Stephen Boyd and Lieven Vandenberghe, *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004.

$\langle 2 \rangle$ $\vec{a}_{\vec{x^*}} \cdot \vec{x^*} + b_{\vec{x^*}} F(\vec{x^*}) = h_{\vec{x^*}}$

$\langle 3 \rangle$ $\vec{a}_{\vec{x^*}} \cdot \vec{x} + b_{\vec{x^*}} z \geq h_{\vec{x^*}}$ for all other $(\vec{x}, z) \in \text{epi}(F)$.

Now we need to prove several results about $\vec{a}_{\vec{x^*}}, b_{\vec{x^*}}, h_{\vec{x^*}}$.

Part ii-a: We must show that if $\vec{x^*}$ has only positive components then $b_{\vec{x^*}} > 0$.
For all $s \geq 0$, Property $\langle 3 \rangle$ and the definition of epi(F) implies $\vec{a}_{\vec{x^*}} \cdot \vec{x} + b_{\vec{x^*}}(s + F(x)) \geq h_{\vec{x^*}}$ which is only true if $b_{\vec{x^*}} \geq 0$.

Now, if it were true that $b_{\vec{x^*}} = 0$ then Property $\langle 1 \rangle$ would imply that $\vec{a}_{\vec{x^*}} \neq \vec{0}$. Properties $\langle 3 \rangle$ and $\langle 2 \rangle$ would then imply $\vec{a}_{\vec{x^*}} \cdot \vec{x} \geq h_{\vec{x^*}} = \vec{a}_{\vec{x^*}} \cdot \vec{x^*}$ for all $\vec{x}$ with no negative components. For any $c \geq 0$, substituting $c\vec{x^*}$ for $\vec{x}$, we get $c\vec{a}_{\vec{x^*}} \cdot \vec{x^*} \geq h_{\vec{x^*}} = \vec{a}_{\vec{x^*}} \cdot \vec{x^*}$ for all $c \geq 0$ and this can only be true if $\vec{a}_{\vec{x^*}} \cdot \vec{x^*} = 0$. This means that $\vec{a}_{\vec{x^*}}$ and $\vec{x^*}$ are mutually orthogonal and $h_{\vec{x^*}} = 0$. However, when $\vec{x^*}$ has only positive components then for small enough $\gamma > 0$, the vector $\vec{y} \stackrel{\text{def}}{=} \vec{x^*} - \gamma \vec{a}_{\vec{x^*}}$ also has only positive components but then $\vec{a}_{\vec{x^*}} \cdot \vec{y} = -\gamma ||\vec{a}_{\vec{x^*}}||_2^2 < 0$ (because $b_{\vec{x^*}} = 0$ implies $\vec{a}_{\vec{x^*}} \neq \vec{0}$ by Property $\langle 1 \rangle$). This is now a contradiction of Property $\langle 3 \rangle$ (since we showed $b_{\vec{x^*}} = 0$ implies $h_{\vec{x^*}} = 0$). Thus $b_{\vec{x^*}} > 0$.

Part ii-b: We must show that if $\vec{x^*}$ has only positive components then $h_{\vec{x^*}} = 0$.
Using Property $\langle 2 \rangle$, then Property $\langle 3 \rangle$ (along with properties of F and the definition of epi(F)),

$$\begin{aligned} \vec{a}_{\vec{x^*}} \cdot \vec{x^*} + b_{\vec{x^*}} F(\vec{x^*}) &= h_{\vec{x^*}} \qquad \text{and} \\ \vec{a}_{\vec{x^*}} \cdot c\vec{x^*} + b_{\vec{x^*}} F(c\vec{x^*}) &= c \times \left(\vec{a}_{\vec{x^*}} \cdot c\vec{x^*} + b_{\vec{x^*}} F(\vec{x^*}) \right) \geq h_{\vec{x^*}} \end{aligned}$$

for any $c \geq 0$. This implies that $h_{\vec{x^*}} = 0$.

Part ii-c: We derive a representation of F for vectors $\vec{x}$ with only positive components. For $\vec{x^*}$ with only positive components, $b_{\vec{x^*}} > 0$ and $h_{\vec{x^*}} = 0$ thus, without loss of generality, we may assume $b_{\vec{x^*}} = 1$ (by dividing $b_{\vec{x^*}}$ and $\vec{a}_{\vec{x^*}}$ by the appropriate positive constant). Then when $\vec{x^*}$ has only positive components, we have

$\langle 1^* \rangle$ $\vec{a}_{\vec{x^*}} \cdot \vec{x^*} + F(\vec{x^*}) = 0$

$\langle 2^* \rangle$ $\vec{a}_{\vec{x^*}} \cdot \vec{x} + F(\vec{x}) \geq 0$ for all $\vec{x}$ with no negative components.

Define

$$\begin{aligned} \mathbb{A} &= \{\vec{a}_{\vec{x^*}} : \vec{x^*} \text{ has only positive components}\} \\ \mathcal{E}(D_i, -\vec{a}_{\vec{x^*}}) &= \vec{a}_{\vec{x^*}}[i] \times |\mathbb{I}| \\ P(D_i) &= 1/|\mathbb{I}| \qquad \text{for all } D_i \end{aligned}$$

Now let $\vec{x}$ and $\vec{y}$ be any two vectors with only positive components. Properties $\langle 1^* \rangle$ and $\langle 2^* \rangle$ imply that the corresponding $\vec{a}_{\vec{y}}, \vec{a}_{\vec{x}}, b_{\vec{y}}, b_{\vec{x}}$ satisfy:

$$-F(\vec{x}) = \vec{a}_{\vec{x}} \cdot \vec{x}; \quad -F(\vec{y}) = \vec{a}_{\vec{y}} \cdot \vec{y}; \quad -F(\vec{x}) \leq \vec{a}_{\vec{y}} \cdot \vec{x}$$

$$\Rightarrow \quad -F(\vec{x}) = \inf_{\vec{a}_{\vec{y}} \in \mathbb{A}} \vec{a}_{\vec{y}} \cdot \vec{x} = \inf_{\vec{a}_{\vec{y}} \in \mathbb{A}} \sum_{D_i \in \mathbb{I}} \vec{a}_{\vec{y}}[i]\ \vec{x}[i]$$

$$\Rightarrow \quad -F(\vec{x}) = \inf_{a \in \mathbb{A}} \sum_{D_i \in \mathbb{I}} P(D_i)\vec{x}[i]\, \mathcal{E}(D_i, a) \tag{5.8}$$

Part ii-d: Finishing the proof. Since F is continuous, Equation 5.8 can be extended to all vectors with no negative components. Plugging this into Equation 5.7 will finish the proof once we show that $\mathcal{E}$ is bounded from below.

Suppose $\mathcal{E}$ is *not* bounded from below. Then there is a sequence of (not necessarily distinct) inputs: $D_1, D_2, D_3, \ldots$ and a sequence of actions $a_1, a_2, \ldots$ such that $\mathcal{E}(D_i, a_i) \leq -3^i$. Define $\mathfrak{M}$ with range$(\mathfrak{M}) = \{\omega_*, \omega_1, \omega_2, \omega_3, \ldots\}$ with the property that for $i = 1, 2, \ldots,$

$$\begin{aligned} P(\mathfrak{M}(D) = \omega_i) &= \begin{cases} 3^{-i} & \text{if } D = D_i \\ 0 & \text{otherwise} \end{cases} \\ P(\mathfrak{M}(D) = \omega_*) &= 1 - P(\mathfrak{M}(D) \in \{\omega_1, \omega_2, \ldots\}) \end{aligned}$$

It follows that

$$\begin{aligned} \mathcal{L}_{\mathbb{I}}(\mathfrak{M}) &= -F(\vec{P}[\mathfrak{M}(\cdot) = \omega_*]) + \sum_{i=1}^{\infty} -F(\vec{P}[\mathfrak{M}(\cdot) = \omega_i]) \\ &= -F(\vec{P}[\mathfrak{M}(\cdot) = \omega_*]) + \sum_{i=1}^{\infty} \inf_{a \in \mathbb{A}} \sum_{D \in \mathbb{I}} P(D) P(\mathfrak{M}(D) = \omega_i)\, \mathcal{E}(D, a) \\ &\leq -F(\vec{P}[\mathfrak{M}(\cdot) = \omega_*]) + \sum_{i=1}^{\infty} \sum_{D \in \mathbb{I}} P(D) P(\mathfrak{M}(D) = \omega_i)\, \mathcal{E}(D, a_i) \\ &\leq -F(\vec{P}[\mathfrak{M}(\cdot) = \omega_*]) + \sum_{i=1}^{\infty} (1/|I|)(3^{-i})(-3^i) = -\infty \end{aligned}$$

This is a contradiction, so $\mathcal{E}$ is bounded from below. □

5.6.13 Proof of Theorem 5.3.3

Proof. $\mathcal{L}_{\mathbb{I}}^{\text{BDT}}$ has the form.

$$\mathcal{L}_{\mathbb{I}}(\mathfrak{M}) = \sum_{\omega \in \text{range}(\mathfrak{M})} f(P(\mathfrak{M}(D_1) = \omega), \ldots, P(\mathfrak{M}(D_n) = \omega))$$

where f has the following property (for all vectors $\vec{x}$ with no negative components and all $c \geq 0$):

$$f(c\vec{x}) = cf(\vec{x})$$

Let $\mathfrak{M}_1$ be an algorithm with range $\mathbb{O}_1$ and $\mathfrak{M}_2$ be an algorithm with range $\mathbb{O}_2$. Let $\mathfrak{M} =$

$\mathfrak{M}_1 \oplus_p \mathfrak{M}_2$. Applying the above property,

$$\begin{aligned}
{\mathcal{L}_{\mathbb{I}}}^{BDT}(\mathfrak{M}) &= \sum_{\omega \in \text{range}(\mathfrak{M})} f\Big(P(\mathfrak{M}(D_1)=\omega),\ \ P(\mathfrak{M}(D_2)=\omega), \dots\Big) \\
&= \sum_{\omega \in \mathbb{O}_1} f\Big(pP(\mathfrak{M}_1(D_1)=\omega),\ \ pP(\mathfrak{M}_1(D_2)=\omega), \dots\Big) \\
&\quad + \sum_{\omega \in \mathbb{O}_2} f\Big((1-p)P(\mathfrak{M}_2(D_1)=\omega),\ \ (1-p)P(\mathfrak{M}_2(D_2)=\omega), \dots\Big) \\
&= p \sum_{\omega \in \mathbb{O}_1} f\Big(P(\mathfrak{M}_1(D_1)=\omega),\ \ P(\mathfrak{M}_1(D_2)=\omega), \dots\Big) \\
&\quad + (1-p) \sum_{\omega \in \mathbb{O}_2} f\Big(P(\mathfrak{M}_2(D_1)=\omega),\ \ P(\mathfrak{M}_2(D_2)=\omega), \dots\Big) \\
&= p\,{\mathcal{L}_{\mathbb{I}}}^{BDT}(\mathfrak{M}_1) + (1-p)\,{\mathcal{L}_{\mathbb{I}}}^{BDT}(\mathfrak{M}_2)
\end{aligned}$$

□

5.6.14 Proof of Theorem 5.4.2

Lemma 5.4.2. *For any utility measure $\mu_{\mathbb{I}}$, sufficing produces a utility measure $\mu_{\mathbb{I}}^{\diamond}$ that always satisfies Axiom 3.2.1 (sufficiency). It is also the "smallest repair" of $\mu_{\mathbb{I}}$ in the following sense: if some utility measure ${\mu_{\mathbb{I}}}'$ satisfies Axiom 3.2.1 and ${\mu_{\mathbb{I}}}'(\mathfrak{M}) \geq \mu_{\mathbb{I}}(\mathfrak{M})$ for all $\mathfrak{M}$ then ${\mu_{\mathbb{I}}}'(\mathfrak{M}) \geq \mu_{\mathbb{I}}^{\diamond}(\mathfrak{M})$ for all $\mathfrak{M}$. As a result, $\mu_{\mathbb{I}} \equiv \mu_{\mathbb{I}}^{\diamond}$ if and only if $\mu_{\mathbb{I}}$ satisfies Axiom 3.2.1.*

Proof. By definition, $\mu^{\diamond}(\mathfrak{M}_1) = \sup_{\mathcal{A}} \mu(\mathcal{A} \circ \mathfrak{M}_1) \leq \mu'(\mathcal{A} \circ \mathfrak{M}_1) \leq \mu'(\mathfrak{M}_1)$. □

5.6.15 Proof of Theorem 5.4.5

Theorem 5.4.5. *Let $\mathbb{I}$ be finite and let the quality function Q be bounded. Then ${\mu_{\mathbb{I}}}^{SM\diamond}$ satisfies Axiom 3.3.3 (quasi-concavity).*

Proof. By Theorem 5.4.4,

$$\begin{aligned}
{\mu_{\mathbb{I}}}^{\text{SM}\diamond}(\mathfrak{M}_1 \oplus_p \mathfrak{M}_2) &= \inf_P {\mu_{\mathbb{I}}}^{\text{BDT}}(\mathfrak{M}_1 \oplus_p \mathfrak{M}_2; P, Q) \\
&\geq \inf_P \min\{{\mu_{\mathbb{I}}}^{\text{BDT}}(\mathfrak{M}_1; P, Q),\ \ {\mu_{\mathbb{I}}}^{\text{BDT}}(\mathfrak{M}_2; P, Q)\} \\
&\quad \text{(Due to quasi-concavity of } {\mu_{\mathbb{I}}}^{\text{BDT}}\text{, Theorem 5.3.3)} \\
&\geq \min\{\inf_P {\mu_{\mathbb{I}}}^{\text{BDT}}(\mathfrak{M}_1; P, Q),\quad \inf_P {\mu_{\mathbb{I}}}^{\text{BDT}}(\mathfrak{M}_1; P, Q)\} \\
&= \min\{{\mu_{\mathbb{I}}}^{\text{SM}\diamond}(\mathfrak{M}_1),\ \ {\mu_{\mathbb{I}}}^{\text{SM}\diamond}(\mathfrak{M}_2)\}
\end{aligned}$$

□

5.6.16 Proof of Theorem 5.4.6

To prove the theorem, we need the following results. As a reminder of notation, recall that Q is a function with the signature $\mathbb{I} \times \mathbb{O}^* \to \mathbb{R}$ and

$$\begin{aligned}
\mu_{\mathbb{I}}^{\mathrm{SE}\diamond}(\mathfrak{M}) &= \sup_{\mathcal{A}} \mu_{\mathbb{I}}^{\mathrm{SE}}(\mathcal{A} \circ \mathfrak{M}) \\
&= \sup_{\mathcal{A}} \sum_{D \in \mathbb{I}} \sum_{\omega \in \mathbb{O}^*} Q(D,\omega) P(D) P(\mathcal{A}(\mathfrak{M}(D)) = \omega) \\
&= \sup_{\mathcal{A}} \sum_{D \in \mathbb{I}} \sum_{\eta \in \mathrm{range}(\mathfrak{M})} \sum_{\omega \in \mathbb{O}^*} Q(D,\omega)\ P(D) P(\mathfrak{M}(D) = \eta) P(\mathcal{A}(\eta) = \omega)
\end{aligned}$$

so that the optimization considers all randomized algorithms $\mathcal{A}$ with $\mathrm{range}(\mathcal{A}) \subseteq \mathbb{O}^*$ and $\mathrm{domain}(\mathcal{A}) \supseteq \mathrm{range}(\mathfrak{M})$.

Since $\mathbb{I}$ is finite, let $\mathbb{I} = \{D_1, \ldots, D_N\}$. Throughout the proof, we will focus on special set of algorithm $\mathfrak{M}_{p_1,\ldots,p_N}$ defined as follows. Let $\eta^*, \eta_1, \eta_2, \ldots, \eta_N$ be distinct. Let $p_1, p_2, \ldots, p_N$ be numbers between 0 and 1 (i.e. probabilities). Define

$$\begin{aligned}
P(\mathfrak{M}_{p_1,\ldots,p_N}(D_i) = \eta_i) &= p_i \qquad \text{for } i = 1, \ldots N \\
P(\mathfrak{M}_{p_1,\ldots,p_N}(D_i) = \eta_\ell) &= 0 \qquad \text{for } i = 1, \ldots, N \text{ and } i \neq \ell \\
P(\mathfrak{M}_{p_1,\ldots,p_N}(D_i) = \eta^*) &= 1 - p_i \qquad \text{for } i = 1, \ldots N
\end{aligned}$$

When $p_1 = p_2 = \cdots = p_N = 1$, the resulting algorithm functions like an identity: from the output, we know exactly what the input was (it never outputs η^*, and if the output is η_i then the input must be D_i). As a shorthand, we denote this algorithm as $\mathfrak{M}_{\mathrm{identity}}$ and refer to it as the *identity algorithm*.

We need some intermediate results. The first result is about continuity:

Lemma 5.6.3. *Let Q be bounded. Then, $\mu_{\mathbb{I}}^{SM\diamond}(\mathfrak{M}_{p_1,\ldots,p_N})$ is continuous in p_1, ...,p_N.*

Proof. Due to the symmetry structure of $\mathfrak{M}_{p_1,\ldots,p_N}$, we only need to prove $\mu_{\mathbb{I}}^{\mathrm{SM}}(\mathfrak{M}_{p_1,\ldots,p_N})$ is continuous in p_1. Let K be an upper bound on $|Q(D,\omega)|$. Recall that $\mu_{\mathbb{I}}^{\mathrm{SM}\diamond}(\mathfrak{M}) = \sup_{\mathcal{A}} \mu_{\mathbb{I}}^{\mathrm{SM}}(\mathcal{A} \circ \mathfrak{M})$. Note that for any $\mathcal{A}$, $|\,\mu_{\mathbb{I}}^{\mathrm{SM}}(\mathcal{A} \circ \mathfrak{M}_{p_1,p_2,\ldots,p_N}) - \mu_{\mathbb{I}}^{\mathrm{SM}}(\mathcal{A} \circ \mathfrak{M}_{p_1+\delta,p_2,\ldots,p_N})| \leq 2\delta K$. Thus, for an arbitrary $\epsilon > 0$, choose an $\mathcal{A}_1$ such that $\mu_{\mathbb{I}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{p_1,\ldots,p_N}) \leq \mu_{\mathbb{I}}^{\mathrm{SM}}(\mathcal{A}_1 \circ \mathfrak{M}_{p_1,\ldots,p_N}) + \epsilon$ and an $\mathcal{A}_2$ such that $\mu_{\mathbb{I}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{p_1+\delta,p_2,\ldots,p_N}) \leq \mu_{\mathbb{I}}^{\mathrm{SM}}(\mathcal{A}_2 \circ \mathfrak{M}_{p_1+\delta,p_2,\ldots,p_N}) + \epsilon$. Then

$$\begin{aligned}
\mu_{\mathbb{I}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{p_1,\ldots,p_N}) &\leq \mu_{\mathbb{I}}^{\mathrm{SM}}(\mathcal{A}_1 \circ \mathfrak{M}_{p_1,\ldots,p_N}) + \epsilon \\
&\leq \mu_{\mathbb{I}}^{\mathrm{SM}}(\mathcal{A}_1 \circ \mathfrak{M}_{p_1+\delta,p_2\ldots,p_N}) + \epsilon + 2\delta K \\
&\leq \mu_{\mathbb{I}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{p_1+\delta,p_2\ldots,p_N}) + \epsilon + 2\delta K
\end{aligned}$$

and a similar argument yields $\mu_{\mathbb{I}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{p_1+\delta,p_2\ldots,p_N}) \leq \mu_{\mathbb{I}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{p_1,p_2\ldots,p_N}) + \epsilon + 2\delta K$. Since ϵ is

arbitrary, we have

$$|\,{\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{p_1,p_2\ldots,p_N}) - {\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{p_1+\delta,p_2\ldots,p_N})| \leq 2\delta K$$

thus showing that utility of $\mathfrak{M}_{p_1,p_2\ldots,p_N}$ is continuous in p_1. □

The second result allows as to replace $\sup_{\mathcal{A}}$ with $\max_{\mathcal{A}}$:

Lemma 5.6.4. *Let $\mathbb{I}$ and $\mathbb{O}^*$ be finite. Let $\mathfrak{S} = \{\mathfrak{M}_{p_1,\ldots,p_N} | \forall i, p_i \geq 0$ and $\sum_{i=1}^{N} p_i = 1\}$. If the domain of ${\mu_{\mathbb{I}}}^{SM\diamond}$ is restricted to $\mathfrak{S}$ (i.e. ${\mu_{\mathbb{I}}}^{SM\diamond} : \mathfrak{S} \to R$), then maximum exists (i.e. ${\mu_{\mathbb{I}}}^{SM\diamond}(\mathfrak{M}) = \max_{\mathcal{A}} {\mu_{\mathbb{I}}}^{SM}(\mathcal{A} \circ \mathfrak{M})$).*

Proof. We will break the optimization problem (i.e. $\sup_{\mathcal{A}} {\mu_{\mathbb{I}}}^{\mathrm{SM}}(\mathcal{A} \circ \mathfrak{M})$) into optimization problem per output. Recall that

$$\begin{aligned}
{\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}) &= \sup_{\mathcal{A}} {\mu_{\mathbb{I}}}^{\mathrm{SM}}(\mathcal{A} \circ \mathfrak{M}) \\
&= \sup_{\mathcal{A}} \inf_{D \in \mathbb{I}} \sum_{\omega \in \mathbb{O}^*} Q(D,\omega) P(\mathcal{A}(\mathfrak{M}(D)) = \omega) \\
&= \sup_{\mathcal{A}} \inf_{D \in \mathbb{I}} \sum_{\omega \in \mathbb{O}^*} Q(D,\omega) \sum_{\eta \in \mathrm{range}(\mathfrak{M})} P(\mathcal{A}(\eta) = \omega) P(\mathfrak{M}(D) = \eta) \\
&= \sup_{\mathcal{A}} \inf_{D \in \mathbb{I}} \sum_{\eta \in \mathrm{range}(\mathfrak{M})} P(\mathfrak{M}(D) = \eta) \sum_{\omega \in \mathbb{O}^*} Q(D,\omega) P(\mathcal{A}(\eta) = \omega)
\end{aligned}$$

Notice that $\forall i$, η_i is only generated when the input is D_i. That is $\forall i \neq j, P(\mathfrak{M}(D_i) = \eta_j) = 0$. Hence, upon on seeing η_i, the response algorithm (i.e. $P(\mathcal{A}(\eta_i) = .)$) only has effect on quality of D_i. Let

$$\omega_i = \max_{\omega} Q(D_i, \omega) \qquad \text{for } i = 1, \ldots N.$$

There exists a best response algorithm $\mathcal{A}$ such that

$$P(\mathcal{A}(\eta_i) = \omega_i) = 1 \qquad \text{for } i = 1, \ldots N.$$

Now, what left is what $\mathcal{A}$ should do when it see η^* (i.e. $P(\mathcal{A}(\eta^*) = .)$).

Let

$$\begin{aligned}
q_i(\mathcal{A}) = \sum_{\omega} P(\mathcal{A}(\eta^*) = \omega) Q(D_i, \omega) \qquad &\text{for } i = 1, \ldots N \text{ and} \\
\vec{q}(\mathcal{A}) = \langle q_1(\mathcal{A}), ..., q_N(\mathcal{A}) \rangle &
\end{aligned}$$

We argue that the range of $\vec{q}$ is a compact convex set. We can view the function Q as a matrix $G = \{g_{i,\omega} = Q(D_i, \omega) | \text{ for } i = 1, \ldots, N \text{ and } \forall \omega \in \mathbb{O}^*\}$ where rows correspond to inputs (with the order $D_1, ..., D_N$) and columns correspond to the outputs (with an arbitrary order). Now,

we can construct a convex hull $\mathcal{H}$ from columns of G. One can easily verify that range of $\vec{q}$ is exactly $\mathcal{H}$. Recall that, upon seeing η_i, a best response generates ω_i. Hence, we can reduce the optimization problem as follows.

$$\begin{aligned}\mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}) &= \sup_{\mathcal{A}} \inf_{i\in\{1,\ldots,N\}} \sum_{\eta\in\mathrm{range}(\mathfrak{M})} P(\mathfrak{M}(D_i)=\eta) \sum_{\omega\in\mathbb{O}_\omega} Q(D_i,\omega)P(\mathcal{A}(\eta)=\omega) \\ &= \sup_{\vec{q}\in\mathcal{H}} \min_{i\in\{1,\ldots,N\}} (1-p_i)q_i + p_i Q(D_i,\omega_i)\end{aligned}$$

Because $\mathcal{H}$ is compact and $\min_{i\in\{1,\ldots,N\}}(1-p_i)q_i+p_iQ(D_i,\omega_i)$ is continuous in $\vec{q}$, the maximum must exist. Hence, we can replace $\sup_{\vec{q}\in\mathcal{H}}$ with $\max_{\vec{q}\in\mathcal{H}}$. This implies that the algorithm $\mathcal{A}$ maximizing $\mu_{\mathbb{I}}{}^{\mathrm{SM}}(\mathcal{A}\circ\mathfrak{M})$ exists. Hence, we can replace $\sup_{\mathcal{A}}$ with $\max_{\mathcal{A}}$. □

The third result is about, if $\mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}$ is not constant, then there exists a special subset of inputs $\mathbb{I}^*\subseteq\mathbb{I}$:

Lemma 5.6.5. *If $\mu_{\mathbb{I}}{}^{SM\diamond}$ is not constant, then there exists $\mathbb{I}^*\subseteq\mathbb{I}$ such that*

- *An algorithm $\mathfrak{M}_{\mathbb{I}^*}$ behaves as generating η^* when the input is $D_i\in\mathbb{I}^*$ (i.e. $\forall D_i\in\mathbb{I}^*, p_i=0$) and generating η_i when the input is $D_i\in\mathbb{I}\setminus\mathbb{I}^*$ (i.e. $\forall D_i\notin\mathbb{I}^*, p_i=1$) . Then, $\mathfrak{M}_{\mathbb{I}^*}$ does not achieve maximum utility (i.e. $\mu_{\mathbb{I}}{}^{SM\diamond}(\mathfrak{M}_{\mathbb{I}^*}) < \mu_{\mathbb{I}}{}^{SM\diamond}(\mathfrak{M}_{identity})$).*
- *An algorithm $\mathfrak{M}_{\mathbb{I}^*\setminus\{D_k\}}$ for some $D_k\in\mathbb{I}^*$ behaves as generating an η^* when the input is $D_i\in\mathbb{I}^*\setminus\{D_k\}$ (i.e. $\forall D_i\in\mathbb{I}^*\setminus\{D_k\}, p_i=0$) and generating η_k when the input is D_k (i.e. $p_k=1$) and generating η_i when the input is $D_i\in\mathbb{I}\setminus\mathbb{I}^*$ (i.e. $\forall D_i\notin\mathbb{I}^*, p_i=1$). Then, $\mathfrak{M}_{\mathbb{I}^*\setminus\{D_k\}}$ achieves maximum utility (i.e. $\mu_{\mathbb{I}}{}^{SM\diamond}(\mathfrak{M}_{\mathbb{I}^*\setminus\{D_k\}}) = \mu_{\mathbb{I}}{}^{SM\diamond}(\mathfrak{M}_{identity})$).*

Proof. We prove this lemma by giving an procedure to find $\mathbb{I}^*$. First we will show that, if $|\,\mathbb{I}^*\,|=1$, then $\mathfrak{M}_{\mathbb{I}^*}$ and $\mathfrak{M}_{\mathbb{I}^*\setminus\{D_k\}}$ acts like $\mathfrak{M}_{\mathrm{identity}}$. Assume that $\mathbb{I}^*=\{D_k\}$. $\mathfrak{M}_{\mathbb{I}^*}$ generates η^* only if the input is D_k and,$\forall i\neq k$, generates η_i only if the input is D_i. Hence, from the output, one know exactly what the input is. One can verify $\mathfrak{M}_{\mathbb{I}^*\setminus\{D_k\}}$ behaves exactly like $\mathfrak{M}_{\mathbb{I}^*}$. Now, let $n=2$, consider all possible $\mathbb{I}^*\subseteq\mathbb{I}$ and $|\,\mathbb{I}^*\,|=n$. If there exists $\mathbb{I}^*$ such that $\mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\mathbb{I}^*}) < \mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\mathrm{identity}})$, then we are done. If not, we increase n by one and check whether there exists $\mathfrak{M}_{\mathbb{I}^*}$ and $\mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\mathbb{I}^*}) < \mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\mathrm{identity}})$. Note that the second condition (i.e. $\mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\mathbb{I}^*\setminus\{D_k\}}) = \mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\mathrm{identity}})$) is checked when $n=n-1$. We argue that this algorithm stops for some $n\leq N$. First, note that $\mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}$ satisfies Axiom 3.2.1 (sufficiency). Hence, if $\mathbb{I}_1^*\subseteq\mathbb{I}_2^*$, then $\mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\mathbb{I}_1^*}) \geq \mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\mathbb{I}_2^*})$. Second, $\mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}$ is not constant and $\mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\mathbb{I}^*}) < \mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\mathrm{identity}})$ when $\mathbb{I}^*=\mathbb{I}$. Hence, the algorithm must stop at some $n\leq|\,\mathbb{I}\,|=N$. □

Let $\mathbb{I}^*\subseteq\mathbb{I}$ satisfies the constraints described in Lemma 5.6.5. Let $n=|\,\mathbb{I}^*\,|$ and, without loss of generality, assume that $\mathbb{I}^*=\{D_1,...,D_n\}$. From here, we will focus on special set of algorithm $\mathfrak{M}_{p_1,\ldots,p_n}$ defined as follows. Let $\eta^*,\eta_1,\eta_2,...,\eta_N$ be distinct. Let $p_1,...,p_n$ be numbers between 0 and 1 (i.e. probabilities). Define

$$P(\mathfrak{M}_{p_1,\ldots,p_n}(D_i)=\eta_i)=p_i \qquad \text{for } i=1,\ldots n$$

$$
\begin{aligned}
P(\mathfrak{M}_{p_1,\ldots,p_n}(D_i) = \eta^*) &= 1 - p_i \qquad \text{for } i = 1, \ldots n \\
P(\mathfrak{M}_{p_1,\ldots,p_n}(D_i) = \eta_i) &= 1 \qquad \text{for } i = n+1, \ldots N \\
P(\mathfrak{M}_{p_1,\ldots,p_n}(D_i) = \eta_\ell) &= 0 \qquad \text{for } i = 1, \ldots, N \text{ and } i \neq \ell
\end{aligned}
$$

The fourth result is about the quality of $D_i \in \mathbb{I}^*$. The quality of D_i preserved by algorithm $\mathfrak{M}$ under the response algorithm $\mathcal{A}$ is defined as

$$
q_i^{\mathcal{A} \circ \mathfrak{M}} = \sum_{\eta \in \text{range}(\mathfrak{M})} P(\mathfrak{M}(D) = \eta) \sum_{\omega \in \mathbb{O}_\omega} P(\mathcal{A}(\eta) = \omega) Q(D_i, \omega)
$$

Lemma 5.6.6. *Let $\mathbb{I}^* \subseteq \mathbb{I}$ satisfies the constraints described in Lemma 5.6.5. Let $\mathfrak{M}$ be an algorithm such that $\mathfrak{M}$ generates η_i when the input is $D_i \in \mathbb{I} \setminus \mathbb{I}^*$. Let $\mathcal{A}$ be a response algorithm of $\mathfrak{M}$. If ${\mu_{\mathbb{I}}}^{SM}(\mathcal{A} \circ \mathfrak{M}) < {\mu_{\mathbb{I}}}^{SM\diamond}(\mathfrak{M}_{identity})$ and $\exists D_\alpha, D_\beta \in \mathbb{I}^*$ $q_\alpha^{\mathcal{A} \circ \mathfrak{M}} > q_\beta^{\mathcal{A} \circ \mathfrak{M}}$, then there exists $\mathcal{A}^*$ such that ${\mu_{\mathbb{I}}}^{SM}(\mathcal{A}^* \circ \mathfrak{M}) > {\mu_{\mathbb{I}}}^{SM}(\mathcal{A} \circ \mathfrak{M})$.*

Proof. Let $\mathfrak{M}$ be an algorithm such that $\mathfrak{M}$ generates η_i when the input is $D_i \in \mathbb{I} \setminus \mathbb{I}^*$ and ${\mu_{\mathbb{I}}}^{\text{SM}\diamond}(\mathfrak{M}) < {\mu_{\mathbb{I}}}^{\text{SM}\diamond}(\mathfrak{M}_{\text{identity}})$. Let $\mathcal{A}$ be a response algorithm and ${\mu_{\mathbb{I}}}^{\text{SM}}(\mathcal{A} \circ \mathfrak{M}) < {\mu_{\mathbb{I}}}^{\text{SM}\diamond}(\mathfrak{M}_{\text{identity}})$ and there exists $D_\alpha, D_\beta \in \mathbb{I}^*$ and $q_\alpha^{\mathcal{A} \circ \mathfrak{M}} > q_\beta^{\mathcal{A} \circ \mathfrak{M}} \geq {\mu_{\mathbb{I}}}^{\text{SM}}(\mathcal{A} \circ \mathfrak{M})$.

Let $\mathfrak{M}_1$ behave as generating an unique η_α for some $D_\alpha \in \mathbb{I}^*$ and generating an η^* when the input $D_i \in \mathbb{I}^* \setminus \{D_\alpha\}$. Let $\mathcal{A}_1^* = \arg\max_{\mathcal{A}}({\mu_{\mathbb{I}}}^{\text{SM}}(\mathcal{A} \circ \mathfrak{M}_1))$. Simple calculation yields that, for $i = 1, ..., n$ and $i \neq \alpha$,

$$
\begin{aligned}
q_i^{\mathcal{A}_1^* \circ \mathfrak{M}_1} &= \sum_{\eta \in \text{range}(\mathfrak{M}_1)} P(\mathfrak{M}_1(D_i) = \eta) \sum_{\omega \in \mathbb{O}^*} P(\mathcal{A}_1^*(\eta) = \omega) Q(D_i, \omega) \\
&= P(\mathfrak{M}_1(D_i) = \eta^*) \sum_{\omega \in \mathbb{O}^*} P(\mathcal{A}_1^*(\eta^*) = \omega) Q(D_i, \omega) \\
&= \sum_{\omega \in \mathbb{O}^*} P(\mathcal{A}_1^*(\eta^*) = \omega) Q(D_i, \omega)
\end{aligned}
$$

Recall that $\mathfrak{M}_1$ achieves maximum utility (i.e. ${\mu_{\mathbb{I}}}^{\text{SM}}(\mathcal{A}_1^* \circ \mathfrak{M}_1) = {\mu_{\mathbb{I}}}^{\text{SM}\diamond}(\mathfrak{M}_{\text{identity}})$). This implies that

$$
\sum_{\omega \in \mathbb{O}^*} P(\mathcal{A}_1^*(\eta^*) = \omega) Q(D_i, \omega) \geq {\mu_{\mathbb{I}}}^{\text{SM}\diamond}(\mathfrak{M}_{\text{identity}}) \qquad \text{for } i = 1, \ldots n \text{ and } i \neq \alpha
$$

We introduce shorthand notations as follows.

$$
\begin{aligned}
x_1 &= \sum_{\omega \in \mathbb{O}^*} P(\mathcal{A}_1^*(\eta^*) = \omega) Q(D_\alpha, \omega) \\
x_2 &= {\mu_{\mathbb{I}}}^{\text{SM}}(\mathcal{A} \circ \mathfrak{M}) \\
x_3 &= q_\alpha^{\mathcal{A} \circ \mathfrak{M}}
\end{aligned}
$$

Recall that $x_2 < x_3$. Now, we consider the cases where $x_1 \leq x_2$ and $x_1 > x_2$.

Now, consider $x_1 \leq x_2$. Let $p = \frac{(x_2+x_3)/2-x_1}{x_3-x_1}$. Because $x_1 \leq x_2 < x_3$, p is a positive number smaller than 1. Let $\mathcal{A}^*$ behaves as $\mathcal{A}_1^*$ when the input is η_i for $i = n+1, ..., N$ and behaves as follows otherwise. With probability p, $\mathcal{A}^*$ behaves as $\mathcal{A}$ and, with probabilities $1-p$, $\mathcal{A}^*$ behaves as $\mathcal{A}_1^*$ with input η^* (i.e. generating ω with probability $P(\mathcal{A}_1^*(\eta^*) = \omega$ for all $\omega \in \mathbb{O}^*$). Simple calculation yields the following results.

$$q_i^{\mathcal{A}^* \circ \mathfrak{M}} = \sum_{\omega \in \mathbb{O}^*} P(\mathcal{A}_1^*(\eta_i) = \omega) Q(D_i, \omega) \geq {\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\mathrm{identity}}) > {\mu_{\mathbb{I}}}^{\mathrm{SM}}(\mathcal{A} \circ \mathfrak{M}) \qquad \text{for } i = n+1, \ldots N$$

$$q_i^{\mathcal{A}^* \circ \mathfrak{M}} \geq p\, {\mu_{\mathbb{I}}}^{\mathrm{SM}}(\mathcal{A} \circ \mathfrak{M}) + (1-p)\, {\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\mathrm{identity}}) > {\mu_{\mathbb{I}}}^{\mathrm{SM}}(\mathcal{A} \circ \mathfrak{M}) \qquad \text{for } i = 1, \ldots n \text{ and } i \neq \alpha$$

$q_\alpha^{\mathcal{A}^* \circ \mathfrak{M}}$ can be computed as follows.

$$\begin{aligned} q_\alpha^{\mathcal{A}^* \circ \mathfrak{M}} &= p x_3 + (1-p) x_1 \\ &= \frac{(x_2+x_3)/2 - x_1}{x_3 - x_1} x_3 + (1 - \frac{(x_2+x_3)/2 - x_1}{x_3 - x_1}) x_1 \\ &= \frac{x_2 + x_3}{2} = \frac{q_{\alpha, \mathcal{A} \circ \mathfrak{M}} + {\mu_{\mathbb{I}}}^{\mathrm{SM}}(\mathcal{A} \circ \mathfrak{M})}{2} > {\mu_{\mathbb{I}}}^{\mathrm{SM}}(\mathcal{A} \circ \mathfrak{M}) \end{aligned}$$

Hence, we get ${\mu_{\mathbb{I}}}^{\mathrm{SM}}(\mathcal{A}^* \circ \mathfrak{M}) > {\mu_{\mathbb{I}}}^{\mathrm{SM}}(\mathcal{A} \circ \mathfrak{M})$.

Now, consider $x_1 > x_2$. Let $\mathcal{A}^*$ be an response algorithm as defined earlier but with $p = 0$. Simple calculation yields that

$$q_i^{\mathcal{A}^* \circ \mathfrak{M}} \geq {\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\mathrm{identity}}) > {\mu_{\mathbb{I}}}^{\mathrm{SM}}(\mathcal{A} \circ \mathfrak{M}) \qquad \text{for } i = n+1, \ldots N$$

$$q_i^{\mathcal{A}^* \circ \mathfrak{M}} = \sum_{\omega \in \mathbb{O}^*} P(\mathcal{A}_1^*(\eta^*) = \omega) Q(D_i, \omega) \geq \min(x_1, {\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\mathrm{identity}})) > {\mu_{\mathbb{I}}}^{\mathrm{SM}}(\mathcal{A} \circ \mathfrak{M}) \qquad \text{for } i = 1, \ldots n$$

Hence, we get ${\mu_{\mathbb{I}}}^{\mathrm{SM}}(\mathcal{A}^* \circ \mathfrak{M}) > {\mu_{\mathbb{I}}}^{\mathrm{SM}}(\mathcal{A} \circ \mathfrak{M})$. □

Combining Lemma 5.6.6 and 5.6.4, we can conclude that if $\mathcal{A}$ maximizes ${\mu_{\mathbb{I}}}^{\mathrm{SM}}(\mathcal{A} \circ \mathfrak{M}_{p_1,\ldots,p_n})$ and ${\mu_{\mathbb{I}}}^{\mathrm{SM}}(\mathcal{A} \circ \mathfrak{M}_{p_1,\ldots,p_n}) < {\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\mathrm{identity}})$, then we must have for $i = 1, ..., n$ and for $\ell = 1, ..., n$, $q_i^{\mathcal{A} \circ \mathfrak{M}_{p_1,\ldots,p_n}} = q_\ell^{\mathcal{A} \circ \mathfrak{M}_{p_1,\ldots,p_n}} = {\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{p_1,\ldots,p_n})$.

Now, we are ready to prove Theorem 5.4.6.

Theorem 5.4.6. *Let $\mathbb{I}$ and $\mathbb{O}^*$ be finite and let the quality function $Q : \mathbb{I} \times \mathbb{O}^* \to \mathbb{R}$ be bounded. Then ${\mu_{\mathbb{I}}}^{SM\diamond}$ satisfies Axiom 3.3.2 (quasi-convexity) if and only if it is constant.*

Proof. The if direction is easy to see. Now, we prove the only if direction. First, recall that if ${\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}$ is not constant, then there exists $\mathbb{I}^*$ satisfying constraints described in Lemma 5.6.5. Recall that $|\mathbb{I}| = N$, $|\mathbb{I}^*| = n$. Recall that we only focus on algorithms $\mathfrak{M}_{p_1,\ldots,p_n}$ defined as follows:

$$P(\mathfrak{M}_{p_1,\ldots,p_n}(D_i) = \eta_i) = p_i \qquad \text{for } i = 1, \ldots n$$

$$P(\mathfrak{M}_{p_1,\ldots,p_n}(D_i) = \eta^*) = 1 - p_i \qquad \text{for } i = 1, \ldots n$$

$$P(\mathfrak{M}_{p_1,\ldots,p_n}(D_i) = \eta_i) = 1 \qquad \text{for } i = n+1, \ldots N$$

$$P(\mathfrak{M}_{p_1,\ldots,p_n}(D_i) = \eta_\ell) = 0 \qquad \text{for } i = 1,\ldots,N \text{ and } i \neq \ell$$

Recall that, in the proof of Lemma 5.6.4, the optimization problem (i.e. $\mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}) = \max_{\mathcal{A}} \mu_{\mathbb{I}}{}^{\mathrm{SM}}(\mathcal{A} \circ \mathfrak{M})$) can be broken into optimization problems per output η. We show that upon seeing η_i, a response algorithm (i.e. $P(\mathcal{A}(\eta_i) = .)$) has only effect on quality of D_i. Let

$$\omega_i = \max_{\omega \in \mathbb{O}^*} Q(D_i, \omega) \qquad \text{for } i = 1,\ldots N.$$

There exists a best response algorithm $\mathcal{A}$ (i.e. $\mathcal{A}$ maximizes $\mu_{\mathbb{I}}{}^{\mathrm{SM}}(\mathcal{A} \circ \mathfrak{M})$) such that $P(\mathcal{A}(\eta_i)) = \omega_i) = 1$ for $i = 1,\ldots,N$. In the following, we restrict $\mathcal{A}$ such that $P(\mathcal{A}(\eta_i) = \omega_i) = 1$ for $i = 1,\ldots,N$.

Let $\mathfrak{M}_\alpha$ be an algorithm where

$$\begin{aligned} p_1 &= \alpha \\ p_i &= 0 \qquad \text{for } i = 2,\ldots N \end{aligned}$$

Let $\mathcal{A}^*_\alpha$ be the best response algorithm of $\mathfrak{M}_\alpha$ (i.e. $\mathcal{A}^*_\alpha = \arg\max_{\mathcal{A}} \mu_{\mathbb{I}}{}^{\mathrm{SM}}(\mathcal{A} \circ \mathfrak{M}_\alpha)$). Now, we will show that there exists some α such that, upon seeing η^*, $\mathcal{A}^*_\alpha$ generates ω^*_1 and ω^*_2 with positive probabilities and, without loss of generality, $Q(D_1, \omega^*_1) > Q(D_1, \omega^*_2)$.

Denote

$$q_{i,\eta}^{\mathcal{A}^*_\alpha} = \sum_{\omega \in \mathrm{range}_{\mathfrak{M}_\alpha}} P(\mathcal{A}^*_\alpha(\eta) = \omega) Q(D_i, \omega)$$

$q_{i,\eta}^{\mathcal{A}^*_\alpha}$ is the average quality of D_i achieved by the response algorithm $\mathcal{A}^*_\alpha$ upon seeing η.

Recall that $q_i^{\mathcal{A}^*_\alpha \circ \mathfrak{M}_\alpha}$ is the overall quality of D_i achieved by $\mathfrak{M}_\alpha$ after the best response algorithm $\mathcal{A}^*_\alpha$. Hence, we have

$$\begin{aligned} q_i^{\mathcal{A}^*_\alpha \circ \mathfrak{M}_\alpha} &= \sum_{\eta \in \mathrm{range}(\mathfrak{M}_\alpha)} P(\mathfrak{M}_\alpha(D_i) = \eta) q_{i,\eta}^{\mathcal{A}^*_\alpha} \\ &= P(\mathfrak{M}_\alpha(D_i) = \eta_i) Q(D_i, \omega_i) + P(\mathfrak{M}_\alpha(D_i) = \eta^*) q_{i,\eta}^{\mathcal{A}^*_\alpha} \\ &= p_i Q(D_i, \omega_i) + (1 - p_i) q_{i,\eta}^{\mathcal{A}^*_\alpha} \end{aligned}$$

Plugging the fact that $p_i = 1$ for $i = n+1,\ldots,N$

$$q_i^{\mathcal{A}^*_\alpha \circ \mathfrak{M}_\alpha} = Q(D_i, \omega_i) \geq \mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\mathrm{identity}}) \qquad \text{for } i = n+1,\ldots,N.$$

If $\mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}_\alpha) < \mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\mathrm{identity}})$, then we also have

$$q_i^{\mathcal{A}^*_\alpha \circ \mathfrak{M}_\alpha} = \mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}_\alpha) \qquad \text{for } i = 1,\ldots,n \tag{5.9}$$

By Lemma 5.6.3, ${\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_\alpha)$ is continuous in α. Hence, $q_i^{\mathcal{A}^*_\alpha \circ \mathfrak{M}_\alpha}$ is continuous in α as long as ${\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_\alpha) < {\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\mathrm{identity}})$. Recall that

$$\begin{aligned} & q_1^{\mathcal{A}^*_\alpha \circ \mathfrak{M}_\alpha} = \alpha Q(D_1, \omega_1) + (1-\alpha) q_{1,\eta^*}^{\mathcal{A}^*_\alpha} \\ \iff \quad & q_{1,\eta^*}^{\mathcal{A}^*_\alpha} = \frac{q_1^{\mathcal{A}^*_\alpha \circ \mathfrak{M}_\alpha} - \alpha Q(D_1, \omega_1)}{1-\alpha} \end{aligned}$$

Now, it is easy to see $q_{1,\eta^*}^{\mathcal{A}^*_\alpha}$ is continuous in α as long as ${\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_\alpha) < {\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\mathrm{identity}})$. By continuity (Lemma 5.6.3 and by the fact that ${\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}$ satisfies Axiom 3.2.1 (sufficiency), there exists α_1 and α_2 such that

$$\alpha_1 > \alpha_2 \tag{5.10}$$

$${\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\mathrm{identity}}) > {\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\alpha_1}) > {\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\alpha_2}) \tag{5.11}$$

Let

$$\mathcal{A}^*_{\alpha_1} = \arg\max_{\mathcal{A}} {\mu_{\mathbb{I}}}^{textSM}(\mathcal{A} \circ \mathfrak{M}_{\alpha_1}) \text{ and} \tag{5.12}$$

$$\mathcal{A}^*_{\alpha_2} = \arg\max_{\mathcal{A}} {\mu_{\mathbb{I}}}^{textSM}(\mathcal{A} \circ \mathfrak{M}_{\alpha_2}). \tag{5.13}$$

We will show that

$$q_{1,\eta^*}^{\mathcal{A}^*_{\alpha_1}} < q_{1,\eta^*}^{\mathcal{A}^*_{\alpha_2}}$$

By Equation 5.9 and Equation 5.11, we have

$${\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\alpha_1}) = q_{1,\eta^*}^{\mathcal{A}^*_{\alpha_1}} > {\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\alpha_2}) = q_{1,\eta^*}^{\mathcal{A}^*_{\alpha_2}}$$

On the contrary, suppose that $q_{1,\eta^*}^{\mathcal{A}^*_{\alpha_1}} \geq q_{1,\eta^*}^{\mathcal{A}^*_{\alpha_2}}$. We have

$$\begin{aligned} q_i^{\mathcal{A}^*_1 \circ \mathfrak{M}_{\alpha_2}} &> {\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\alpha_2}) \qquad \text{for } i = 2, \ldots, N \\ q_1^{\mathcal{A}^*_1 \circ \mathfrak{M}_{\alpha_2}} &\geq {\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\alpha_2}) \qquad \text{for } i = 2, \ldots, N \end{aligned}$$

Hence, by Lemma 5.6.6, there exists $\mathcal{A}_3$ such that ${\mu_{\mathbb{I}}}^{\mathrm{SM}}(\mathcal{A}_3 \circ \mathfrak{M}_{\alpha_2}) > {\mu_{\mathbb{I}}}^{\mathrm{SM}}(\mathcal{A}^*_2 \circ \mathfrak{M}_{\alpha_2})$ which contradicts Equation 5.13. Thus, we must have Equation 5.14 holds. By the same argument, it is easy to see that $q_1^{\mathcal{A}^*_1 \circ \mathfrak{M}_\alpha}$ strictly decreases as α increases for $\alpha_2 \leq \alpha \leq \alpha_1$. Because $q_1^{\mathcal{A}^*_1 \circ \mathfrak{M}_\alpha}$ is continuous in α for $\alpha_2 \leq \alpha \leq \alpha_1$ and $\mathbb{O}^*$ is finite, there must exists some α such that

$$q_1^{\mathcal{A}^*_1 \circ \mathfrak{M}_\alpha} \neq Q(D_1, \omega) \qquad \forall \omega \in \mathbb{O}^* .$$

That is there exists ω_1^* and ω_2^* such that $P(\mathcal{A}^*_\alpha(\eta^*) = \omega_1^*) > 0$, $P(\mathcal{A}^*_\alpha(\eta^*) = \omega_2^*) > 0$ and without loss of generality $Q(D_1, \omega_1^*) > Q(D_1, \omega_2^*)$.

Now, we will show that $Q(D_k, \omega_1^*) < Q(D_k, \omega_2^*)$ for some $k \leq n$ and $k > 1$. On the contrary, suppose that

$$Q(D_i, \omega_1^*) \geq Q(D_i, \omega_2^*) \qquad \text{for } i = 2, \ldots, n$$

Then, replacing ω_2^* with ω_1^* will increase the overall utility (due to the fact that $Q(D_1, \omega_1^*) > Q(D_1, \omega_2^*)$ and by Lemma 5.6.6). Hence, we have $Q(D_k, \omega_1^*) < Q(D_k, \omega_2^*)$ and, without loss of generality, let $k = 2$.

Hence, we have $Q(D_1, \omega_1^*) > Q(D_1, \omega_2^*)$ and $Q(D_2, \omega_1^*) < Q(D_2, \omega_2^*)$. Now, we construct $\mathfrak{M}_{\delta,\beta}$ from $\mathfrak{M}_\alpha$ by setting $p_1 = \alpha - \delta$ and $p_2 = \beta$ and leaving everything else unchanged (i.e. $p_i = 0$ for $i = 3, \ldots, n$ and $p_i = 1$ for $i = n+1, \ldots, N$). Let $\mathcal{A}_{\delta,\beta}^*$ be the best response algorithm of $\mathfrak{M}_{\delta,\beta}$ (i.e. $\mathcal{A}_{\delta,\beta}^* = \arg\max_{\mathcal{A}} {\mu_{\mathbb{I}}}^{\mathrm{SM}}(\mathcal{A} \circ \mathfrak{M}_{\delta,\beta})$).

Denote

$$\vec{q}_{\eta^*}^{\mathcal{A}_{\delta,\beta}^*} = \langle q_{1,\eta^*}^{\mathcal{A}_{\delta,\beta}^*}, ..., q_{n,\eta^*}^{\mathcal{A}_{\delta,\beta}^*} \rangle$$

to be the average quality vector of $D_1, ..., D_n$ achieved by the response algorithm $\mathcal{A}_{\delta,\beta}^*$ upon seeing η^*. As long as ${\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\delta,\beta}) < {\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\text{identity}})$, for $i = 1, \ldots, n$, $q_{i,\eta^*}^{\mathcal{A}_{\delta,\beta}^*}$ is continuous in δ and in β (by the same argument that $q_{1,\eta^*}^{\mathcal{A}_\alpha^*}$ is continuous in α). Hence, $\vec{q}_{\eta^*}^{\mathcal{A}_{\delta,\beta}^*}$ is continuous in δ and in β. Hence, $\vec{q}_{\eta^*}^{\mathcal{A}_{\delta,\beta}^*}$ can be arbitrary close to $\vec{q}_{\eta^*}^{\mathcal{A}_\alpha^*}$ with small enough δ and β. Let $G_{\mathbb{I}^*} = \{g_{i,\omega} = Q(D_i, \omega) | \forall 1 \leq i \leq n, \omega \in \mathbb{O}^*\}$ be a matrix where columns corresponds to $\omega \in \mathbb{O}^*$ and rows correspond to $D_1, ..., D_n$. Let $\mathcal{H}_{\mathbb{I}^*}$ be the convex hull formed by columns of $G_{\mathbb{I}^*}$. Combining with the fact that $\vec{q}_{\eta^*}^{\mathcal{A}_{\delta,\beta}^*}$ is continuous and inside the convex hull $\mathcal{H}_{\mathbb{I}^*}$, with small enough δ and β, $\vec{q}_{\eta^*}^{\mathcal{A}_{\delta,\beta}^*}$ can be expressed as positively affine combination of quality vectors (i.e. $\langle Q(D_1, \omega), ..., Q(D_n, \omega) \rangle$) of ω_1^*, ω_2^* and some other outputs.

Recall that ${\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\delta,\beta})$ is continuous in δ and β. By Lemma 5.6.6, increasing β strictly increases ${\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\delta,\beta})$ if ${\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\delta,\beta}) < {\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\text{identity}})$ (this is due to the fact that, if applying the same best response algorithm, increasing β will increase the overall quality of D_2 and does not change the overall quality of other inputs.) Similarly, increasing δ strictly decreases ${\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\delta,\beta})$ if ${\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\delta,\beta}) < {\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\text{identity}})$. Hence, one can find arbitrary small δ and β such that ${\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\delta,\beta}) = {\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_\alpha)$. Combining with the previous results, there exists positive and small enough δ and β such that

- ${\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\delta,\beta}) = {\mu_{\mathbb{I}}}^{\mathrm{SM}\diamond}(\mathfrak{M}_\alpha)$ and
- $P(\mathcal{A}_{\delta,\beta}^*(\eta^*) = \omega_1^*) > 0$ and $P(\mathcal{A}_{\delta,\beta}^*(\eta^*) = \omega_2^*) > 0$ (i.e. upon seeing η^* generated from $\mathfrak{M}_{\delta,\beta}$, the best response algorithm, $\mathcal{A}_{\delta,\beta}^*$, generates ω_1^* and ω_2^* with positive probabilities).

Now consider $\mathfrak{M}_3 = \mathfrak{M}_\alpha \oplus_{0.5} \mathfrak{M}_{\delta,\beta}$. Denote $\{\eta, \mathfrak{M}\}$ where $\eta \in \mathrm{range}(\mathfrak{M})$ and $\mathfrak{M} \in \{\mathfrak{M}_\alpha, \mathfrak{M}_{\delta,\beta}\}$ be the output of $\mathfrak{M}_3$. $\{\eta, \mathfrak{M}\}$ means that $\mathfrak{M}$ is chosen and η is generated from $\mathfrak{M}$. Let $c = \min(P(\mathcal{A}_\alpha^*(\eta^*) = \omega_1^*)), P(\mathcal{A}_\alpha^*(\eta^*) = \omega_2^*)), P(\mathcal{A}_{\delta,\beta}^*(\eta^*) = \omega_1^*)), P(\mathcal{A}_{\delta,\beta}^*(\eta^*) = \omega_2^*)))$. Let $\mathcal{A}_3$ behave

as $\mathcal{A}^*_\alpha$ if $\mathfrak{M}_\alpha$ is chosen and behave as $\mathcal{A}^*_{\delta,\beta}$ if $\mathfrak{M}_{\delta,\beta}$ is chosen. Simple calculation yields that

$$\begin{aligned}\mu_{\mathbb{I}}{}^{\mathrm{SM}}(\mathcal{A}_3 \circ \mathfrak{M}_3) &= 0.5\,\mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}_\alpha) + 0.5\,\mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\delta,\beta}) \\ &= \mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}_\alpha) = \mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\delta,\beta}).\end{aligned}$$

Let $\mathcal{A}^*_3$ behaves as $\mathcal{A}_3$ except the following conditions.

$$\begin{aligned}P(\mathcal{A}^*_3(\{\eta^*, \mathfrak{M}_\alpha\}) = \omega^*_1) &= P(\mathcal{A}^*_\alpha(\eta^*) = \omega^*_1)) - c \\ P(\mathcal{A}^*_3(\{\eta^*, \mathfrak{M}_\alpha\}) = \omega^*_2) &= P(\mathcal{A}^*_\alpha(\eta^*) = \omega^*_2)) + c \\ P(\mathcal{A}^*_3(\{\eta^*, \mathfrak{M}_{\delta,\beta}\}) = \omega^*_1) &= P(\mathcal{A}^*_{\delta,\beta}(\eta^*) = \omega^*_1)) + c \\ P(\mathcal{A}^*_3(\{\eta^*, \mathfrak{M}_{\delta,\beta}\}) = \omega^*_2) &= P(\mathcal{A}^*_{\delta,\beta}(\eta^*) = \omega^*_2)) - c\end{aligned}$$

Now, we compute the $\mu_{\mathbb{I}}{}^{\mathrm{SM}}(\mathcal{A}^*_3 \circ \mathfrak{M}_3)$. Denote q_i be the overall quality of D_i. For $i = 3, \ldots, n$, q_i can be computed as follows.

$$\begin{aligned}q_i &= \mu_{\mathbb{I}}{}^{\mathrm{SM}}(\mathcal{A}_3 \circ \mathfrak{M}_3) + (\mu_{\mathbb{I}}{}^{\mathrm{SM}}(\mathcal{A}^*_3 \circ \mathfrak{M}_3) - \mu_{\mathbb{I}}{}^{\mathrm{SM}}(\mathcal{A}_3 \circ \mathfrak{M}_3)) \\ &= \mu_{\mathbb{I}}{}^{\mathrm{SM}}(\mathcal{A}_3 \circ \mathfrak{M}_3) \\ &\quad + P(\mathfrak{M}_3(D_i) = \{\eta^*, \mathfrak{M}_\alpha\}) \sum_{\omega \in \{\omega^*_1, \omega^*_2\}} P(\mathcal{A}^*_3(\{\eta^*, \mathfrak{M}_\alpha\}) = \omega) - P(\mathcal{A}_3(\{\eta^*, \mathfrak{M}_\alpha\}) = \omega))Q(D_i, \omega) \\ &\quad + P(\mathfrak{M}_3(D_i) = \{\eta^*, \mathfrak{M}_{\delta,\beta}\}) \sum_{\omega \in \{\omega^*_1, \omega^*_2\}} P(\mathcal{A}^*_3(\{\eta^*, \mathfrak{M}_{\delta,\beta}\}) = \omega) - P(\mathcal{A}_3(\{\eta^*, \mathfrak{M}_{\delta,\beta}\}) = \omega))Q(D_i, \omega) \\ &= \mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}_\alpha) + 0.5[-cQ(D_i, \omega^*_1) + cQ(D_i, \omega^*_2)] + 0.5[cQ(D_i, \omega^*_1) - cQ(D_i, \omega^*_2)] \\ &= \mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}_\alpha)\end{aligned}$$

Now, we compute q_1.

$$\begin{aligned}q_1 &= \mu_{\mathbb{I}}{}^{\mathrm{SM}}(\mathcal{A}_3 \circ \mathfrak{M}_3) \\ &\quad + P(\mathfrak{M}_3(D_1) = \{\eta^*, \mathfrak{M}_\alpha\}) \sum_{\omega \in \{\omega^*_1, \omega^*_2\}} P(\mathcal{A}^*_3(\{\eta^*, \mathfrak{M}_\alpha\}) = \omega) - P(\mathcal{A}_3(\{\eta^*, \mathfrak{M}_\alpha\}) = \omega))Q(D_1, \omega) \\ &\quad + P(\mathfrak{M}_3(D_1) = \{\eta^*, \mathfrak{M}_{\delta,\beta}\}) \sum_{\omega \in \{\omega^*_1, \omega^*_2\}} P(\mathcal{A}^*_3(\{\eta^*, \mathfrak{M}_{\delta,\beta}\}) = \omega) - P(\mathcal{A}_3(\{\eta^*, \mathfrak{M}_{\delta,\beta}\}) = \omega))Q(D_1, \omega) \\ &= \mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}_\alpha) + 0.5(1-\alpha)[-cQ(D_1, \omega^*_1) + cQ(D_1, \omega^*_2)] + 0.5(1-\alpha+\delta)[cQ(D_1, \omega^*_1) - cQ(D_1, \omega^*_2)] \\ &= \mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}_\alpha) + 0.5c\delta[Q(D_1, \omega^*_1) - Q(D_1, \omega^*_2)] \\ &> \mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}_\alpha)\end{aligned}$$

Similarly, q_2 can be computed as follows.

$$\begin{aligned}q_2 &= \mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}_\alpha) + 0.5[cQ(D_2, \omega^*_1) - cQ(D_2, \omega^*_2)] + 0.5(1-\beta)[-cQ(D_2, \omega^*_1) + cQ(D_2, \omega^*_2)] \\ &= \mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}_\alpha) + 0.5c\beta[Q(D_2, \omega^*_2) - Q(D_2, \omega^*_1)]\end{aligned}$$

$$> \quad \mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\alpha})$$

Now, by Lemma 5.6.6, there exists $\mathcal{A}$ such that $\mu_{\mathbb{I}}{}^{\mathrm{SM}}(\mathcal{A} \circ \mathfrak{M}_3) > \mu_{\mathbb{I}}{}^{\mathrm{SM}}(\mathcal{A}_3^* \circ \mathfrak{M}_3)$. Hence, we find the contradiction.

$$\mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\alpha} \oplus_{0.5} \mathfrak{M}_{\delta,\beta}) \geq \mu_{\mathbb{I}}{}^{\mathrm{SM}}(\mathcal{A} \circ (\mathfrak{M}_{\alpha} \oplus_{0.5} \mathfrak{M}_{\delta,\beta})) > \mu_{\mathbb{I}}{}^{\mathrm{SM}}(\mathcal{A}_3^* \circ (\mathfrak{M}_{\alpha} \oplus_{0.5} \mathfrak{M}_{\delta,\beta})) = \max(\mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\alpha}), \mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}(\mathfrak{M}_{\delta,\beta}))$$

Thus, we conclude that $\mu_{\mathbb{I}}{}^{\mathrm{SM}\diamond}$ satisfies Axiom 3.3.2 only if it is a constant function. $\square$

Chapter 6

Sorted Histogram Estimation

In this chapter, we put the theoretical results of Sections 5.3 and 5.4, connecting utility to data analysis via Bayesian decision theory, to the test by applying decision theory to the analysis of sanitized data. The target application is the sorted histogram problem, introduced by Hay et al. [21].

End-users need to process sanitized data to extract query answers and build predictive models while taking advantage of probabilistic knowledge and constraints that are known to hold (e.g., [11, 19, 17, 20, 21, 22, 23, 24, 25, 26]). Since the ability of a sanitizing algorithm to preserve information should (according to the axioms) be measured as the expected error of a Bayesian decision maker, it stands to reason that this Bayesian methodology should play a more prominent role in the analysis of sanitized data. We apply this insight to the sorted histogram problem (also known as the unattributed histogram) first studied by Hay et al. [21]. The goal is to reconstruct a sorted histogram from differentially private sanitized data. The resulting sorted histogram can be used to study edge distributions in social networks, identify power laws, etc. [21]. Using hidden Markov models, we develop a reconstruction algorithm for the sorted histogram problem based on Bayesian decision theory. We show experimentally that our technique outperforms both the least-squares approach of Hay et al. [21] and maximum likelihood estimation.

6.1 The Sorted Histogram Problem

A sorted histogram, also known as the *unattributed histogram* [21], is a histogram in which the attribute names are not important so that the histogram buckets are sorted by their counts and the attribute labels are thrown away (for example, if the histogram counts are $\langle 5, 1, 0, 3\rangle$, the corresponding sorted histogram S is $\langle 0, 1, 3, 5\rangle$). It can be used to study structural properties of discrete distributions such as identifying power laws. For further motivation and a discussion of applications, see [21].

When privacy is a concern, the sorted histogram can be protected with ϵ-differential privacy

by adding independent Laplace($1/\epsilon$) noise (with density $f(x) = \frac{\epsilon}{2}e^{-\epsilon|x|}$) to each cell [21] to get a noisy histogram $\tilde{S}$. The counts in this noisy version no longer appear in sorted order, so the challenge is to derive an estimator $\widehat{S}$ of the underlying sorted histogram S. Thus the counts in $\widehat{S}$ must also be in increasing order.

This is where *usability* plays an important role. The noisy histogram $\tilde{S}$ contains much information but it is not directly useful for a user who wants to study the underlying sorted histogram. To help such a user, we must extract from $\tilde{S}$ a good estimate of the original histogram.

Hay et al. [21] proposed an estimator $\widehat{S}_{\text{LS}}$ whose squared distance from $\tilde{S}$ is smallest (using order-constrained least-squares regression). Accuracy can be improved slightly by changing negative estimated cell counts to 0. Note that the counts in this estimator are generally not integers. Also note that this method does not use knowledge of the noise distribution. Thus we also consider another baseline: a maximum likelihood estimator $\widehat{S}_{\text{ML}}$ (using the pool-adjacent violators algorithm [94, 95, 96]) which is essentially the same as order-constrained L_1 regression [94, 95, 96]. Our competing estimation algorithm is described in Section 6.1.1.

6.1.1 The Estimation Algorithm

Applying Bayesian decision theory to the sorted histogram problem requires choosing a prior distribution, a set of actions, an error function $\mathcal{E}$, and finally an algorithm to select the best action given the noisy histogram $\tilde{S}$. We note that the Bayesian approximation algorithm proposed by Williams and McSherry [17] is *not* applicable. First, [17] requires that the likelihood of the true input be modeled as an independent product distribution; this is *not* true of *sorted* histograms (in fact, we ended up modeling it as a Markov chain). Second, [17] provides an approximation scheme while our algorithm is exact. The estimation procedure is shown in Algorithm 1. In the rest of the section, we explain the algorithm.

6.1.2 Choice of prior (Lines 1, 2, 3)

To ensure that our algorithm does not receive an unfair advantage (e.g., we should not use our knowledge of the distribution of degree sequences in typical social networks), we chose a uniform prior over sorted histograms with n cells. In order to do this, we first need an upper bound m on the maximum value of any cell in the true sorted histogram S. We set $m = 500 + \max_j \tilde{S}(j)$ (where $\tilde{S}$ is the publicly released noisy histogram). The addition of 500 ensures that for all choices of ϵ (parameter of the noise, see Section 6.1) between 0.1 and 1, m is larger than the true maximum of S with probability ≈ 1 (up to 22 decimal places).

The crucial point is to recognize that the uniform prior can be viewed as a non-stationary Markov chain. A non-stationary Markov chain has a notion of time ℓ, a state $S(\ell)$ at time ℓ, an initial distribution $P(S(1) = k)$ at time $\ell = 1$, a transition probability $P(S(\ell+1) = a \mid S(\ell) = b)$, and a requirement called the Markov property: $P(S(\ell + 1) \mid S(\ell)) = P(S(\ell + 1) \mid S(\ell), S(\ell - 1), \ldots, S(1))$.

To see how the uniform prior over sequences can be modeled as a Markov chain, we let time

Algorithm 1: BDThist

Input: n, number of buckets
Input: $\tilde{S}$, noisy histogram
1 m$\leftarrow \tilde{S}[n] + 500$
`// Define HMM initial probabilities,` $k = 0, \ldots, m$
2 **def** $P[S(1) = k] = \binom{m-k+n-1}{n-1} / \binom{m+n}{n}$
`// Define HMM transition probabilities,`
`// for` $\ell = 1, \ldots, n-1; 0 \leq b \leq a \leq m$
3 **def** $P[S(\ell+1) = a \mid S(\ell) = b] = \binom{m-a+n-\ell-1}{n-\ell-1} / \binom{m-b+n-\ell}{n-\ell}$
`// Define the emissions function`
4 **def** $f(\omega \mid b) = \frac{\epsilon}{2} e^{-\epsilon|\omega - b|}$
`// Get the function` $z(\ell, j) \equiv P(\tilde{S}, S[\ell] = j)$
5 $z \leftarrow$forward-backward$(P, f, \tilde{S})$
`// Compute probability of` $\tilde{S}$
6 denom$\leftarrow \sum_{j=0}^{m} z(n, j)$
7 **for** $\ell = 1, \ldots, n$ **do**
8 $\quad \widehat{S}_{HMM}[\ell] = \frac{1}{\text{denom}} \sum_{j=0}^{m} j z(\ell, j)$
9 **return** $\widehat{S}_{HMM}$

correspond to bucket id (which ranges from 1 to n) and we let the state at time ℓ be the count in bucket ℓ, which, by abuse of notation, we also refer to as $S(\ell)$. The initial probability is given by Line 2 of the algorithm (the numerator is the number of sorted histograms with $S(1) = k$ and the denominator is the total number of sorted histograms. The problem of the total number of sorted histogram with n buckets and at most m elements per cell is equivalent to the problem of number of ways to put n sticks in a line of m balls. The analogy is that the number of balls in a group corresponds to the increase in height of a histogram. More specifically, we said that the i-th group of balls is the balls between the i-th stick and the $(i-1)$-th stick with the convention that the 0-th stick is in front of line. Thus, the number of balls in the i-th group corresponds to the difference between the number of elements in i-th bucket and the number of elements in the $(i-1)$-th bucket. Similarly, the number of sorted histograms with $S(1) = k$ equals to number of ways to put $n-1$ sticks in a line of $m-k$ balls.

The transition probabilities are given by Line 3. This is because $P[S(\ell+1) = a | S(\ell) = b]$ equals to (number of sorted histograms with $S(\ell) = b$ and $S(\ell+1) = a$) divided by (number of sorted histograms with $S(\ell) = b$). The denominator can be computed as (number of sorted partial histograms from the first bucket to the ℓ-th bucket with $S(\ell) = b$) $\times$ (number of sorted partial histograms from the ℓ-th bucket to the n-th bucket with $S(\ell) = b$). The numerator can be computed as (number of sorted partial histograms from the first bucket to the ℓ-th bucket with $S(\ell) = b$) $\times$ (number of sorted partial histograms from the ℓ-th bucket to the $(\ell+1)$-the bucket with $S(\ell) = b$ and $S(\ell+1) = a$) $\times$ (number of sorted partial histograms from the $(\ell+1)$-th bucket to the n-th bucket with $S(\ell+1) = a$). Note that (number of sorted partial histograms from the first bucket to the ℓ-th bucket with $S(\ell) = b$) appears both in denominator and numerator and hence cancels

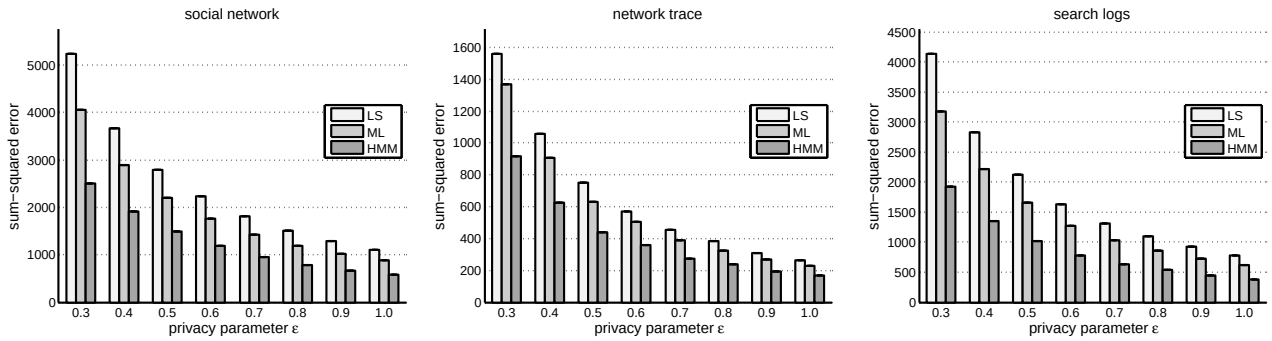

Figure 6.1. Sum-squared error of the estimators $\widehat{S}_{LS}$, $\widehat{S}_{ML}$, $\widehat{S}_{HMM}$ averaged over 100 runs for each ϵ. Average error of $\widehat{S}_{HMM}$ outperformed baselines by at least 26%.

out. Also note that (number of sorted partial histograms from the ℓ-th bucket to the $(\ell+1)$-th bucket with $S(\ell) = b$ and $S(\ell+1) = a$) is 1. Hence, we can simplify $P[S(\ell+1)|a|S(\ell) = b] = \frac{\text{number of sorted partial histograms from the } (\ell+1)\text{-th bucket to the } n\text{-th bucket with } S(\ell+1)=a}{\text{number of sorted partial histograms from the } \ell\text{-th bucket to the } n\text{-th bucket with } S(\ell)=b}$. Reducing numerator and denominator to problems of putting sticks in a line of balls yields the result of Line 3.

Note that we never see the true sorted histogram and thus we never get to observe the states. Instead, we observe noisy values $\tilde{S}(1), \ldots, \tilde{S}(n)$ where the conditional distribution of the noisy count $P(\tilde{S}(i) \mid S(1), \ldots, S(n))$ is equal to $P(\tilde{S}(i) \mid S(i))$ and has the density $f(x) = \frac{\epsilon}{2} e^{-|x-S(i)|}$ since Laplace noise is added to each bucket. This is called an *emission probability* and the overall process with the Markov chain prior and the emission probabilities is called a hidden Markov model [97].

6.1.3 Choice of the set $\mathbb{A}$ of actions

The constrained least-squares regression in [21] produced a sorted histogram whose cell entries do not have to be integers. Likewise we set $\mathbb{A}$, the set of actions, to be the set of all sorted histograms whose cell entries are bounded by m but are not necessarily integers (so an action $a_{\widehat{S}} \in \mathbb{A}$ corresponds to using $\widehat{S}$ as the estimator of the true sorted histogram).

6.1.4 Choice of error function

The constrained least-squares regression in [21] seeks to minimize the sum squared error between the noisy output $\tilde{S}$ and the candidate estimator $\widehat{S}$. We use this sum squared error as our choice of error function $\mathcal{E}$. This is also the error measure we will use to experimentally evaluate the quality of an estimator (out of fairness to the constrained least-squares regression, since that is what it attempts to optimize).

		social network			network trace			search logs		
	Estimator	μ(error)	σ(error)	Wins	μ(error)	σ(error)	Wins	μ(error)	σ(error)	Wins
$\epsilon = 1$	$\widehat{S}_{\text{HMM}}$	**577.2**	49.1	**100**	**166.3**	29.4	**100**	**384.6**	35.0	**100**
	$\widehat{S}_{\text{LS}}$	1,111.2	66.2	0	264.9	33.9	0	779.1	49.5	0
	$\widehat{S}_{\text{ML}}$	889.4	52.0	0	227.3	31.7	0	624.2	43.2	0
$\epsilon = 0.5$	$\widehat{S}_{\text{HMM}}$	**1,494.5**	114.0	**100**	**438.8**	73.2	**100**	**1,024.5**	111.5	**100**
	$\widehat{S}_{\text{LS}}$	2,793.7	163.4	0	751.8	98.8	0	2,121.6	144.5	0
	$\widehat{S}_{\text{ML}}$	2,197.8	147.6	0	630.6	87.6	0	1,664.2	124.9	0
$\epsilon = 0.1$	$\widehat{S}_{\text{HMM}}$	**7,376.5**	1,360.3	**100**	**4,576.0**	1336.4	**99**	**8,277.5**	1,454.6	**100**
	$\widehat{S}_{\text{LS}}$	21,983.1	2,324.9	0	8,297.5	1,750.0	0	17,261.3	2,070.7	0
	$\widehat{S}_{\text{ML}}$	16,430.9	1,720.4	0	6,972.9	1,556.7	1	13,062.2	1,507.6	0

Table 6.1. Results for sorted histograms sanitized with Laplace(ϵ) noise. Each setting was repeated 100 times with new noise. Measurements for the estimators are: average squared error (μ), standard deviation (σ), and the number of times (among the 100 repetitions) it had the lowest error among all the estimators (wins)

6.1.5 Computing the Estimator (Lines 4–8)

To use Bayesian decision theory, we need to take the noisy output $\tilde{S}$ and produce an action (sorted histogram) with least cost, as specified in Equation 5.3 (Section 5.3.1). Since the error function $\mathcal{E}$ is just sum-squared error, it is well-known [97] that the optimal action (sorted histogram) is the expectation of the posterior distribution:

$$E[S \mid \tilde{S}] \quad = \quad \sum_{S \in \mathbb{I}} S \times P(\text{data} = S \mid \mathfrak{M}(\text{data}) = \tilde{S}) \tag{6.1}$$

Note that here $\mathbb{I}$ is the set of sorted histograms, and $S \in \mathbb{I}$ is treated mathematically as a vector of dimension n. Equation 6.1 is indeed a sorted histogram as it is the weighted average of sorted histograms.

Lines 4–8 of Algorithm 1 are designed to compute this value as follows. Since S is modeled as a non-stationary Markov chain, then the noisy version $\tilde{S}$ becomes a *hidden Markov model* (HMM) with the same state transitions as S and with emissions probabilities specified by the Laplace($1/\epsilon$) distribution. The forward-backward [97] algorithm for HMMs will produce the value of $P(\tilde{S}, S[\ell] = j)$ for all ℓ and j simultaneously; it is stored as the function $z(\ell, j)$. This is done in Line 5. Then Line 6 computes $P(\tilde{S})$. Now note that $z(\ell, j)/P(\tilde{S}) = P(S[\ell] = j \mid \tilde{S})$. Using this fact, the loop on Lines 7 and 8 computes the expected value $E[S[\ell] \mid \tilde{S}]$ so that the return value $\widehat{S}_{HMM}$ is the desired expectation from Equation 6.1.

6.2 Experiments

We now compare our estimator (based on Bayesian decision theory) for the sorted histogram problem against the baselines of least-squares estimation [21] and maximum likelihood estimation. These experiments can also be viewed as a partial validation of the axiomatic approach to utility,

since it implied that we should be using Bayesian decision theory to work with sanitized data.

6.2.1 The Setup

6.2.1.1 Datasets

For comparison with prior work [21], we evaluate the estimators for the sorted histogram using the three datasets considered by Hay et al. [21]. These are the **social network** dataset (consisting of $11,342$ histogram cells), the **net trace** dataset (consisting of $65,536$ histogram cells), and the **search logs** dataset (consisting of $32,768$ histogram cells).

6.2.1.2 The Baselines

Let $\widehat{S}_{HMM}$ denote the estimator for the sorted histogram problem that is based on Bayesian decision theory (as described in Section 6.1.1). We compare our estimator against a maximum likelihood estimator $\widehat{S}_{ML}$. Because Laplace noise was added to the original sorted histogram S to get the noisy histogram $\tilde{S}$, a maximum likelihood estimator is simply the sorted histogram that is closest to $\tilde{S}$ in the L_1 distance. It can be found using the pool-adjacent violators (PAV) algorithm [94, 95, 96]. We also compare against $\widehat{S}_{LS}$, the original solution of Hay et al. which returned the sorted histogram that is closest to $\tilde{S}$ in terms of sum-squared error (i.e. L_2 distance).

6.2.1.3 Implementation

We ran all of our experiments on computing clusters that used a variety configurations of the compute nodes, ranging from 4-6 core, 2.66-3.06 GHz, and 48-96GB RAM. The code for $\widehat{S}_{LS}$ (from [21]) was written in Python, as was our implementation of the $\widehat{S}_{ML}$, the maximum likelihood estimator. Our new estimator $\widehat{S}_{HMM}$, based on Bayesian decision theory, was written in C – this is because it is significantly slower (the apparent cost for improved accuracy) and we needed to run thousands of experimental runs. For this reason we will later analyze both time complexity and wall-clock running time.

6.2.2 Results

We now compare the three estimators $\widehat{S}_{HMM}$, $\widehat{S}_{ML}$, and $\widehat{S}_{LS}$ on the three datasets **social network**, **network trace**, and **search logs** considered by [21].

6.2.2.1 Accuracy

The baseline estimator $\widehat{S}_{LS}$ (from [21]) was designed to minimize sum-squared error and thus our experiments compare the sum-squared error between an estimator and the true sorted histogram S.

Our first results are shown in Figure 6.1. For each combination of dataset and ϵ value, we generated 100 noisy histograms $\tilde{S}_1, \ldots, \tilde{S}_{100}$ by adding Laplace$(1/\epsilon)$ noise to each. For each of

the hundred $\tilde{S}_i$, we computed the estimators $\widehat{S}_{LS}, \widehat{S}_{ML}, \widehat{S}_{HMM}$ and then computed the sum-squared error between the estimators and the true original sorted histogram. The values shown in Figure 6.1 are averaged over those 100 repetitions. It is clear that $\tilde{S}_{HMM}$ outperforms the other estimators. For each dataset and ϵ value, the average error of $\widehat{S}_{HMM}$ was at least 26% less than the error of the others. We consider statistical significance next.

6.2.2.2 Statistical Significance and Consistency

For each dataset and ϵ value, we generated 100 noisy histograms $\tilde{S}_1, \ldots, \tilde{S}_{100}$. For each $\tilde{S}_i$, we determined which of the 3 estimators had lowest error. We tabulated the results in Table 6.1. As we can see, $\widehat{S}_{HMM}$ was consistently the best estimator, outperforming the others 100 out of 100 times for almost all experimental settings (the only exception was the network trace dataset with $\epsilon = 0.1$ for which it was the best 99 out of 100 times). This is highly statistically significant with p-value less than 10^{-29}.

6.2.2.3 Time Complexity and Running times

The time complexity of constrained least-squares regression for producing $\widehat{S}_{\mathrm{LS}}$ is $O(n)$ [21], where n is the number of cells in the sorted histogram. Maximum likelihood inference for producing $\widehat{S}_{\mathrm{ML}}$ used our $O(n \log n)$ implementation of the pool-adjacent violators (PAV) algorithm [94, 95, 96]. The forward-backward algorithm for producing $\widehat{S}_{\mathrm{HMM}}$ is $O(nm^2)$ where m is the upper bound on the maximum value of any cell. Thus we see that improvements in accuracy come at the cost of running time. Experimentally, we saw that the running time for computing $\widehat{S}_{\mathrm{HMM}}$ was ≈ 20 hours for **network trace**, ≈ 2.5 hours for **search logs**, and ≈ 4.5 hours for **social network**. On the other hand, computing $\widehat{S}_{\mathrm{LS}}$ and $\widehat{S}_{\mathrm{ML}}$ often took less than a second (except for $\widehat{S}_{\mathrm{ML}}$ on **network trace**, which took < 5 seconds on average) .

6.2.2.4 Intangibles

Our experimental results showed that the improved estimation accuracy came at the cost of a higher running time. In general, for sophisticated end-users there are other considerations. Extracting utility from sanitized data is nontrivial: the constrained least-squares regression estimator $\widehat{S}_{LS}$ relies on a clever algorithm [21] as does the PAV algorithm [94, 95, 96] for the maximum likelihood estimator $\widehat{S}_{ML}$. Our HMM-based estimator relies on the nontrivial forward-backward dynamic programming algorithm with scaling factors for numerical stability [97]. Overall this points to exciting research opportunities in the statistical analysis of sanitized data.

6.3 Conclusions

We develop a new estimation algorithm for the privacy-preserving sorted histogram problem [21]. It is based on the idea that Bayesian decision theory should play a role in data processing since information preservation should be measured as the expected error of a Bayesian decision

maker. A thorough experimental analysis shows that our algorithm consistently and significantly outperforms competing approaches. This serves as a partial empirical validation of the theoretical axiomatic approach.

Chapter 7

Conclusions and Future Directions

We present a first principle approach toward the theory of privacy and utility. We build a framework to extract semantic guarantees of privacy definitions and study utility measures axiomatically. The contributions are as follows.

- We present the first general framework to extract semantics guarantees of privacy definitions in Bayesian sense. The data owner (publisher) can use the framework to analyze whether the privacy definition is too weak or too strong.
- The framework enables the data owner (publisher) to relax the privacy definition, if it is too strong, by manipulating the rowcone. We use randomized response as an example and show that a direct relaxation results in differential privacy.
- We examine generalizations of many existing approaches for measuring utility and identify conditions under which they satisfy necessary utility axioms.
- We show the only class of utility measures satisfying utility axioms is expected error of Bayesian decision maker.
- We present a natural procedure, called sufficing, to fix violation of Axiom 3.2.1 (sufficiency). Applying sufficing to expected discrepancy and worst-case discrepancy shows additional connections to Bayesian decision theory.
- Our decision-theoretic estimation of sorted histogram outperforms stat-of-the-art approaches. This promotes to use decision-theoretic approaches on private data analysis.

This thesis builds a fundamental theory of privacy and utility and opens up several avenues for future works.

- Consistent normal form and rowcone can only extract the semantics guarantees that always holds. One future direction is to extend this framework to extract the semantic guarantee that fails to protect privacy with probabilities, for example probabilistic relaxations of differential privacy.

- The sanitized output generated by a high information preservation algorithms does not necessary mean it is easy to work with. One future direction is to study the tradeoff between information preservation and usability.
- Selecting the algorithm having the highest utility among all algorithms satisfying the same privacy definition is often not an easy task. One future direction is to build the framework that allows the data owner to select the privacy algorithm with the highest utility.
- Accuracy comes with the cost of computations. Decision-theoretic approaches provide a more accurate answer but may not be easily computed. One future direction is to work on efficient and possibly approximate algorithms based on Bayesian decision theory for more complicated problems.

Bibliography

[1] Arrington, M. (2006), "AOL Proudly Releases Massive Amounts of Private Data," TechCrunch: http://www.techcrunch.com/2006/08/06/aol-proudly-releases-massive-amounts-of-user-search-data/.

[2] Barbaro, M. and T. Zeller (2006) "A face is exposed for AOL Searcher No. 4417749," *New York Times*.

[3] Singel, R. (2010), "NetFlix Cancels Recommendation Contest After Privacy Lawsuit," Wired: http://www.wired.com/threatlevel/2010/03/netflix-cancels-contest/.

[4] Netflix (2006), "The Netflix Prize Rules: http://www.netflixprize.com/rules," .

[5] Narayanan, A. and V. Shmatikov (2006), "How To Break Anonymity of the Netflix Prize Dataset," .
URL `http://arxiv.org/abs/cs/0610105`

[6] Warner, S. L. (1965) "Randomized Response: A survey technique for eliminating evasive answer bias," *Journal of the American Statistical Association*.

[7] Agrawal, S. and J. R. Haritsa (2005) "A Framework for High-Accuracy Privacy-Preserving Mining," in *ICDE*.

[8] Gouweleeuw, J., P. Kooiman, L. Willenborg, and P.-P. de Wolf (1998) "Post Randomisation for Statistical Disclosure Control: Theory and Implementation," *Journal of Official Statistics*, **14**(4).

[9] McSherry, F. and K. Talwar (2007) "Mechanism Design via Differential Privacy," in *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science*, pp. 94–103.

[10] Li, C., M. Hay, V. Rastogi, G. Miklau, and A. McGregor (2010) "Optimizing Linear Counting Queries Under Differential Privacy," in *PODS*.

[11] Ghosh, A., T. Roughgarden, and M. Sundararajan (2009) "Universally Utility-Maximizing Privacy Mechanisms," in *STOC*, pp. 351–360.

[12] Alvim, M. S., M. E. Andrés, K. Chatzikokolakis, and C. Palamidessi (2011) "On the Relation between Differential Privacy and Quantitative Information Flow," in *ICALP*.

[13] Alvim, M. S., M. E. Andrés, K. Chatzikokolakis, P. Degano, and C. Palamidessi (2011), "Differential Privacy: on the trade-off between Utility and Information Leakage," `http://arxiv.org/abs/1103.5188`.

[14] Askari, M., R. Safavi-Naini, and K. Barker (2012) "An information theoretic privacy and utility measure for data sanitization mechanisms," in *CODASPY*.

[15] Li, C., D. Y. Li, G. Miklau, and D. Suciu (2013) "A Theory of Pricing Private Data," in *ICDT*.

[16] Ghosh, A. and A. Roth (2011) "Selling Privacy at Auction," in *EC*.

[17] Williams, O. and F. McSherry (2010) "Probabilistic Inference and Differential Privacy," in *NIPS*.

[18] Kifer, D. and B.-R. Lin, "An Axiomatic View of Statistical Privacy and Utility," To appear in Journal of Privacy and Confidentiality.

[19] Gupte, M. and M. Sundararajan (2010) "Universally Optimal Privacy Mechanisms for Minimax Agents," in *PODS*.

[20] Barak, B., K. Chaudhuri, C. Dwork, S. Kale, F. McSherry, and K. Talwar (2007) "Privacy, Accuracy and Consistency Too: A Holistic Solution to Contingency Table Release," in *Proceedings of the ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS)*.

[21] Hay, M., V. Rastogi, G. Miklau, and D. Suciu (2010) "Boosting the Accuracy of Differentially-Private Histograms Through Consistency," in *VLDB*.

[22] Little, R. J. (1993) "Statistical Analysis of Masked Data," *Journal of Official Statistics*, **9**(2), pp. 407–426.

[23] Raghunathan, T., J. Reiter, and D. Rubin (2003) "Multiple imputation for statistical disclosure limitation," *Journal of Official Statistics*, **19**, pp. 1–16.

[24] Zhang, Q., N. Koudas, D. Srivastava, and T. Yu (2007) "Aggregate Query Answering on Anonymized Tables," in *Proceedings of the IEEE International Conference on Data Engineering (ICDE)*.

[25] Proserpio, D., S. Goldberg, and F. McSherry (2012) "A Workflow for Differentially-Private Graph Synthesis," in *SIGCOMM Workshop on Online Social Networks*.

[26] Aggarwal, C. C. (2008) "On Unifying Privacy and Uncertain Data Models," in *Proceedings of the 24th International Conference on Data Engineering (ICDE)*, pp. 386–395.

[27] Dwork, C. (2006) "Differential privacy," in *ICALP*.

[28] Samarati, P. (2001) "Protecting Respondents' Identities in Microdata Release," *TKDE*, **13**(6).

[29] Chen, B.-C., D. Kifer, K. LeFevre, and A. Machanavajjhala (2009) "Privacy-Preserving Data Publishing," *Foundations and Trends in Databases*, **2**(1-2), pp. 1–167.

[30] Winkler, W. E. (2004) "Re-identification Methods for Masked Microdata," in *Privacy in Statistical Databases*, Springer.

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

[32] Duncan, G. T. and D. Lambert (1989) "The risk of disclosure for microdata," *Journal of Business and Economic Statistics*, **7**(2).

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

[34] ——— (2000) *Elements of Statistical Disclosure Control*, Springer.

[35] Dwork, C. and M. Naor (2010) "On the Difficulties of Disclosure Prevention in Statistical Databases or The Case for Differential Privacy," *JPC*, **2**(1).

[36] Lambert, D. (1993) "Measures of Disclosure Risk and Harm," *Journal of Official Statistics*, **9**(2), pp. 313–331.

[37] Dalenius, T. (1986) "Finding a Needle in a Haystack, or Identifying Anonymous Census Records," *Journal of Official Statistics*, **2**(3), pp. 329–336.

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

[39] Machanavajjhala, A., J. Gehrke, D. Kifer, and M. Venkitasubramaniam (2006) "ℓ-Diversity: Privacy Beyond k-Anonymity," in *ICDE*.

[40] Dinur, I. and K. Nissim (2003) "Revealing information while preserving privacy," in *PODS*.

[41] Dwork, C., F. McSherry, and K. Talwar (2007) "The Price of Privacy and the Limits of LP Decoding," in *STOC*.

[42] Dwork, C., M. Naor, O. Reingold, G. N.Rothblum, and S. Vadhan (2009) "On the Complexity of Differentially Private Data Release: Efficient Algorithms and Hardness Results," in *STOC*, pp. 381–390.

[43] Kargupta, H., S. Datta, Q. Wang, and K. Sivakumar (2003) "On the Privacy Preserving Properties of Random Data Perturbation Techniques," in *ICDM*.

[44] Huang, Z., W. Du, and B. Chen (2004) "Deriving Private Information from Randomized Data," in *SIGMOD*.

[45] Liu, K., C. Giannella, and H. Kargupta (2008) *A survey of attack techniques on privacy-preserving data perturbation methods*, chap. 15, Springer, pp. 357–380.

[46] Ganta, S. R., S. P. Kasiviswanathan, and A. Smith (2008) "Composition Attacks and Auxiliary Information in Data Privacy," in *KDD*.

[47] Wong, R., A. Fu, K. Wang, and J. Pei (2007) "Minimality Attack in Privacy Preserving Data Publishing," in *VLDB*.

[48] Kifer, D. (2009) "Attacks on Privacy and de Finetti's Theorem," in *SIGMOD*.

[49] Backstrom, L., C. Dwork, and J. Kleinberg (2007) "Wherefore art thou r3579x?: anonymized social networks, hidden patterns, and structural steganography," in *WWW*.

[50] Clifton, C., M. Kantarcioglu, and J. Vaidya (2002) "Defining Privacy For Data Mining," in *Proc. of the National Science Foundation Workshop on Next Generation Data Mining*.

[51] Wu, X., X. Ying, K. Liu, and L. Chen (2010) "A Survey of Privacy-Preservation of Graphs and Social Networks," in *Managing and Mining Graph Data*, chap. 14, Springer US, pp. 421–453.

[52] Li, N., T. Li, and S. Venkatasubramanian (2007) "t-Closeness: Privacy Beyond k-Anonymity and l-Diversity," in *ICDE*.

[53] Fung, B. C. M., K. Wang, R. Chen, and P. S. Yu (2010) "Privacy-Preserving Data Publishing: A Survey on Recent Developments," *ACM Computing Surveys*, **42**(4).

[54] Evfimievski, A., J. Gehrke, and R. Srikant (2003) "Limiting Privacy Breaches in Privacy-Preserving Data Mining," in *PODS*.

[55] Dwork, C., F. McSherry, K. Nissim, and A. Smith (2006) "Calibrating Noise to Sensitivity in Private Data Analysis." in *TCC*.

[56] Lin, B.-R. and D. Kifer (2013) "Information Preservation in Statistical Privacy and Bayesian Estimation of Unattributed Histograms," in *SIGMOD*.

[57] Kifer, D. and B.-R. Lin (2010) "Towards an axiomatization of statistical privacy and utility," in *PODS*.

[58] Paris, J. B. (1994) *The Uncertain Reasoner's Companion*, Cambridge University Press.

[59] von Neumann, J. and O. Morgenstern (2007) *Theory of Games and Economic Behavior*, Princeton University Press.

[60] Savage, L. J. (1972) *The Foundations of Statistics*, Dover.

[61] Myerson, R. B. (1986) *Axiomatic Foundations of Bayesian Decision Theory*, *Tech. Rep. 671*, Northwestern University Center for Mathematical Studies in Economics and Management Science. URL `http://www.kellogg.northwestern.edu/research/math/papers/671.pdf`

[62] Reiter, J. (2005) "Using CART to generate partially synthetic public use microdata," *Journal of Official Statistics*, pp. 441–462.

[63] Abowd, J. M. and S. D. Woodcock (2001) "Disclosure Limitation in Longitudinal Linked Data," *Confidentiality, Disclosure, and Data Access: Theory and Practical Applications for Statistical Agencies*, pp. 215–277.

[64] Iyengar, V. S. (2002) "Transforming data to satisfy privacy constraints," in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

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

[66] Blum, A., K. Ligett, and A. Roth (2008) "A Learning Theory Approach to Non-interactive Database Privacy," in *STOC*, pp. 609–618.

[67] Ding, B., M. Winslett, J. Han, and Z. Li (2011) "Differentially private data cubes: optimizing noise sources and consistency," in *SIGMOD*.

[68] Evfimievski, A., R. Srikant, R. Agrawal, and J. Gehrke (2002) "Privacy preserving mining of association rules," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

[69] Rastogi, V., D. Suciu, and S. Hong (2007) "The boundary between privacy and utility in data publishing," in *Proceedings of the 33rd international conference on Very large data bases (VLDB)*, pp. 531–542.

[70] Xiao, X. and Y. Tao (2006) "Anatomy: Simple and Effective Privacy Preservation," in *Proceedings of the 32nd International Conference on Very Large Databases (VLDB)*.

[71] Inan, A., M. Kantarcioglu, and E. Bertino (2009) "Using Anonymized Data for Classification," in *ICDE*.

[72] Fang, C. and E.-C. Chang (2008) "Information Leakage in Optimal Anonymized and Diversified Data," in *Information Hiding*.

[73] Narayanan, A. and V. Shmatikov (2009) "De-Anonymizing Social Networks," in *IEEE Symposium on Security and Privacy*.

[74] McClure, D. and J. P. Reiter (2012) "Differential Privacy and Statistical Disclosure Risk Measures: An Investigation with Binary Synthetic Data," *Trans. Data Privacy*, **5**(3), pp. 535–552.

[75] Dwork, C. and S. Yekhanin (2008) "New Efficient Attacks on Statistical Disclosure Control Mechanisms," in *CRYPTO*.

[76] Kasiviswanathan, S. P., M. Rudelson, A. Smith, and J. Ullman (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*.

[77] Choromanski, K. and T. Malkin (2012) "The Power of the Dinur-Nissim Algorithm: breaking Privacy of Statistical and Graph Databases," in *PODS*.

[78] Kifer, D. and A. Machanavajjhala (2011) "No Free Lunch in Data Privacy," in *SIGMOD*.

[79] ———, "A Rigorous and Customizable Framework for Privacy," Unpublished manuscript, available upon request.

[80] Boreale, M. and M. Paolini (2012) "Worst- and Average-Case Privacy Breaches in Randomization Mechanisms," in *TCS*.

[81] Kasiviswanathan, S. P. and A. Smith (2008), "A Note on Differential Privacy: Defining Resistance to Arbitrary Side Information," http://arxiv.org/abs/0803.3946.

[82] Rastogi, V., M. Hay, G. Miklau, and D. Suciu (2009) "Relationship Privacy: Output Perturbation for Queries with Joins," in *PODS*, pp. 107–116.

[83] Bhaskar, R., A. Bhowmick, V. Goyal, S. Laxman, and A. Thakurta (2011) "Noiseless database privacy," in *ASIACRYPT*.

[84] Gehrke, J., E. Lui, and R. Pass (2011) "Towards Privacy for Social Networks: A Zero-Knowledge Based Definition of Privacy," in *TCC*.

[85] Miklau, G. and D. Suciu (2004) "A Formal Analysis of Information Disclosure in Data Exchange," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*.

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

[87] Xiao, X., Y. Tao, and N. Koudas (2010) "Transparent Anonymization: Thwarting Adversaries Who Know the Algorithm," *TODS*, **35**(2).

[88] Cormode, G., D. Srivastava, N. Li, and T. Li (2010) "Minimizing minimality and maximizing utility: analyzing method-based attacks on anonymized data," *VLDB*.

[89] Zhang, L., S. Jajodia, and A. Brodsky (2007) "Information disclosure under realistic assumptions: privacy versus optimality," in *CCS*.

[90] Kasiviswanathan, S. P., H. K. Lee, K. Nissim, S. Raskhodnikova, and A. Smith (2008) "What Can We Learn Privately?" in *FOCS*.

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

[92] Burns, F., M. Fiedler, and E. Haynsworth (1974) "Polyhedral Cones and Positive Operators," in *Linear Algebra and Its Applications*, pp. 547–559.

[93] Gelman, A., J. B. Carlin, H. S. Stern, and D. B. Rubin (2003) *Bayesian Data Analysis*, 2nd ed., Chapman & Hall/CRC.

[94] Barlow, R. E., D. J. Bartholomew, J. M. Bremner, and H. D. Brunk (1972) *Statistical Inference Under Order Restrictions*, John Wiley and Sons.

[95] Robertson, T. and P. Waltman (1968) "On Estimating Monotone Parameters," *Ann. Math. Statist.*, **39**(3), pp. 1030–1039.

[96] Robertson, T. and F. T. Wright (1980) "Algorithms in Order Restricted Statistical Inference and the Cauchy Mean Value Property," *Ann. Statist.*, **8**(3), pp. 645–651.

[97] Bishop, C. M. (2006) *Pattern Recognition and Machine Learning*, Springer.

Vita

Bing-Rong Lin

EDUCATION

Sep. 2006- May 2014 **The Pennsylvania State University, USA (PSU)**
Ph.D., **Computer Science and Engineering**, **GPA: 3.77**
• Dissertation: A First Principle Approach Toward Data Privacy and Utility

Sep. 2002- Jun. 2004 **National Chiao Tung University, Taiwan (NCTU)**
Master of Science, **Computer Science and Information Engineering**
• Master Thesis: Dynamic Channel Allocation Policies for IEEE 802.11 Access Points

Sep. 1998- Jun. 2002 **National Chiao Tung University, Taiwan**
Bachelor of Science, **Computer Science and Information Engineering**

RESEARCH/WORKING EXPERIENCE

Aug. 2007- Present **Research Assistant, Pennsylvania State University**

Aug. 2012- May 2012 **Summer Intern, Mitsubishi Electric Research Laboratories**

Aug. 2008- May 2008 **Summer Intern, Telcordia Technologies**

Aug. 2005- May 2006 **Research Staff (Full-time), Computer and Communication Research Center, National Tsing Hua University (NTHU)**

Sep. 2002- Dec. 2004 **Research Assistant, High Speed Communication and Computing Lab., NCTU**

TEACHING EXPERIENCE

Aug. 2006- May 2007 **Teaching Assistant, CSE Department, PSU**
Database Management and *Programming Language*

Sep. 2002- Jan. 2004 **Teaching Assistant, CSIE Department, NCTU**
Wireless Network, *System Programming*, and *Operation Systems*.

HONORS

2006 **Harvey & Geraldine Brush Graduate Fellowship**, PSU

2004 **Excellent Paper Award**, "A Mechanism for Quick Bluetooth Device Discovery", *The tenth Mobile Computing Workshop*, Taichung, Taiwan, Mar. 2004

2004 **Institute for Information Industry (III) Scholarship**

2003 **NCTU Academic Achievement Award**, awarded to top students

2001 **Annual Research Project Competition Award**, "Ad Hoc Network Implementation", CSIE Department, NCTU

2000 **National Science Council Scholarship**