PRESERVING DIFFERENTIAL PRIVACY IN COMPLEX DATA ANALYSIS

by

Yue Wang

A dissertation submitted to the faculty of
The University of North Carolina at Charlotte
in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in
Computing and Information Systems

Charlotte

2015

Approved by:

_____
Dr. Xintao Wu

_____
Dr. Yong Ge

_____
Dr. Zbigniew W. Ras

_____
Dr. Shan Yan

_____
Dr. Yuliang Zheng

ABSTRACT

YUE WANG. Preserving differential privacy in complex data analysis. (Under the direction of DR. XINTAO WU)

Omnipresent databases from various resources, such as social networks, electronic commercial websites, and health related wearable devices, have provided researchers with unprecedented opportunities to analyze complex social phenomena. While society would like to encourage such scientific endeavors, we are faced with the problem of providing researchers with a fairly precise picture of the quantities or trends of complex data without disclosing sensitive information about individuals. In this dissertation, we investigate how to apply differential privacy model in complex data analysis. Differential privacy is a paradigm of post-processing the output of queries or mining tasks on databases such that the inclusion or exclusion of a single individual from a database makes no statistical difference to the results found. It provides formal privacy guarantees that do not depend on an adversary's background knowledge. There has been extensive research on how to enforce differential privacy in analyzing tabular data and several mechanisms have been developed to achieve differential privacy protection. However, there are significant challenges to achieve differential privacy protection on complex data including social networks and biological sequence data mainly due to high sensitivity of desired statistics and the complexity of mining tasks. In this dissertation, we focus on how to enable accurate analysis of complex data while preserving differential privacy. We firstly propose a general divide and conquer framework to deal with complex computation tasks by decomposing a complex target computation into several less complex unit computations connected by basic mathe-

matical operations (e.g., addition, subtraction, multiplication, division), and perturbing the output of each unit with Laplace noise derived from its own sensitivity value and the distributed privacy threshold. Next, we develop solutions to more complicated applications: differential privacy graph generation and differential privacy preserving spectral analysis of network topology. We examine the state-of-the-art differential privacy preserving mechanisms including the exponential mechanism and the smooth sensitivity and develop feasible solutions to these problems. Additionally, we consider the potential information disclosure from differential privacy preserving outputs. We propose two attacking models to show how genome-wide association studies (GWAS) results can be used to infer the trait or the identity of individuals even if those results are under differential privacy protection. We also provide the countermeasure for model inversion attacks where the released regression model under the differential privacy protection can still be exploited by the attacker to derive information about sensitive attributes used in the model. We develop a novel approach for releasing differential private regression models by leveraging the functional mechanism to perturb coefficients of the polynomial representation of the objective function while balancing the privacy budgets for sensitive and non-sensitive attributes in learning the regression models. Our approach can effectively retain the models' utility while preventing model inversion attacks. Finally, we consider the problem of enforcing differential privacy at the client-side against an untrusted server in the data collection scenario. Our proposed technique which uses the randomized response technique incurs less utility loss than the traditional output perturbation mechanism especially when the sensitivity of desired computation is high, and also provides the individuals a simple manner to protect their sensitive information by themselves against anyone with ulterior motives.

ACKNOWLEDGMENTS

In pursuing this PhD. degree, I have received much help, suggestions and encouragement from many people. I hereby express my sincere gratitude to all of them.

First of all, I would like to thank for my dissertation committee members, Dr. Xintao Wu, Dr. Yong Ge, Dr. Zbigniew W. Ras, Dr. Shan Yan and Dr. Yuliang Zheng. I appreciate their time and effort on my research and dissertation, as well as their advice on my future life and career. Especially, I would like to thank my advisor, Dr. Xintao Wu. He is my role model as an excellent researcher who always has clear thinking and rigorous logic. He guided me on the road of research. His great support helped me build up self-confidence. He also gave me many useful suggestions to improve my thinking, writing and presenting. I appreciate his carefulness and patience very much. I also would like to thank for co-authors for their help in developing research ideas, presenting results and publishing papers, especially Dr. Xinghua Shi, Dr. Xiaowei Ying, Dr. Leting Wu, Dr. Zhilin Luo, Jun Zhu from University of North Carolina at Charlotte and Cheng Si from University of Arkansas.

It is a great pleasure to work in Data Privacy Lab. Thank for my labmates Yuemeng Li and Dr. Kai Pan who are great colleagues to work with. I would also like to thank for my friends, Xiongwei Xie, Jinyue Xia, Boshu Ru, Jinglin Li, Rongcheng Lin, Bingxuan Li, Yazhe Zhu, Dan Jia and many others for their support, especially in those hard times.

Finally, I would like to thank for my parents for their love. This dissertation is dedicated to them.

This work was supported in part by U.S. National Science Foundation IIS-0546027,CNS-0831204, CCF-1047621 and National Institute of Health 1R01GM103309.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

Big data is used to describe the data growth challenges and opportunities in three dimensions: volume, velocity and variety. Specific technology and analytical methods are now being developed to transform the information assets with high volume, velocity and variety into value. The collection and manipulation of such large and complex datasets can bring enormous real-world benefits in various fields, such as healthcare, scientific research, agriculture, logistics, urban design, energy, retailing, crime reduction, and business operations. Typical examples of the applications based on complex data analysis include: Google Flu Trends that uses aggregated Google search data to estimate flu activity in United States; Microsoft Band that can monitor health related physiological indicators and give insights about happiness and well-being; and Facebook's interest based advertising that targets marketers' ads based on the not only what people do on Facebook, but also the websites Facebook users visit and the apps they use when they are not on the social network.

However, privacy and intellectual property laws have not kept up with the pace of technological change. No one can really answer the question: who has right to the data. It is often advertised by social network or social media platforms that the users own right to their own content. But the ambiguous or even opposite claims on how that content is used or shared with third parties are often listed somewhere in the long installation agreement which users would directly ignore. Consequently, as people post, they actually give up the control over how the post content, as long as the location, device and other information, is

used in future.

To protect individuals' privacy as well as promoting complex data analysis, we are faced with the problem of providing researchers with a fairly precise picture of the quantities or trends of complex data without disclosing sensitive information about participants. Differential privacy [29, 32] was recently proposed to provide formal privacy guarantees that can maximize the query accuracy from statistical databases while minimizing the identifying risk of a participant's record. In this dissertation, we investigate how to apply differential privacy model in the applications based on complex data analysis. Particularly, the term 'complex' indicates that we will mostly focus on the challenges due to the high variety of data.

1.1 Differential Privacy

The privacy preserving data mining community has expended great effort in developing sanitization techniques to effectively anonymize data so that the sanitized data can be published or shared with others. Researchers have proposed various privacy models such as k-anonymity [96], l-diversity [80], and t-closeness [76] and developed various sanitization approaches including suppression, generalization, randomization, permutation, and synthetic data generation. The aim is that an honest analyst should be able to perform a variety of ad hoc analysis and derive accurate results whereas a malicious attacker should be unable to exploit the published data to infer private information about individuals.

All these sanitization approaches adopt the idea of pre-processing the raw data such that each individual's record or her sensitive attribute values are hidden within a group of other individuals. However there is no guarantee to achieve strict privacy protection since they could not completely prevent adversaries from exploiting various auxiliary information

(e.g., via background knowledge attacks [27, 82] and composition attacks [40]) to breach privacy. A well-known instance of the failure of sanitization approaches is the 'Linkage Attack' on the Netflix database. Netflix offered $\$1,000,000$ prize for a 10% improvement in its recommendation system. In order to protect customer privacy, Netflix had removed all personal information as well as the real customer ids when it released a training dataset. But such strategy failed to protect privacy since attackers could partially de-anonymize the Netflix training database, compromising the identity of users, by linking it with another movie rating database, IMDB, using the date of rating by a user.

Differential privacy [29, 32] was then proposed to provide formal privacy guarantees that do not depend on an adversary's background knowledge (including access to other databases) or computational power. Differential privacy is a paradigm of post-processing the output of queries such that the inclusion or exclusion of a single individual from the data set makes no statistical difference to the mining results found.

Differential privacy is usually achieved by directly adding calibrated noise on the output of a specific computation f. The calibrating process includes the calculation of the global sensitivity of the computation f that bounds the possible change in the computation output over any two neighboring databases. The added noise is then generated from a Laplace distribution with the scale parameter determined by the global sensitivity of f and the user-specified privacy threshold ϵ.

Most previous research on differential privacy theoretically studied the problem of enforcing differential privacy in relational databases [8, 12, 13, 17, 30, 32, 33, 39, 40, 42, 53, 65, 97, 123]. The applicability of enforcing differential privacy in real world applications has also been studied, e.g., the application of differential privacy to collaborative recom-

mendation system [85], logistic regression [17], publishing contingency tables [8, 124] or data cubes [25], privacy preserving integrated queries [84]. However, there are significant challenges to achieve differential privacy protection on complex data including social networks and biological sequence data mainly due to high sensitivity of desired statistics and the complexity of mining tasks.

In this dissertation, we focus on how to enable accurate analysis of complex data while preserving differential privacy. We first propose several techniques to apply differential privacy model into the process of mining social network topology (in Chapters 2,3 and 4). We then study how and to what extend attackers can learn sensitive information from data analysis results which are under the protection of differential privacy and propose countermeasures to the possible attacks (in Chapters 5 and 6). At last, we propose a solution to the problem of enforcing differential privacy at the client-side against an untrusted server in the data collection scenario. Our proposed approach enables individuals to protect their sensitive information by themselves against anyone with ulterior motives (in Chapter 7).

1.2 Mining Social Network Topology

Social networks are of significant importance in various application domains. The management and analysis of these networks have attracted interests in various research communities such as sociology, statistics, and data mining. Studies are mostly focused on revealing interesting properties of networks and discovering efficient and effective analysis approaches [6, 10, 43, 63, 66, 67, 69, 78, 79, 98, 100, 103, 116, 118, 121, 125, 128, 129].

Social networks often contain some private attribute information about individuals as well as their sensitive relationships. In particular, privacy disclosure risks arise when the data owner wants to publish or share the social network data with another party for research

or business-related applications. The privacy concerns associated with data analysis over social networks have incurred the recent research. A detailed survey of the recent work on this topic can be found in [122].

In this dissertation, we study the problem of how to protect privacy in the process of analyzing social network topology under the framework of differential privacy. The general noise injection technique to achieve differential privacy works well for traditional aggregate functions (e.g., count and sum) over tabular data. In social network data analysis, various graph features such as cluster coefficient and modularity often have much higher sensitivity value (proportional to the number of nodes) than the traditional aggregate functions on tabular data do. In addition, for graph topology itself, direct noise injection according to global sensitivity may break the constraints on the data (e.g., the adjacency matrix should have binary entries) or destroy the underlying characters of the data. Furthermore, for some computations such as spectral decomposition, the classic noise injection approach cannot be applied since there is no explicit formula to calculate global sensitivity. We propose techniques to meet such challenges in this dissertation.

1.2.1 General Graph Feature Calculation

A divide and conquer approach (denoted as *D&C* in this dissertation) has been suggested in the literature [32] for computing complicated mining functions. The basic procedure of this approach is to first decompose the target computation f into several less complex unit computations $f_1, \cdots, f_m$ connected by basic mathematical operations (e.g., addition, subtraction, multiplication, division), then perturb the output of each f_i with Laplace noise derived from its own sensitivity value and the distributed privacy threshold ϵ_i, and finally combine those perturbed outputs as the perturbed output of computation f.

However, this straightforward adaptation could lead to poor performance especially when multiplication or division operations are involved. And there is no theoretical study on calculating the unbiased estimate of f from perturbed results of f_is. In Chapter 2, we theoretically examine how various operations affect the accuracy of complex computations. When unit computations have large global sensitivity values, we enforce the differential privacy by calibrating noise based on the smooth sensitivity [90], rather than the global sensitivity. By doing this, we achieve the strict differential privacy guarantee with smaller magnitude noise. We illustrate our approach by learning clustering coefficient (a popular graph feature used in social network analysis) from private networks. Empirical evaluations on five real social networks and various synthetic graphs generated from three random graph models show the developed divide and conquer approach outperforms the direct approach.

1.2.2 Graph Generation

Graph topologies play an irreplaceable role in the network analysis. Attempts [87, 95] have been made in enforcing differential privacy in graph topology generation and publishing. The idea is to enforce differential privacy on graph model parameters learned from the original network and then generate the graphs for releasing using the graph model with the private parameters. The released graphs then can be used for various analyses. The authors in [87] tried to generate differentially private graph topology with the stochastic Kronecker graph generation model [73]. However, the stochastic Kronecker graph generation model often cannot accurately capture graph properties of real social networks due to its simplicity in the generation process. The authors in [95] developed a private dK-graph model. The dK-graph model [81], which constructs graphs to satisfy a family of properties based

on various types of node degree correlations, has been shown an effective graph generation model. However, the private 2K-graph model proposed in [95] was based on the local sensitivity of degree correlations due to the large global sensitivity. As a result, the model could not achieve rigorous differential privacy protection.

In Chapter 3, we propose a private 2K-graph generation model that achieves rigorous differential privacy. Our idea is to enforce the differential privacy by calibrating noise based on the smooth sensitivity [90]. By doing this, we achieve the strict differential privacy guarantee with smaller magnitude noise. We conduct experiments on four real networks and compare the performance of our private dK-graph models with the stochastic Kronecker graph generation model in terms of utility and privacy tradeoff. Empirical evaluations show the effectiveness of our proposed private dK-graph models.

1.2.3 Spectral Analysis

We then focus on differential privacy preserving spectral graph analysis. Spectral graph analysis deals with the analysis of the spectra (eigenvalues and eigenvector components) of the graph's adjacency matrix or its variants.

In Chapter 4, we develop two approaches to computing the ϵ-differential private spectra, the first k eigenvalues and the corresponding eigenvectors, from the input graph G. The first approach, denoted as *LNPP*, is based on the Laplace Mechanism [32] that calibrates Laplace noise on the eigenvalues and every entry of the eigenvectors based on their sensitivities. We derive the global sensitivities of both eigenvalues and eigenvectors based on the matrix perturbation theory [101]. Because the output eigenvectors after perturbation are no longer orthogonormal, we postprocess the output eigenvectors by using the state-of-the-art vector orthogonalization technique [41]. The second approach, denoted as *SBMF*, is based

on the exponential mechanism [86] and the properties of the matrix Bingham-von Mises-Fisher density for network data spectral analysis [55]. We prove that the Gibbs sampling procedure [55] achieves differential privacy. We conduct empirical evaluation on a real social network data and compare the two approaches in terms of utility preservation (the accuracy of spectra and the accuracy of low rank approximation) under the same differential privacy threshold. Our empirical evaluation results show that *LNPP* generally incurs smaller utility loss.

1.3 Potential Information Disclosure from Private Outputs

After exploring techniques of protecting the mining results from complex data especially social network topology with differential privacy, we are then exploring whether those differentially private results are secure enough to prevent the attackers from learning sensitive information accurately. In this section, we introduce two pieces of work about possible attacks on the private mining outputs and the countermeasures for those attacks.

1.3.1 Trait and Identity Inference using Results from GWAS

Except for graph data from social network, another type of commonly studied sensitive data is the human genome sequence data. Intensive attention has been attracted to Genome-wide association studies (GWAS) due to the rapid decrease of genotyping costs and promising potential in genetic diagnostics. GWAS focus on associations between single-nucleotide polymorphisms (SNPs) and human traits like common diseases.

The biomedical community generally used to agree that statistics (e.g., allele frequency) learned from SNP data(not like the SNP sequence itself) are safe to directly release to public since they cannot be used to identify individuals. However, the findings in [56] showed that GWAS statistics do not completely conceal identity since it is straightforward

to assess the probability a person participated in a GWA study.

In Chapter 5, we study a related but different privacy protection problem, i.e., whether and to what extent GWAS statistics can be exploited by an attacker to learn private information of regular people (rather than those GWAS participants). Specifically, we formulate two potential attacks: 1) *trait inference attack* that aims to infer the probability of a target developing some private trait when the target's SNP profile is available to the attacker; and 2) *identity inference attack* that aims to infer the probability of a record in an anonymized genebank database belonging to the target when some traits of the target are available.

Both attacks pose a serious threat to individuals when their SNP profiles are exploited by attackers. Our results show that unexpected privacy breaches can occur because aggregation statistics (or even perturbed statistics under differential privacy) provide no explicit security guarantees and these statistics could be exploited by attackers to identify individuals or derive private information.

1.3.2 Regression Model Release against Model Inversion Attack

Similar to the GWAS statistics learned from DNA sequence data, regression models may also be exploited by an adversary to breach privacy of both participant individuals in the datasets and regular non-participant individuals, even if the released model is under the protection of differential privacy.

Differential privacy preserving regression models guarantee protection against attempts to infer whether a subject was included in the training set used to derive a model. However, a recently proposed model inversion attack [38] indicates that such models can't provide effective protection to the sensitive attribute value of a target individual. In model inversion attack, given the released regression model, an adversary with some background knowledge

about the target individual can make accurate prediction on some sensitive attribute value, as long as such attribute is used as input in the regression model. Model inversion attack may fail when the privacy threshold ϵ is very small; but in that way, the regression model also loses acceptable accuracy.

In Chapter 6, we develop a novel approach which leverages the functional mechanism to perturb coefficients of the polynomial representation of the objective function but effectively balances the privacy budget for sensitive and non-sensitive attributes in learning the differential privacy preserving regression models. Theoretical analysis and empirical evaluations demonstrate our approach can effectively prevent model inversion attacks and retain the model utility.

1.4 Private Data Collection

In previous sections, we introduce several approaches to preserve individuals' privacy in the data publishing scenario, as in which a trusted server releases datasets of individual information or answers queries on such datasets. In this section, we introduce our work on how to enforce differential privacy in the data collection scenario, as in which an untrusted server collects personal information from individuals. Our work is based on the randomized response strategy, a client-based privacy solution that does not rely upon a trusted third-party server. Such strategy puts control over data back to clients.

Given a client's value, the client runs the randomized algorithm to get a corresponding perturbed value and reports it to the untrusted server. The use of randomized response in surveys enables easy estimations of accurate population statistics while preserving the privacy of the individual respondents. In Chapter 7, we study the privacy guarantee provided by randomized response under the differential privacy model. We also compare the per-

formance of randomized response with that of the standard Laplace mechanism which is based on query-output independent adding of Laplace noise.

One advantage of the use of the randomized response in the data collection is that the collected data can be released for as much analysis as needed without worrying further privacy disclosure. This is different from the output perturbation where each additional analysis consumes further privacy budget. Moreover, the use of the randomized response for collecting data incurs less utility loss than the output perturbation when the sensitivity of functions is high, as demonstrated in our experiment where we calculate the number of triangles in the social network while preserving differential privacy.

1.5 Paper Organization

The dissertation is organized as follows:

In Chapter 2, we introduce the divide and conquer approach of ensuring differential privacy protection in accurate data analysis of data collected from social network. Using cluster coefficient as an example, we illustrate our divide and conquer approach framework for decomposing complex target computations and calibrating noise based on either smooth sensitivity [90] or global sensitivity, finally combining private unit computation results to get the private analysis result for the original target computation.

In Chapter 3, we propose a differentially private dK-graph generation model that enforces rigorous differential privacy while preserving utility. We conduct theoretical analysis and empirical evaluations to show that the developed private dK-graph generation models significantly outperform the approach based on the stochastic Kronecker generation model.

In Chapter 4, we present two approaches, *LNPP* and *SBMF*, to enforce differential privacy in spectral graph analysis. We apply and evaluate the Laplace Mechanism [32] and

the exponential mechanism [86] on the differential privacy preserving eigen decomposition on the graph topology.

In Chapter 5, we study whether and to what extent GWAS statistics (even under the protection of differential privacy) can be exploited by an attacker to learn private information of general population, not limited to GWAS participants. We propose two potential attacks, *trait inference attack* and *identity inference attack*. Both attacks exploit the released GWAS statistics about the associations between SNP genotypes and human traits.

In Chapter 6,we develop a novel approach to learn regression model under differential privacy. Our proposed approach is based on the functional mechanism which perturbs coefficients of the polynomial representation of the objective function. By effectively balancing the privacy budget for sensitive and non-sensitive attributes, we can effectively prevent model inversion attacks and retain the model utility.

In Chapter 7, we study how to enforce differential privacy by using the randomized response in the process of collecting sensitive data. Our research starts from the simple case with one single binary attribute and extends to the general case with multiple polychotomous attributes. We provide the explicit formula of the mean squared error of various derived statistics based on the randomized response theory and prove the randomized response outperforms the classic Laplace mechanism.

In Chapter 8, we offer our concluding remarks and discuss future work.

CHAPTER 2: COMPUTING GRAPH TOPOLOGY STATISTICS UNDER DIFFERENTIAL PRIVACY

2.1 Introduction

As introduced in the previous chapter, differential privacy [29, 32] provides formal privacy guarantees that do not depend on an adversary's background knowledge (including access to other databases) or computational power. And it is usually achieved by directly adding calibrated laplace noise on the output of the computation f.

In social network analysis, various graph features such as cluster coefficient and modularity often have a high sensitivity (proportional to the number of nodes), which is different from traditional aggregate functions on tabular data. Calibrating noise according to such high global sensitivity will leads to poor mining performance, since the results found is mostly tortured by noise. And for some mining functions, we may not have explicit formula to calculate global sensitivity.

In this chapter, we propose a divide and conquer approach to deal with graph feature calculation under differential privacy protection. We theoretically examine how various operations affect the accuracy of complex computations. We achieve the strict differential privacy guarantee with smaller magnitude noise leveraging noise injection with smooth sensitivity [90] rather than global sensitivity when unit computations have large global sensitivity values. We illustrate our approach by learning clustering coefficient from private networks topologies.

2.2 Background

We first revisit the formal definition differential privacy and the classic mechanism of enforcing differential privacy by calibrating Laplace noise based on global sensitivity in Section 2.2.1. We then introduce the smooth sensitivity framework [90] when global sensitivity yields unacceptable high noise levels in Section 2.2.2. The smooth sensitivity framework can calibrate the instance-specific noise with smaller magnitude than the worst-case noise based on the global sensitivity. In Section 2.2.3, we introduce complex graph models by which we generate various synthetic graphs for our empirical evaluation in Section 2.5.

2.2.1 Differential Privacy

In prior work on differential privacy, a database is treated as a collection of *rows*, with each row corresponding to the data of a different individual. Here we focus on how to compute graph features from private network topology described as its adjacency matrix. We aim to ensure that the inclusion or exclusion of a link between two individuals from the graph make no statistical difference to the results found.

Definition 1. (Differential Privacy [29]) A graph analyzing algorithm Ψ that takes as input a graph G, and outputs $\Psi(G)$, preserves (ε, δ)-differential edge privacy if for all closed subsets S of the output space, and all pairs of neighboring graphs G and G' from $\Gamma(G)$,

$$Pr[\Psi(G) \in S] \leq e^{\epsilon} \cdot Pr[\Psi(G') \in S] + \delta, \tag{1}$$

where

$$\Gamma(G) = \{G'(V, E') | \exists!(u, v) \in G \; but \; (u, v) \notin G'\}. \tag{2}$$

A differentially private algorithm provides an assurance that the probability of a partic-

ular output is almost the same whether or not any individual edge is included. The privacy parameter pair (ϵ, δ) controls the amount by which the distributions induced by two neighboring graphs may differ (smaller values enforce a stronger privacy guarantee).

A general method for computing an approximation to any function f while preserving ϵ-differential privacy is given in [32]. The mechanism for achieving differential privacy computes the sum of the true answer and random noise generated from a Laplace distribution. The magnitude of the noise distribution is determined by the sensitivity of the computation and the privacy parameter specified by the data owner. The sensitivity of a computation bounds the possible change in the computation output over any two neighboring graphs (differing at most one link).

Definition 2. (Global Sensitivity [32]) The global sensitivity of a function $f : D \rightarrow \mathbb{R}^d$ ($G \in D$),in the analysis of a graph G, is

$$GS_f(G) := \max_{G, G' s.t. G' \in \Gamma(G)} ||f(G) - f(G')||_1 \tag{3}$$

Theorem 1. (The Mechanism of Adding Laplace noise [32]) An algorithm A takes as input a graph G, and some $\epsilon > 0$, a query Q with computing function $f : D^n \rightarrow \mathbb{R}^d$, and outputs

$$A(G) = f(G) + (Y_1, ..., Y_d) \tag{4}$$

where the Y_i are drawn i.i.d from $Lap(GS_f(G)/\epsilon)$. The Algorithm satisfies $(\epsilon, 0)$-differential privacy.

Differential privacy maintains composability, i.e., differential privacy guarantees can be provided even when multiple differentially-private releases are available to an adversary.

Theorem 2. (Composition Theorem [31]) If we have n numbers of (ϵ, δ)-differentially pri-

vate mechanisms $M_1, \cdots, M_n$, computed using graph G, then any composition of these mechanisms that yields a new mechanism M is $(n\epsilon, n\delta)$-differentially private.

Differential privacy can extend to group privacy as well: changing a group of k edges in the data set induces a change of at most a multiplicative $e^{k\epsilon}$ in the corresponding output distribution. In this dissertation, we mainly focus on the edge privacy. We can extend the algorithm to achieve the node privacy by using the above composition theorem [31].

2.2.2 Smooth Sensitivity

It may be hard to derive the global sensitivity of a complex function or global sensitivity yields unacceptable high noise levels. Nissim et al. [90] introduces a framework that calibrates the instance-specific noise with smaller magnitude than the worst-case noise based on the global sensitivity.

Definition 3. (Local Sensitivity [32, 90]) The local sensitivity of a function $f : D \to \mathbb{R}^d$, ($G \in D$) is

$$LS_f(G) := \max_{G's.t.G' \in \Gamma(G)} ||f(G) - f(G')||_1. \tag{5}$$

Under the definition of *local sensitivity*, we only consider the set of G' for a given and predetermined G, such that the inclusion or exclusion of a single link between individuals cannot change the output distribution appreciably. We would emphasize that the release $f(G)$ with noise proportional to $LS_f(G)$ cannot achieve rigorous differential privacy as the noise magnitude might reveal information about the database. Refer to Example 1 in [90] for an illustrative example. To satisfy the strict differential privacy, Nissim et al. [90] proposes the β-smooth sensitivity and shows that adding noise proportional to a smooth upper bound on the local sensitivity yields a private output perturbation mechanism.

Definition 4. (Smooth Sensitivity [90]) For $\beta > 0$, the β-smooth sensitivity of $f : D \to \mathbb{R}^d$ ($G \in D$),in the analysis of a given graph G, is

$$S^*_{f,\beta}(G) = \max_{G' \in D} \left(LS_f(G') \cdot e^{-\beta d(G,G')} \right) \tag{6}$$

where $d(G, G')$ is the distance between graphs G and G'.

Nissim et al. [90] introduces how to compute smooth sensitivity based on the local sensitivity at distance s (measuring how much the sensitivity can change when up to s entries of G are modified).

Definition 5. (Computing Smooth Sensitivity)The sensitivity of f at distance s is

$$LS^{(s)}_f(G) = \max_{G' \in D: d(G,G') \leq s} LS_f(G') \tag{7}$$

The β-smooth sensitivity can be expressed in terms of $LS^{(s)}_f(G)$ as

$$\begin{aligned} S^*_{f,\beta}(G) &= \max_{s=0,1,\cdots,n} e^{-s\beta} \left(\max_{G': d(G,G')=s} LS_f(G') \right) \\ &= \max_{s=0,1,\cdots,n} e^{-s\beta} LS^{(s)}_f(G) \end{aligned} \tag{8}$$

Theorem 3 shows the mechanism of calibrating noise to the smooth bound to achieve (ϵ, δ)-differential privacy.

Theorem 3. (Mechanism to Add Noise Based on Smooth Sensitivity [90]) For a function $f : D \to \mathbb{R}^d (G \in D)$, the following mechanism achieves (ϵ, δ)-differential privacy ($\epsilon > 0, \delta \in (0, 1)$):

$$A(G) = f(G) + \frac{S^*_{f,\beta}(G)}{\alpha} \cdot (Z_1, \cdots, Z_d) \tag{9}$$

where $\alpha = \epsilon/2$, $\beta = \frac{\epsilon}{4(d+ln(2/\delta))}$, and Z_i $(i = 1, \cdots, d)$ is drawn i.i.d from $Lap(0, 1)$.

Specifically when d=1, β can be reduced as $\beta = \frac{\epsilon}{2ln(2/\delta)}$.

2.2.3 Graph Generation Models

Several network models have been proposed for studying the topological properties of real networks. Among them, the Erdös and Rényi random graphs, the Watts and Strogatz small world model, and the Barabási-Albert scale-free networks have been widely used [23]. In our work, we use the above three models with various parameters to generate synthetic graphs for empirical evaluation.

The Erdös and Rényi Random Graph [34] is the most basic model of complex networks which defines a graph with n vertices and a probability p of connecting each pair of vertices. In this model, the average degree of each node is $p(n-1)$ and the degree distribution is a Poisson distribution. The global cluster coefficient of the graph equals p. We refer to this model as the ER model in this chapter.

The Small-World Model of Watts and Strogatz [115] is the most popular model of random networks with the small-world property, i.e., most vertices can be reached from others through a small number of edges. The Watts and Strogatz model can also generate graphs with the presence of a large number of triangles. In contrast, ER networks have the small world property but a small average clustering coefficient. We can construct a small-world network by staring with a regular lattice of n nodes in which each node is connected to k nearest neighbors in each direction. Next each edge is randomly rewired with probability p. When p near zero, the generated graph tends to have a high number of triangles but large distances; when p gets close to 1, the generated graph becomes a random graph with short distances but few triangles. The degree distribution for small-world networks is similar to

that of random networks, with average degree $2k$. The cluster coefficient of the graph is correlated to $\frac{3(k-1)}{2(2k-1)}(1-p)^3$ [23]. We refer to this model as the WS model in this chapter.

The Scale-free Networks of Barabási and Albert [7] was proposed after the WS model in order to capture the characteristics of some networks whose degree distributions follow a power law. In scale-free networks, some vertices are highly connected while others have few connections. The Barabási and Albert model generates a graph by starting with a set of m_0 nodes, afterwards, at each step of the construction the network grows with the addition of new nodes. For each new node, m_1 new edges are added between the new node and some previous nodes. The nodes which receive the new edges are picked following a linear preferential attachment rule so that the most connected nodes have greater probability to receive new nodes as neighbors. The degree distribution of graphs generated by this model is $P(d) \sim d^{-3}$. The average degree is $2m_1$. The cluster coefficient of the graph is correlated to $n^{-0.75}$. We refer to this model as the BA model in this chapter.

2.3 A Divide and Conquer Algorithm

Our divide and conquer approach is to express a function f in terms of unit computations $f_1, \cdots, f_m$ such that f can be calculated from results of f_i ($i = 1, \cdots, m$) via basic mathematical operations $\odot$. In this chapter, we limit $\odot$ as linear combination, multiplication, and division. In our future work, we will extend our study to other mathematical operations such as root square and logarithm which are often used in data mining algorithms. For each unit computation f_i, we introduce noise in order to maintain its differential privacy requirement (ϵ_i, δ_i). Specifically, we can run the randomization mechanism with noise distribution on each f_i to achieve (ϵ_i, δ_i)-differential privacy. Using Theorem 2, we can achieve (ϵ, δ)-differential privacy of f where $\epsilon = \sum_{i=1}^{m} \epsilon_i$ and $\delta = \sum_{i=1}^{m} \delta_i$.

Input: Graph G, a target graph statistic function f, privacy parameters (ϵ, δ)
Output: $\tilde{f}(G)$ satisfies (ϵ, δ)-differential privacy

1: *Decompose* f into unit computations $f_1, \cdots, f_m$ connected by basic mathematical operations $\odot$
2: *Distribute* (ϵ_i, δ_i) for each f_i such that $\epsilon = \sum \epsilon_i$ and $\delta = \sum \delta_i$
3: FOR $i = 1 \rightarrow m$
4: IF $GS_{f_i}(G)$ is uncomputable or too large for perturbation
5: Derive the formula of $LS^{(s)}_{f_i}(G)$
6: Using the (ϵ_i, δ_i) to compute (β, α) //Theorem 3
7: Compute the β-smooth sensitivity $S^*_{f_i,\beta}(G)$ using $\beta, LS^{(s)}_{f_i}(G)$ //Equation 8
8: Compute $\tilde{f}_i(G)$ using $\alpha, S^*_{f_i,\beta}(G)$ //Equation 9
9: ELSE
10: Compute $\tilde{f}_i(G)$ using $GS_{f_i}(G)$ // Theorem 1
11: ENDIF
12: ENDFOR
13: *Integrate* $\tilde{f}_1(G), \cdots, \tilde{f_m}(G)$ into the (ϵ, δ)-differential private estimator of f: $\tilde{f}(G)$.

Algorithm 1: Differentially private graph statistics learning: $D\&C$ Approach

Algorithm 1 illustrates our divide and conquer approach. In Line 2, the total privacy budget (ϵ, δ) is distributed among unit computations such that each f_i has a privacy threshold (ϵ_i, δ_i). It is a challenging problem to determine the optimal distribution of privacy budget such that the combined output $\tilde{f}$ achieves the optimal approximation of f. In this chapter, we simply distribute privacy budget equally among all unit computations, i.e., $\epsilon_i = \frac{1}{m}\epsilon$ and $\delta_i = \frac{1}{m}\delta$. In our evaluation, we show that the accuracy of the output $\tilde{f}$ varies significantly when we have different privacy budget distributions among f_i. In Lines 3-12, we enforce (ϵ_i, δ_i)-differential privacy on each f_i. For f_i that has small global sensitivity $GS_{f_i}(G)$, we apply Theorem 1 directly (Line 10). For f_i that may still have large global sensitivity or may not have an explicit formula for deriving global sensitivity, we first calculate its local sensitivity at distance s (Line 5), derive the smooth sensitivity parameter (β, α) based on the (ϵ_i, δ_i) (Line 6), compute its β-smooth sensitivity (Line 7), and finally enforce (ϵ_i, δ_i)-differential privacy on f_i by following Theorem 3 to calibrate noise based on the derived

smooth sensitivity (Line 8). In Line 13, we output (ϵ, δ)-differential private $\tilde{f}$ by integrating $\tilde{f}_i$ $(i = 1, \cdots, m)$.

Next we show that our divide and conquer approach achieves unbiased estimate of f when operations $\odot$ contain linear combination, multiplication, and division. For simplicity, we choose one pair of functions, f_i and f_j. We assume the true value of f_i (f_j) on a given data set is a (b) and f_i (f_j) is perturbed by a Laplace noise $Lap(0, a')$ $(Lap(0, b'))$. In other words, $\tilde{f}_i = a + Lap(0, a')$ and $\tilde{f}_j = b + Lap(0, b')$. Lemma 1 shows the linear combination of Laplace noise perturbed results ($\tilde{f}_i$ and $\tilde{f}_j$) is an unbiased estimate for the linear combination of the original variables (f_i and f_j). This lemma covers the mathematical operations of addition and subtraction. Similarly, Lemma 2 and Lemma 3 shows the result for multiplication and division, respectively.

Lemma 1. The linear combination of two perturbed values with Laplace noise is an unbiased estimate for the linear combination of the two original values without the perturbations.

$$E(u \cdot (a + Lap(0, a')) + v \cdot (b + Lap(0, b'))) = E(u \cdot a + v \cdot b) \tag{10}$$

Assuming that $a, b, a', b' \in \mathbb{R}$; and $u, v \in \mathbb{R}$ are parameters of the linear combination.

Proof.

$$E(u \cdot (a + Lap(0, a')) + v \cdot (b + Lap(0, b')))$$

$$= E(u \cdot a + v \cdot b) + u \cdot E(Lap(0, a')) + v \cdot E(Lap(0, b'))$$

Since $E(Lap(0, a')) = 0$ and $E(Lap(0, b')) = 0$, we have

$$E(u \cdot (a + Lap(0, a')) + v \cdot (b + Lap(0, b'))) = E(u \cdot a + v \cdot b)$$

□

Lemma 2. The product of two perturbed values with independent Laplace noise is an unbiased estimate for the product of the two original values without the perturbations.

$$E((a + Lap(0, a')) \cdot (b + Lap(0, b'))) = E(a \cdot b) \tag{11}$$

Assuming that $a, b, a', b' \in \mathbb{R}$ and a, b are independently perturbed.

Proof.

$$\begin{aligned}
&E((a + Lap(0, a')) \cdot (b + Lap(0, b'))) \\
&= E(a \cdot b + b \cdot Lap(0, a') + a \cdot Lap(0, b') + Lap(0, a') \cdot Lap(0, b')) \\
&= E(a \cdot b) + b \cdot E(Lap(0, a')) + a \cdot E(Lap(0, b')) + E(Lap(0, a') \cdot Lap(0, b'))
\end{aligned}$$

$E(Lap(0, a')) = E(Lap(0, b')) = 0$; besides, $E(Lap(0, a') \cdot Lap(0, b')) = E(Lap(0, a')) \cdot E(Lap(0, b'))$ since a, b are independently perturbed with Laplace noise; hence

$$E((a + Lap(0, a')) \cdot (b + Lap(0, b'))) = E(a \cdot b)$$

□

Lemma 3. The quotient of two perturbed results with Laplace noise is an unbiased estimate for the quotient of the two original values without the perturbation.

$$E(\frac{a + Lap(0, a')}{b + Lap(0, b')}) = E(\frac{a}{b}) \tag{12}$$

Assuming that $a, b, a', b' \in \mathbb{R}$ and $b \neq 0$.

Proof.

$$E(\frac{a+Lap(0,a')}{b+Lap(0,b')}) = E(\frac{a}{b}) + E(\frac{a+Lap(0,a')}{b+Lap(0,b')} - \frac{a}{b})$$

Since $E(Lap(0,a')) = 0$ and $E(0, Lap(b')) = 0$, we have

$$\begin{aligned}
&E(\frac{a+Lap(0,a')}{b+Lap(0,b')} - \frac{a}{b}) \\
&= E(\frac{(a+Lap(0,a')b - a(b+Lap(0,b')))}{b\cdot(b+Lap(0,b'))}) \\
&= E(\frac{b\cdot Lap(0,a')}{b\cdot(b+Lap(0,b'))}) - E(\frac{a\cdot Lap(0,b')}{b\cdot(b+Lap(0,b'))}) = 0
\end{aligned}$$

□

2.4 Learning Vertex Clustering Coefficient

In this section, we illustrate our D&C-based differential privacy preserving approach by learning vertex clustering coefficient, a widely used graph metric in social network analysis. Specifically, we show how to derive the local sensitivity at distance s (a key step in Algorithm 1 Line 5) for vertex clustering coefficient. We would emphasize here that our approach works naturally with other graph metrics such as graph modularity and data mining tasks (where functions can be decomposed by unit computations connected by basic mathematical operations).

The vertex clustering coefficient of vertex *i* in a graph quantifies how close i's neighbors are to being a clique (complete graph). This measure was first introduced by Watts and Strogatz in 1998 [114] to determine whether a graph is a small-world graph.

$$C_i = \frac{N_\Delta(i)}{N_3(i)} = \frac{N_\Delta(i)}{d_i(d_i-1)/2} \tag{13}$$

Table 1: Notations of graph metrics used in Chapter 2

The original graph with n nodes and m edges	$G(n, m)$
Adjacent matrix of G	A
An entry in A	a_{ij}
The degree of a node i	d_i
The maximum node degree in G	d_{max}
The clustering coefficient of a node i	C_i
The set of the neighboring nodes of node i	$Ngb(i)$
Number of triangles involving node i	$N_\Delta(i)$
Number of connected triples (i as the central node)	$N_3(i)$
The vector of all C_i	C
The vector of all d_i	D
The vector of all $N_\Delta(i)$	N_Δ
The vector of all $N_3(i)$	N_3

where $N_\Delta(i)$ is the number of triangles involving vertex *i*, $N_3(i)$ is the number of connected triples having *i* as the central vertex, and d_i is the degree of vertex *i*.

We can see that C_i can be naturally expressed as a quotient of two unit computations $N_\Delta(i)$ and $N_3(i)$ or a quotient of $N_\Delta(i)$ and $d_i(d_i - 1)/2$. In social network analysis, data miners often query for the vector C=$(C_1, \cdots, C_n)'$, which contains the clustering coefficients of all the vertices. For example, the average vertex clustering coefficient among all the vertices, which is defined as $\widetilde{C} = \frac{1}{n}\sum_i C_i$, is a widely used metric for graph analysis. We can see that C can also be expressed by two vectors, N_Δ=$(N_\Delta(1), \cdots, N_\Delta(n))'$ and N_3=$(N_3(1), \cdots, N_3(n))'$. Similarly, N_3 could be further decomposed to D=$(d_1, \cdots, d_n)'$. Table 1 shows the notations used in this chapter.

Table 2 shows the global sensitivity and local sensitivity for the vertex clustering coefficient C_i (as well as its decomposed unit computations $N_\Delta(i)$, $N_3(i)$, d_i) and all vertices's clustering coefficients C (as well as its decomposed unit computations N_Δ, N_3, D). We skip the proof details in this chapter since most of them are either well known or can be easily derived. We would point out that degree sequence D has a low global sensitivity

Table 2: Global sensitivity and local sensitivity of graph metrics

Function f	GS_f	LS_f
C_i	1	$2/d_i$
$N_\Delta(i)$	$n-2$	$\max_{a_{ij}=1} \lvert Ngb(i) \cap Ngb(j) \rvert$
$N_3(i)$	$n-2$	$d_i - 1$
d_i	1	1
C	$n-1$	d_{max}
N_Δ	$3(n-2)$	$3\max_{a_{ij}=1} \lvert Ngb(i) \cap Ngb(j) \rvert$
N_3	$2n-4$	$2d_{max}-2$
D	2	2

while other functions such as N_Δ have very high global sensitivity value.

To apply our $D\&C$ algorithm, we need to derive the formulas of the local sensitivity at distance s for all above computations. Result 1 shows the formula of the local sensitivity at distance s for the vertex clustering coefficient C_i and Result 2 shows the formulas of the local sensitivity at distance s for $N_\Delta(i)$ and $N_3(i)$. We omit the proof of Result 2 since it is straightforward.

Result 1. The local sensitivity at distance s for the vertex clustering coefficient C_i is

$$LS^{(s)}_{C_i}(G) = \begin{cases} \frac{2}{d_i - s} & \text{for } d_i - s > 2 \\ 1 & otherwise \end{cases} \tag{14}$$

Proof. In order to derive $LS^{(s)}_{C_i}(G)$, we first consider the case for $s = 0$, i.e. , $LS_{C_i}(G)$.

Let *G* and *G′* respectively denote the original graph G and its neighbor graph by deleting an edge from *G*. For a given node *i*: let $N_\Delta(i)$ and $N_3(i)$ denote the attributes of *i* in *G*; while $N'_\Delta(i)$ and $N'_3(i)$ denote the same attributes in *G′*. By definitioan, we have $0 \le N_\Delta(i) \le N_3(i) = 2/(d_i(d_i-1))$, When deleting an edge from G, $N_\Delta(i)$ would be

decreased by at most $d_i - 1$; while $N_3(i)$ would be decreased by exactly $d_i - 1$. Therefore,

$$LS_{C_i}(G) = \frac{N_\Delta(i)}{d_i(d_i-1)/2} - \frac{N'_\Delta(i)}{d_i(d_i-1)/2 - (d_i-1)} \tag{15}$$

When $0 \leq N_\Delta(i) \leq d_i - 1$, we have

$$LS_{C_i}(G) \leq \frac{N_\Delta(i)}{d_i(d_i-1)/2} \leq \frac{d_i-1}{d_i(d_i-1)/2} = 2/d_i$$

When $d_i - 1 \leq N_\Delta(i) \leq d_i(d_i-1)/2$, we have

$$\begin{aligned} LS_{C_i}(G) &\leq \frac{N_\Delta(i)}{d_i(d_i-1)/2} - \frac{N_\Delta(i)-(d_i-1)}{d_i(d_i-1)/2-(d_i-1)} \\ &= \frac{2}{d_i-2} - \frac{4N_\Delta(i)}{d_i(d_i-1)(d_i-2)} \\ &\leq \frac{2}{d_i-2} - \frac{4(d_i-1)}{d_i(d_i-1)(d_i-2)} = 2/d_i \end{aligned}$$

So that $LS_{C_i}(G) = \frac{2}{d_i}$.

In general case, for $s > 0$, we have (Equ. 7)

$$LS^{(s)}_{C_i}(G) = \max_{G' \in D: d(G,G') \leq s} LS_{C_i}(G') = \frac{2}{d_i - s}$$

for $d_i - s > 2$; and $LS^{(s)}_{C_i}(G) = GS_{C_i}(G) = 1$ otherwise. □

Result 2. ([90]) The local sensitivity at distance s for $N_\Delta(i)$ is

$$LS^{(s)}_{N_\Delta(i)}(G) = \max_{i \neq j; j \in [n]} c_{ij}(s) \tag{16}$$

where

$$c_{ij}(s) = \min(|Ngb(i) \cap Ngb(j)| + \lfloor \frac{s + min(s, b_{ij})}{2} \rfloor, n-2)$$

and $b_{ij} = \sum_{k \in [n]} a_{ik} \oplus a_{kj}$ is the number of half-built triangles involving edge(i,j).

The local sensitivity at distance s for $N_3(i)$ is

$$LS^{(s)}_{N_3(i)}(G) = \min(d_i + s - 1, n - 2) \tag{17}$$

Result 3 shows the formula of the local sensitivity at distance s for the clustering coefficient vector C and Result 4 shows the formulas of the local sensitivity at distance s for vector N_Δ and N_3.

Result 3. The local sensitivity at distance s for C=$(C_1, \cdots, C_n)'$ is

$$LS^{(s)}_C(G) = \max_{i\neq j; i,j\in[n]} \{\min(d_{max} + \lfloor \frac{s + min(s, b_{ij})}{2} \rfloor, n-1)\} \tag{18}$$

where $b_{ij} = \sum_{k\in[n]} a_{ik} \oplus a_{kj}$ is the number of half-built triangles involving edge(i,j)

Proof. We first consider the situation of $s = 0$,

$$\begin{aligned} LS_C(G) &= \max_{a_{ij}=1} \left\{ \frac{2}{d_i} + \frac{2}{d_j} + \sum_{a_{ik}a_{jk}=1} \frac{1}{d_k(d_k-1)/2} \right\} \\ &\leq \max_{a_{ij}=1} \left\{ \frac{2}{d_i} + \frac{2}{d_j} + \frac{2(d_{max}-1)}{d_k(d_k-1)} \right\} \\ &\leq 2\left(\frac{1}{2} + \frac{1}{2} + \frac{2(d_{max}-1)}{2*(2-1)} \right) = d_{max} \end{aligned}$$

$LS^{(s)}_C(G) = LS_C(G) + s$ for $s \leq b_{ij}$ because we may add one edge to complete a half-built triangle involving edge(i, j) which makes the sensitivity increased by at most one; meanwhile, $LS^{(s)}_C(G) = LS_C(G) + \lfloor \frac{s+b_{ij}}{2} \rfloor$ for $s > b_{ij}$ because we have to add two edges to form a triangle to make the sensitivity increased by one, after completing all the b_{ij} half built triangles involving edge(i, j). Besides, $LS^{(s)}_C(G) \leq GS_C(G) = n - 1$. So we have Equ. 18. □

Result 4. The local sensitivity at distance s for N_Δ is

$$LS^{(s)}_{N_\Delta}(G) = 3 \cdot \max_{i\neq j; j\in[n]} c_{ij}(s)$$

where

$$c_{ij}(s) = \min(|Ngb(i) \cap Ngb(j)| + \lfloor \frac{s + min(s, b_{ij})}{2} \rfloor, n-2)$$

and $b_{ij} = \sum_{k\in[n]} a_{ik} \oplus a_{kj}$ is the number of half-built triangles involving edge(i,j) (This result was appeared in [90]).

The local sensitivity of N_3 at distance s is

$$LS^{(s)}_{N_3}(G) = \max_{i\neq j; i,j\in[n]} \left\{ 2n-4, \min(d_{max} + d_{sencondmax} - 2 + \lfloor \frac{s + min(s, b_{ij})}{2} \rfloor) \right\}$$

where $b_{ij} = \sum_{k\in[n]} a_{ik} \oplus a_{kj}$ is the number of half-built triangles involving edge(i,j).

Proof. In addition to the proof of Result. 2 given in [90], $LS^{(s)}_{N_\Delta}(G) = 3 \cdot \max_{i\neq j; j\in[n]} c_{ij}(s)$ because each of the two entries corresponding to vertex *i* and *j* will be decreased by at most $\max_{i\neq j; j\in[n]} c_{ij}(s)$, when edge (i, j) is deleted from *G*. Besides, there are $\max_{i\neq j; j\in[n]} c_{ij}(s)$ other entries whose value will be decreased by one, corresponding to the neighbours in common by vertex *i* and *j*.

For N_3, we first consider the situation of $s = 0$,

$$\begin{aligned} LS_{N_3}(G) &= \max_{a_{ij}=1} \left\{ \frac{d_i(d_i-1)}{2} - \frac{(d_i-1)(d_i-2)}{2} + \frac{d_j(d_j-1)}{2} - \frac{(d_j-1)(d_j-2)}{2} \right\} \\ &\leq \max_{a_{ij}=1} \{d_i - 1 + d_j - 1\} \leq d_{max} + d_{secondmax} - 2 \end{aligned}$$

In general case, $LS^{(s)}_{N_3}(G) = LS_{N_3}(G) + s$ for $s \leq b_{ij}$ and $LS^{(s)}_{N_3}(G) = LS_{N_3}(G) + \lfloor \frac{s+b_{ij}}{2} \rfloor$ for $s > b_{ij}$ which are similar as those of $LS^{(s)}_C(G)$, and $LS^{(s)}_{N_3}(G) \leq GS_{N_3}(G) = 2n-4$. So we have the form for $LS^{(s)}_{N_3}(G)$. □

For vertex clustering coefficient C_i, we have two decomposition strategies: ($N_\Delta(i)$, $N_3(i)$) or ($N_\Delta(i)$, d_i). Similarly for clustering coefficient vector C, we can also have two decomposition strategies: (N_Δ, N_3) or (N_Δ, D). When we apply the second decomposition strategy, we use the global sensitivity of d_i or D because they are very small. However, we should adjust our estimate of $d_i(d_i - 1)/2$, as shown in Lemma 4, if we use the same $\tilde{d}_i$ twice in the calculation. Of course, we can query twice to get two perturbed values of d_i and calculate the unbiased estimate of $d_i(d_i-1)/2$ based on Lemma 2. In this case, two queries of d_i should split the privacy budget assigned to d_i. This example illustrates the importance of deriving unbiased estimate of f from its perturbed values of unit computations.

Lemma 4. The unbiased estimate for the product of the linear combinations of the same perturbed value with Laplace noise is

$$E((u_1 \cdot a + v_1)(u_2 \cdot a + v_2)) = E((u_1 \cdot \tilde{a} + v_1)(u_2 \cdot \tilde{a} + v_2)) - u_1 \cdot u_2 \cdot a'^2$$

Assuming that $a \in \mathbb{R}$ and $\tilde{a} = a + Lap(0, a')$.

Proof. Since $E(Lap(0, a')) = 0$ and $E(Lap^2(0, a')) = a'^2$, $\tilde{a} = a + Lap(0, a')$,therefore

$$\begin{aligned} E((u_1 \cdot \tilde{a} + v_1)(u_2 \cdot \tilde{a} + v_2)) &= E((u_1 \cdot a + v_1)(u_2 \cdot a + v_2)) + u_1 \cdot u_2 \cdot E(Lap^2(0, a')) \\ &\quad + (u_1 \cdot (u_2 \cdot a + v_2) + u_2 \cdot (u_1 \cdot a + v_1))E(Lap(0, a')) \\ &= E((u_1 \cdot a + v_1)(u_2 \cdot a + v_2)) + u_1 \cdot u_2 \cdot a'^2 \end{aligned}$$

□

Table 3: General relativity and quantum cosmology collaboration network dataset statistics

Number of nodes	5242
Number of edges	28980
Nodes in the largest connected component	4158
Edges in the largest connected component	26850
Average clustering coefficient	0.5296
Number of triangles	48260
Fraction of closed triangles	0.6298
Diameter	17

2.5 Empirical Evaluation

In this section, we conduct evaluations to compare the utility of the *direct* approach and the $D\&C_D$ approach on five real graphs and several synthetic graphs generated with three graph models, ER model, WS model and BA model (refer to section 2.2.3).

The five real graphs are denoted respectively as GrQc, Enron, Polbooks, Polblogs, YesIWell. GrQc is the General Relativity and Quantum Cosmology collaboration network from the SNAP Stanford [60]. Table 3 gives some published statistics of the GrQc dataset. We mainly use the GrQc graph data when comparing the utility preservations of different approaches in Section 2.5.1 and exploring the privacy budget distribution in Section 2.5.2. Enron[1] is an email network collected and prepared by the CALO Project and it has 148 nodes and 869 edges; Polbooks[2] is a network of books about US politics published around the time of the 2004 presidential election and sold by Amazon.com and it has 105 nodes and 441 edges; Polblog [1] is a network of hyperlinks between weblogs on US politics; YesIWell is a human physical activities related social network dataset with 185 nodes and 684 edges, which is part of the data gained from the YesIWell study[3] conducted in 2010-2011

[1] $http://www.cs.cmu.edu/\ enron/$
[2] $http://www-personal.umich.edu/\ mejn/netdata/$
[3] $http://aimlab.cs.uoregon.edu/smash/$

as collaboration among several health laboratories and universities to help people maintain active lifestyles and lose weight.

Synthetic graphs are generated using the software Gephi[4] with the Complex Generators plugin. For each graph model (ER, WS, and BA), we generate several graphs by varying graph generation parameters. Table 4 shows some basic characteristics of the generated graphs, where n denotes the number of nodes, m denotes the number of edges, $\bar{c}$ is the average cluster coefficient, n_Δ is the total number of triangles in the graph, fr_Δ is the fraction of the closed triangles, which is defined as the total number of triangles divided by the total number of length two paths. The generation parameters for the ER model are listed as (n, p), those for the WS model as (n, k, p) and those for the BA model as (n, m_0, m_1) (refer to Section 2.2.3). For all synthetic graphs, we fix the number of nodes as 1000.

2.5.1 Utility

We compare our divide and conquer approach with the *direct* approach that directly adds calibrated laplace noise on the output of the computation of f. For vertex cluster coefficient C_i, to examine how different decomposition strategies affect the accuracy of the final output $\tilde{C}(i)$, we include evaluation results on two decomposition strategies: ($N_\Delta(i)$, $N_3(i)$) and ($N_\Delta(i)$, d_i).

Table 5 shows comparisons of these three methods, denoted as *direct*, $D\&C_{N_3(i)}$, and $D\&C_{d_i}$ respectively. In our experiments, we fix $\delta = 0.01$ and vary $\epsilon \in \{0.01, 0.1, 1, 10\}$. We choose the node with the largest degree ($d_i = 81$) for the vertex cluster coefficient. For each of three methods with every privacy setting, we repeat the randomization process for 3,000 times. We report the the mean and standard deviation of the absolute error between

[4] $https://gephi.org/$

Table 4: Statistics for datasets used in Chapter 2

Graphs	n	m	$\bar{c}$	n_Δ	fr_Δ	Generation parameters
ER1	1000	25094	0.0496	20787	0.0496	$(1000, 0.05)$
ER2	1000	50129	0.1002	167671	0.1002	$(1000, 0.1)$
ER3	1000	249755	0.5000	20768657	0.4999	$(1000, 0.5)$
WS1	1000	25000	0.0661	27153	0.0659	$(1000, 50, 0.7)$
WS2	1000	25000	0.1280	52354	0.1272	$(1000, 50, 0.5)$
WS3	1000	25000	0.3886	158301	0.3863	$(1000, 50, 0.2)$
WS4	1000	50000	0.4080	672623	0.4069	$(1000, 100, 0.2)$
WS5	1000	250000	0.5679	23694389	0.5696	$(1000, 500, 0.2)$
BA1	1000	24975	0.0759	64660	0.1274	$(1000, 50, 25)$
BA2	1000	49950	0.1462	443582	0.2189	$(1000, 100, 50)$
BA3	1000	249750	0.6171	31524992	0.6748	$(1000, 500, 250)$
BA4	1000	4995	0.0156	617	0.0308	$(1000, 10, 5)$
BA5	1000	5975	0.0483	20355	0.4162	$(1000, 50, 5)$
BA6	1000	9450	0.1068	162689	0.7614	$(1000, 100, 5)$
BA7	1000	127250	0.6459	20711072	0.9861	$(1000, 500, 5)$
Enron	151	869	0.5018	1700	0.3441	–
Polbook	105	441	0.4875	560	0.3484	–
Polblog	1222	16714	0.3203	101043	0.2260	–
YesIWell	185	342	0.2018	223	0.1710	–

Table 5: Mean and standard deviation ($mean \pm std$) of the absolute error for the clustering coefficient of one node($\delta = 0.01$)

ϵ	$direct$	$D\&C_{N_3(i)}$	$D\&C_{d_i}$
0.01	0.4986 ± 0.1369	0.3656 ± 0.1568	0.4651 ± 0.1510
0.1	0.4695 ± 0.1594	0.3578 ± 0.1800	0.3772 ± 0.1942
1.0	0.0338 ± 0.0332	0.0629 ± 0.0611	0.0549 ± 0.0533
10	0.0036 ± 0.0036	0.0062 ± 0.0057	0.0055 ± 0.0053

$\tilde{C}_i$ and C_i in Table 5.

Table 6: Mean and standard deviation ($mean \pm std$) of all the absolute error for the clustering coefficient vector ($\delta = 0.01$)

ϵ	$direct$	$D\&C_D$	$D\&C_{N_3}$
0.01	0.3257 ± 0.0045	0.2971 ± 0.0045	0.3269 ± 0.0040
0.1	0.1043 ± 0.0021	0.1069 ± 0.0028	0.1398 ± 0.0029
1.0	0.0118 ± 0.0002	0.0145 ± 0.0005	0.0182 ± 0.0006
10	0.0012 ± 0.0001	0.0015 ± 0.0001	0.0019 ± 0.0001

Similarly for clustering coefficient vector C, we use $D\&C_{N_3}$ and $D\&C_D$ to denote divide and conquer approaches based on two decomposition strategies. We set ϵ' here with the magnitude of $n \cdot \epsilon$, where $\epsilon \in \{0.01, 0.1, 1, 10\}$. As a result, each entry of the vector achieves the same (ϵ, δ)-differential privacy as the previous experiment. Table 6 shows our comparisons.

Note that in our experiments, we also use the β-smooth sensitivity in the *direct* approach. This is because the utility is significantly lost if we use the global sensitivity. For example, if we use the the global sensitivity for C_i when $\epsilon = 1$ the error is 0.3558 ± 0.1915, which is significantly larger than 0.0338 ± 0.0332 (shown in Table 5) of the *direct* approach using the smooth sensitivity.

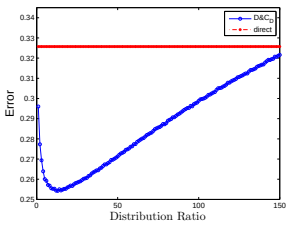
We have following observations from our evaluation results. First, in general, the D&C approach achieves equivalent utility as, if not better than, that of the *direct* approach. This result indicates that we can still enforce differential privacy by decomposing a complex function into unit computations even though the complex function may have a large global sensitivity or may not have an explicit formula of its global sensitivity. Second, for the $D\&C$ approach, querying for the degree sequence D instead of the N_3 vector will probably lead to better utility. This is because the degree sequence has a low global sensitivity.

Third, under the same privacy threshold, it is much better to query for the vector of all the clustering coefficients at once rather than to query for the vertex clustering coefficient one by one.

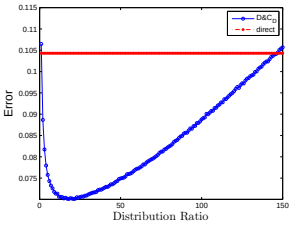
2.5.2 Distribute Privacy Budget

Note that in our previous experiments, we adopted a simple strategy, i.e., distributing privacy budget equally among unit computations. One conjecture is that the $D\&C$ approach would achieve much better utility preservation if we have a better strategy of distributing privacy budget. For example, in our $D\&C_D$ method that obtains the clustering coefficient vector C by querying for the vectors N_Δ and D, the sensitivity magnitude of the vector N_Δ is much larger than that of the vector D. Hence we expect to achieve better utility if we distribute more privacy budget to N_Δ than to D. On the other hand, one characteristic of the division is that the denominator is more sensitive than the numerator, having more influence on the quotient result. As a result, the denominator vector D may need more privacy budget under certain conditions.

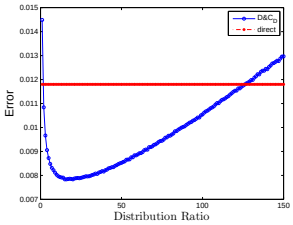
Figure 1 shows how preservation of utility (in terms of approximation error shown as X-axis) varies when we change the ratio of the privacy budget on N_Δ (numerator) and the privacy budget on D (shown as Y-axis) when we apply our $D\&C_D$ method with the total budget $\epsilon = 0.1, 1.0, 10$. The red lines correspond to the *direct* method and the blue curves correspond to the $D\&C_D$ method. The points of those blue curves that are under the red line show the privacy budget distributions with which the $D\&C_D$ method outperforms the *direct* method. Our evaluation results (shown as the third column in Table 6) correspond to the first point (with ratio=1) in each figure. In our future work, we will study the use of Newton iterative method to find out the optimal ratio so that we can achieve optimal utility



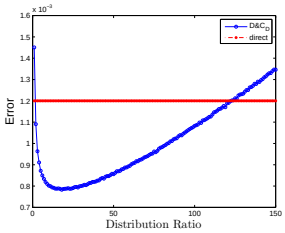
(a) $\epsilon = 0.01, \delta = 0.01$



(b) $\epsilon = 0.1, \delta = 0.01$



(c) $\epsilon = 1.0, \delta = 0.01$



(d) $\epsilon = 10, \delta = 0.01$

Figure 1: Average entrywise absolute error change with the distribution ratio of ϵ', given varying , ϵ, δ

preservation in our divide and conquer approach.

2.5.3 Evaluation on other real graphs and synthetic graphs

In this section, we conduct evaluations to compare the utility of the *direct* approach and the $D\&C_D$ approach on the other four real graphs and various synthetic graphs generated by the three graph models (ER, WS, and BA). In all our experiments, we fix $\delta = 0.01$ and set ϵ' here with the magnitude of $n \cdot \epsilon$, where $\epsilon \in \{0.01, 0.1, 1, 10\}$. As a result, each entry of the vector achieves the same (ϵ, δ)-differential privacy as the previous experiment. For the $D\&C_D$ approach, we simply distribute privacy budget equally, i.e., setting the distribution ratio of privacy budget as 1. As illustrated in Section 2.5.2, we would achieve even better utility preservation for the $D\&C$ approach when we adopt a better strategy of distributing privacy budget.

The evaluation results are shown in Table 7. Specifically, we have the following observations.

Table 7: The average entrywise absolute error for the clustering coefficient vector ($\delta = 0.01$)

Graphs	$\epsilon = 0.01$		$\epsilon = 0.1$		$\epsilon = 1.0$		$\epsilon = 10$	
	$direct$	$D\&C_D$	$direct$	$D\&C_D$	$direct$	$D\&C_D$	$direct$	$D\&C_D$
ER1	0.4920	0.0753	0.2499	0.0016	0.0438	4.08e-5	0.0055	4.07e-6
ER2	0.4947	0.0281	0.3402	2.56e-4	0.0793	2.58e-5	0.0096	2.57e-6
ER3	0.4968	0.0013	0.4697	1.27e-4	0.2832	1.26e-5	0.0393	1.27e-6
WS1	0.4730	0.0472	0.2633	0.0011	0.0423	5.78e-5	0.0049	5.81e-6
WS2	0.4844	0.0904	0.2522	0.0010	0.0455	9.09e-5	0.0047	9.10e-6
WS3	0.4830	0.0721	0.2881	0.0015	0.0424	1.55e-4	0.0042	1.54e-5
WS4	0.4864	0.0162	0.3729	7.55e-4	0.0823	7.55e-5	0.0083	7.57e-6
WS5	0.4967	0.0015	0.4682	1.54e-4	0.2769	1.55e-5	0.0382	1.55e-6
BA1	0.4856	0.1270	0.3389	0.0042	0.0754	4.26e-4	0.0102	4.53e-5
BA2	0.4890	0.0189	0.4005	0.0019	0.1274	1.93e-4	0.0164	1.93e-5
BA3	0.4969	0.0033	0.4739	3.28e-4	0.3071	3.29e-5	0.0499	3.30e-6
BA4	0.1899	0.1377	0.0623	0.0141	0.0094	5.51e-4	0.0011	5.48e-5
BA5	0.2136	0.1433	0.1110	0.0180	0.0182	0.0025	0.0023	2.45e-04
BA6	0.2473	0.1573	0.1634	0.0370	0.0334	0.0057	0.0043	5.86e-04
BA7	0.4792	0.2297	0.4357	0.1813	0.2012	0.0550	0.0284	0.0063
Enron	0.4758	0.4535	0.4283	0.1873	0.2026	0.0436	0.0253	0.0048
Polbook	0.4968	0.4860	0.4263	0.3442	0.1492	0.0438	0.0163	0.0045
Polblog	0.4025	0.2808	0.3537	0.1118	0.1392	0.0336	0.0161	0.0040
YesIWell	0.2253	0.2210	0.1866	0.1586	0.0583	0.0247	0.0065	0.0026

- For the ER model with parameters n and p, ER1, ER2, and ER3 are generated with the same number of nodes $n = 1000$ with increasing p values, $0.05, 0.1, 0.5$. The number of edges (m), the average cluster coefficient ($\bar{c}$), and the fraction of triangle (fr_{Δ}) of these three graphs increase with the increasing of p. For the *direct* approach, we can see that the entrywise absolute error increases with p. On the contrary, the entrywise absolute error decreases for the $D\&C_D$ approach. Thus we can conclude that the $D\&C_D$ approach tends to achieve much better utility preservation than the *direct* approach for ER random graphs. The utility of $D\&C_D$ approach increases as p increases.

- For the WS model with parameters n, k and p, from Section 2.2.3, we know that the graph tends to have more triangles and larger distance when p is near 0 and the graph is more random with less triangles and shorter distance when p approaches 1. WS1, WS2, and WS3 are generated with the same parameters n and k but decreasing p as 0.7, 0.5 and 0.2 respectively. These three graphs have the same density but increasing $\bar{c}$ and fr_{Δ}. We see no clear trend of the absolute entrywise error change for both approaches. WS3, WS4, and WS5 are generated with the same parameters n and p (0.2) but increasing k as 50, 100, and 500 respectively. As a result, these three graphs have increasing m, $\bar{c}$, and fr_{Δ}. For the *direct* approach, we can see that the entrywise absolute error increases with k whereas the entrywise absolute error decreases for the $D\&C_D$ approach. Thus we can conclude that the $D\&C_D$ approach tends to achieve much better utility preservation than the *direct* approach among WS small world graphs and the utility of the $D\&C_D$ approach increases as k increases.

- For the BA model with parameters n, m_0, and m_1, BA1, BA2, and BA3 are generated with the same parameter n and increasing m_1 as 25, 50, and 250 respectively. Recall that in the BA model m_1 denotes the number of newly added edges between each new node and existing nodes. Note that when the ratio between m_0 and m_1 is fixed, the increase of m_1 indicates the increases of m, $\bar{c}$, and fr_{Δ}. For the *direct* approach, we can see that the entrywise absolute error increases with the increase of m_1 whereas the entrywise absolute error decreases for the $D\&C_D$ approach. Thus we can conclude that the $D\&C_D$ approach tends to achieve much better utility preservation than the *direct* approach among BA small world graphs with large density.

- For the BA model with parameters n, m_0 and m_1, BA4, BA5, BA6, and BA7 are generated with the same parameters n and $m_1 = 5$ but increasing m_0 as 10, 50, 100, and 500 respectively. Recall that in the BA model m_0 denotes the number of nodes in the starting set. When m_1 is fixed, the increase of m_0 indicates the increases of $\bar{c}$ and fr_{Δ}. We can see that the utility of the $D\&C_D$ approach is still better than that of the *direct* approach but not as significant as the graphs BA1, BA2, and BA3. The absolute entrywise error for the $D\&C_D$ approach is increasing as parameter m_0 increases, which shows the same trend as that of the *direct* approach. Thus we can conclude that the $D\&C_D$ approach still tends to achieve better utility preservation than the *direct* approach among BA small world graphs with low density but its advantage decreases when m_0 is larger.

- For all four real networks (Enron, Polbook, Polblog and YesIWell), we can see that the $D\&C_D$ approach outperforms the *direct* approach. The extent of the advantage

of the $D\&C_D$ approach is similar as that observed in BA graphs with low density (BA4, BA5, BA6, and BA7).

In summary, we draw the following conclusions. First, the $D\&C_D$ approach outperforms the *direct* approach in all graphs (15 synthetic graphs and four real graphs). Second, the $D\&C_D$ approach shows overwhelming superiority in all three graphs generated by the ER model, all five graphs generated the WS model, and the first three graphs (BA1, BA2, BA3) from the BA model. In the above eleven graphs, the entrywise error of the output of $D\&C_D$ approach tends to much smaller (by several orders of magnitude) than that of the *direct* approach. Third, for real graphs (Enron, Polbook, Polblog, and YesIWell), and the last four graphs (BA4, BA5, BA6 and BA7) generated with the BA model, the advantage of the $D\&C_D$ approach is still obvious and the entrywise error of the output of $D\&C_D$ approach is still smaller (by 10% with small ϵ or by one order of magnitude with large ϵ than that of the *direct* approach. We observe that in the four real networks and the last four BA graphs are relatively sparse and most nodes tend to have a very small magnitude of degree (less than 10) and hence a small number of triangles. When calibrating the noises to N_Δ and D in our $D\&C_D$ approach, the influence due to the distortion is large. As a result, the advantage of our $D\&C_D$ approach decreases when the graphs have low density.

2.6 Summary

Enabling accurate analysis of graph data while preserving differential privacy is of great importance and poses great challenge due to potential high global sensitivity. In this chapter, we have presented a divide and conquer approach that can be used to enforce differential privacy for complex functions. We have conducted theoretical analysis and exten-

sive empirical evaluations to show that the developed divide and conquer approach generally outperforms the approach of directly enforcing differential privacy in terms of utility preservation. This result is especially promising for data mining or exploration tasks with interactive processes, in which a user can adaptively query the system about the data. The user now has options of reusing previous intermediate query results rather than submitting to the system 'new' queries that can be expressed by previous ones. Some preliminary results of this work can be found in [112] and [113].

CHAPTER 3: PRESERVING DIFFERENTIAL PRIVACY IN DEGREE-CORRELATION BASED GRAPH GENERATION

3.1 Introduction

Graph topologies play an irreplaceable role in the network analysis. Previous research in security and privacy has shown potential risks for individual identification with the real graph topologies or the anonymized graph topologies of social networks [5, 52, 126].

There have been attempts [29, 31, 32, 90] to formalize notions of differential privacy in releasing aggregate information about a statistical database and the mechanism to providing privacy protection to participants of the databases. Directly enforcing differential privacy in computing graph properties is challenging due to the high sensitivity. Previous attempts to enforce differential privacy on graph data include computing graph properties such as degree distributions [51] and clustering coefficient [93]. Recently, attempts [87, 95] have been made in enforcing differential privacy in graph generation. The idea is to enforce differential privacy on graph model parameters learned from the original network and then generate the graphs for releasing using the graph model with the private parameters. The released graphs then can be used for various analysis. The authors in [87] tried to generate differentially private graph topology with the stochastic Kronecker graph generation model [73]. However, the stochastic Kronecker graph generation model often cannot accurately capture graph properties of real social networks due to its simplicity in the generation process. The authors in [95] developed a private dK-graph model. The dK-graph model

[81], which constructs graphs to satisfy a family of properties based on various types of node degree correlations, has been shown an effective graph generation model. However, the private 2K-graph model proposed in [95] was based on the local sensitivity of degree correlations due to the large global sensitivity. As a result, the model could not achieve rigorous differential privacy protection.

In this chapter, we present a private 2K-graph generation model that achieves rigorous differential privacy. Our idea is to enforce the differential privacy by calibrating noise based on the smooth sensitivity [90]. By doing this, we achieve the strict differential privacy guarantee with smaller magnitude noise. We conduct experiments on four real networks and compare the performance of our private dK-graph models with the stochastic Kronecker graph generation model in terms of utility and privacy tradeoff. Empirical evaluations show the effectiveness of our proposed private dK-graph models.

3.2 Background

We omit the formal definition and the mechanism of differential privacy here and all of them can be referred from chapter 2. Here we only go over the classic graph generation models.

3.2.1 Graph Generation Models

Over the years, researchers have proposed various graph models to generate graphs that match properties of real networks. Among them, the simplest and most convenient one is the classical E-R random graph $G_{n,p}$ [34], which lays the foundation for the typical stochastic approach [14, 21, 81] to generate graphs. With a given expected average degree $\bar{d}$, we can reproduce an n-sized graph by connecting every pair of n nodes with probability $\bar{d}/n$. In this section, we revisit two widely used graph generation models: the dK-graph

model [81] and the stochastic Kronecker graph model (SKG) [73].

3.2.1.1 dK Graph Model

The dK graph model for graph construction mainly applies pseudograph approach, the most common class of graph generation algorithms [3,88], in constructing graphs matching a desired family of properties called the dK-series in [81]. The dK-series is a finite set of graph properties to describe and constrain random graphs in successively finer detail with the increasing values of d.

The dK-series is defined as the series of properties constraining the generated graph's dK-distribution to be the same form as in a given graph G. dK-distributions are degree correlations within non-isomorphic simple connected subgraphs of size d. For a given graph G, the 0K distribution is simply the average node degree; the 1K distribution is the degree distribution; the 2K-distribution is the joint degree distribution of G which represents the probability that two nodes of degrees k and k' are connected; the 3K-distribution of G is the interconnectivity among triples of nodes. Overall, the dK-series of larger values of d would capture more and more complex detailed properties of the original graph G. In the limit, the dK-distribution describes any given graph completely.

For a given input graph G, the output synthetic 0K-graphs require maintaining the 0K-distribution of G, that is the average node degree; while the output synthetic 1K-graphs reproduce the original graph node degree distribution, and so forth. It is worth pointing out that the degree distribution is different from the degree sequence. The degree sequence is the sequence of length n where each entry $D(i) = d_i$ corresponds to each node's degree whereas the the degree distribution is a distribution vector where each entry $P_1(d_i) = N_{d_i}$ represents the number of nodes whose degree is d_i. Generally, the set of $(d+1)$K-graphs

is a subset of dK-graphs. In the whole space of random graphs, the number of possible graphs satisfying the constrain of dK-series would decrease abruptly with the increase of the value of d.

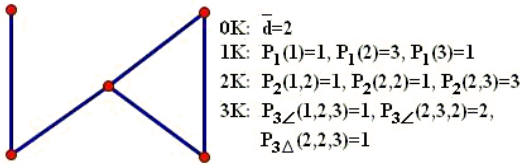


Figure 2: An example of dK-distributions

Figure 2 shows as an example of computing the dK-distributions from a graph G of size 5. For simplicity, the entry value of a dK-distribution is the total number of corresponding d-sized subgraphs. For the given graph G, the 0K-distribution, $P_0 = 2$, is G's average degree; the 1K-distribution, P_1, is G's node degree distribution: $P_1(2) = 3$ means that there are three nodes with degree two; the 2K-distribution, P_2, is the graph G's joint degree distribution; $P_2(2,3) = 3$ means that G contains three edges between 2- and 3-degree nodes; for the 3K-distribution, there are two types of three-sized subgraphs, the triangle and triple that does not form a triangle. In Figure 2, $P_{3\angle}(2,3,2) = 2$ denotes that there are two non-triangle triples where the mid node's degree is three and degrees of the other two nodes are two; while $P_{3\Delta}(2,2,3) = 1$ denotes that there is one triangle formed by three nodes whose degrees respectively are $2, 2, 3$.

The dK-graph model [81] shows surprisingly great performance in capturing global graph structure properties like spectrum and betweenness.

3.2.1.2 Stochastic Kronecker Graph Model

Kronecker graphs [73] are based on a recursive construction process. The process starts with an initiator graph G_1 with N_1 nodes. By a recursive procedure, larger graphs

$G_2, ..., G_n$ are generated. The rth graph generated in the rth recursion, G_r, has about $(N_1)^r$ nodes. Usually, we set $N_1 = 2$. This procedure (Definition 7) is formalized by introducing the concept of Kronecker product (Definition 6) of the adjacency matrices of two graphs.

Definition 6. (Kronecker Product) Given two matrices A and B of size $n \times m$ and $n' \times m'$ respectively, their Kronecker Product is a matrix C of dimensions $(n \cdot n') \times (m \cdot m')$ defined as

$$C = A \otimes B = \begin{pmatrix} a_{11}B & a_{12}B & \cdots & a_{1m}B \\ a_{21}B & a_{22}B & \cdots & a_{2m}B \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1}B & a_{n2}B & \cdots & a_{nm}B \end{pmatrix} \tag{19}$$

Definition 7. (Kronecker Power) Given a Kronecker initiator matrix Θ_1, the kth power of Θ_1 is defined by

$$\Theta_1^{[k]} = \Theta_1^{[k-1]} \otimes \Theta_1 = \Theta_1 \otimes \Theta_1 \cdots \otimes \Theta_1 \tag{20}$$

The Stochastic Kronecker graph (SKG) model was proposed in [73]. In the SKG model, each entry of the initiator matrix Θ takes values in the range $[0, 1]$ instead of binary values, representing the probability of that edge being present. Thus following Definition 7 to compute the Kronecker power of Θ_1, each entry of the reproduced stochastic adjacency matrices represents the probability of that particular edge appearing in the corresponding graph. The final synthetic stochastic Kronecker graph is obtained by choosing edge independently with a probability specified by the corresponding entry in the stochastic adjacency matrix(Definition 8).

Definition 8. (Stochastic Kronecker Graphs [73])If Θ is an $N_1 \times N_1$ probability matrix such

that $\Theta_{ij} \in \Theta$ denotes the probability that edge (i, j) is present, $\Theta_{ij} \in [0, 1]$. Then the kth Kronecker power $P = \Theta^{[k]}$, is a stochastic matrix where each entry $P_{uv} \in P$ encodes the probability of edge (u, v) appearing. This stochastic matrix encodes a stochastic Kronecker graph. To obtain a graph G^*, an instance or realization of the distribution, denoted as $R(P)$, an edge (u, v) is included in $G^* = R(P)$ with probability P_{uv}.

Applying SKG model to a given graph G, i.e. learning model parameters from G, requires the assumption that G is generated by an SKG model with a specific initiator matrix Θ. Extensive researches [44, 72, 87] have been conducted to studying the problem of the estimation of the true parameter, the initiator matrix Θ for G. In [44], the authors proposed an estimation algorithm which made it possible to enforce differential privacy into SKG generation. Based on their approach, recently, differential privately SKG generation [87] has been achieved by first computing the differentially private degree sequence and the total number of triangles from the original graph, and then using them to compute the private input parameters $\{E, H, T, \Delta\}$ of Moment based estimation [44] which are used to finally generate the private graphs.

3.3 Private dK-Graph Model

In this section, we respectively propose the approaches to enforcing differential privacy into the 1K- and 2K-distributions based dK-graph generation models. The basic definitions and terminologies of a graph used in our work are listed in Table 8.

3.3.1 DP-1K Graph Model

The basic idea of our approach to generate differentially private graphs based on the 1K-graph model is to firstly enforce differential privacy in the calculation of the 1K-distribution

Table 8: Basic definitions and terminologies used in Chapter 3

The original graph	G
Number of nodes in G	n
Adjacent matrix of G	A
An entry in A	a_{ij}
The degree of a node i	d_i
The vector of all d_i	D
The degree/1K- distribution	P_1
The 2K-distribution	P_2
The neighboring node set of a node i	$Ngb(i)$
Nodes in $Ngb(i)$ but not in $Ngb(j)$	$Ngb(i) - Ngb(j)$

and then use the private 1K-distribution as the input of the 1K-graph generator to generate the private 2K-graphs.

In order to enforce differential privacy in the calculation of the 1K-distribution, we firstly give the sensitivity of the 1K-distribution in Claim 1. Based on the global sensitivity, we follow Theorem 1 to add Laplacian noise to the real 1K-distribution computed from the original graph with a given privacy parameters ϵ ($\delta = 0$ in this case). Taking the perturbed 1K-distribution as input, the 1K graph model generator could then generate lots of synthetic graphs while satisfying $(\epsilon, 0)$-differential privacy. In our work, we use the graph model generator software provided in [81]. We conclude this process into Algorithm 2.

Claim 1. The global sensitivity of degree distribution (1K-distribution) of a graph G is $GS_{P_1}(G) = 4$.

Proof. When deleteing an arbitrary edge (i, j) from graph G, the following four entries of the degree distribution will be changed by exactly one: d_i,d_j,$d_i - 1$,$d_j - 1$. So that the global sensitivity is 4. □

Another possible approach to compute the private 1K-distribution is firstly querying the private degree sequence(the vector containing each node's degree), whose global sensitiv-

Input: Graph G, privacy parameters (ε)
Output: $\tilde{G}$ satisfies $(\varepsilon, 0)$-differential privacy
1: Compute the 1K-distribution $P_1(G)$of G
2: Using ε to perturb $P_1(G)$ and acquire ϵ-differentially private $\tilde{P}_1(G)$ //Theorem 1
3: Call procedure *1K_graph_generation*$(\tilde{P}_1(G))$ to generate $\tilde{G}$.

Algorithm 2: Private Generation of 1K-graph

ity is 2, and then computing the private 1K-distribution from the private degree sequence. However, the degree sequence is a much longer vector than the 1K-distribution; additionally, every entry of the degree sequence vector would have smaller value than that of the 1K degree distribution. Therefore, the degree sequence vector would suffer more from the relative error though with smaller global sensitivity. With those considerations, we directly query the 1K-distribution from the private graph database server for perturbation.

3.3.2 DP-2K Graph Model

Algorithm 3 illustrates the detail of our differentially private 2K-graph model, DP-2K. The idea is first computing the differentially private 2K-distribution from the original graph, and then using the private 2K-distribution as the input of the 2K-graph generator to generate the private 2K-graphs.

One challenge here is that the global sensitivity of the 2K-distribution is $4n - 7$ (as shown in Claim 2), which is too large to be used for calibrating noise. In [95], the authors explored the use of the local sensitivity to calibrate noise to the 2K-distribution and developed a private 2K-graph model. However, the approach based on the local sensitivity cannot achieve rigorous differential privacy, as shown in [29, 90]. In our algorithm, we use the smooth sensitivity, which is proved to achieve the rigorous differential privacy. We firstly derive the local sensitivity at distance s for the original 2K-distribution $LS^{(s)}_{P_2}(G)$(Claim

2); then compute the smooth sensitivity parameters (β, α) with the given (ϵ, δ) (Line 2) based on Theorem 3; derive the β-smooth sensitivity for P_2 with $\beta, LS^{(s)}_{P_2}(G)$ (Line 3); calibrate noise based on the derived smooth sensitivity and acquire the (ϵ, δ)-differentially private 2K-distribution $\tilde{\tilde{P}}_2$ (Line 4); and finally generate private 2K-graphs (Line 5) with the package provided by [81]. Through this process, our Algorithm 3 achieves rigorous (ϵ, δ)-differential privacy.

Claim 2. The global sensitivity of 2K-distribution is $GS_{P_2}(G) = 4n-7$. The local sensitivity of 2K-distribution is $LS_{P_2}(G) = \max_{i,j\in[n]} 2(d_i + d_j) - 3$. The local sensitivity at distance s of 2K-distribution is $LS^{(s)}_{P_2}(G) = \min\{\max_{i,j\in[n]}\{2(d_i + d_j) - 3 + 2 * s\}, GS_{P_2}(G)\}$

Proof. When deleting an arbitrary edge (i, j) from graph G, the total value change among entries used to involve d_i as one parameter is d_i, for the reason that i leaves the node set of degree d_i; similarly those used to involve d_j will also decrease by d_j in total; specifically, the total amount of decreased value of the above two cases is $d_i + d_j - 1$ since the entry $P_2(d_i, d_j)$ should be only counted once. After deleting edge (i, j), the degree of i, j will be $d_i - 1, d_j - 1$, so that the value of entries that used to involve parameter $d_i - 1$ or $d_j - 1$ will be increased by $d_i - 1 + d_j - 1$ in total. So that the local sensitivity is $LS_{P_2}(G) = \max_{i,j\in[n]} 2(d_i + d_j) - 3$. When $d_i = d_j = n - 1$, we have $GS_{P_2}(G) = 4n - 7$. Every time s increase by one, the d_i or d_j will increase by at most one, so that we have $LS^{(s)}_{P_2}(G) = \min\{\max_{i,j\in[n]}\{2(d_i + d_j) - 3 + 2 * s\}, GS_{P_2}(G)\}$. □

Input: Graph G, privacy parameters (ε)
Output: $\tilde{G}$ satisfies $(\varepsilon, 0)$-differential privacy
1: Compute the 2K-distribution $P_2(G)$ of G
2: Using the (ε, δ) to compute (β, α) //Theorem 3
3: Compute the β-smooth sensitivity $S^*_{P_2,\beta}(G)$ using $\beta, LS^{(s)}_{P_2}(G)$ //Equation 8
4: Compute $\tilde{P}_2(G)$ using $\alpha, S^*_{P_2,\beta}(G)$ //Equation 9
5: Call procedure $2K\_graph\_generation(\tilde{P}_2(G))$ to generate $\tilde{G}$.

Algorithm 3: Private Generation of 2K-graph

3.3.3 DP-3K Graph Model

Through in the limit, the dK-series are expected to describe any given graph completely. In principle, we can develop private dK-graph models for varying d. However, when $d >= 3$, the representation of the dK-distribution is complex, which causes the sensitivity (both global sensitivity and smooth sensitivity) significantly large.

Claim 3 shows the sensitivity values of the 3K-distribution. Recall that $P_{3\angle}(d_1, d_2, d_3)$ and $P_{3\Delta}(d_1, d_2, d_3)$ respectively represent the two types of three-sized subgraphs: the triangle and the triple that does not form a triangle. When an arbitrary edge (i, j) is deleted from graph G, many entries in the 3K-distribution will be affected.

Claim 3. The global sensitivity of the 3K-distribution is

$GS_{P_3}(G) = \frac{3}{2}(n-2)^2 + 2(n-2)$.

The local sensitivity of the 3K-distribution is

$$LS_{P_3}(G) = \max_{i,j\in[n]}\{|S1| + 2|S2| + 2(\sum_{k\in S1}(d_k - 1) - \sum_{k_1,k_2\in T_i,k_1<k_2}(a_{k_1k_2}) - \sum_{k_1,k_2\in T_j,k_1<k_2}(a_{k_1k_2}))\}$$

where $S1 = T_i \cup T_j = \{\{Ngb(i) - Ngb(j)\} \cup \{Ngb(j) - Ngb(i)\}\}$ and $S2 = \{Ngb(i) \cap Ngb(j)\}$.

The local sensitivity at distance s of the 3K-distribution is

$$LS^{(s)}_{P_3}(G) = \min\{GS_{P_3}(G), LS_{P_3(G)} + s + 2\max_{k_q\in S3,t\in i,j}(\sum_{q=1}^{s}(d_{k_q} - \sum_{k_p\in S1t}a_{k_qk_p}) - \sum_{q_1,q_2\in[s];q_1<q_2}a_{k_{q_1}k_{q_2}}(a_{ik_{q_1}}a_{ik_{q_2}} + a_{jk_{q_1}}a_{jk_{q_2}})\}$$

where $S3 = \{V - \{i,j\} - Ngb(i) - Ngb(j)\}$

Proof. We use V to denote the vertex set of original graph G; $T_i = Ngb(i) - Ngb(j)$ denotes the set of nodes i's neighbor excluding those being j's neighbor at the same time; $T_j = Ngb(j) - Ngb(i)$ denotes the set of j's neighbor excluding those being i's neighbor; and $S1 = T_i \cup T_j$ is the set of nodes which are either i's neighbor or j's neighbor but not both; $S2 = Ngb(i) \cap Ngb(j)$ is the set of common neighbors of i and j; $S3$ is the set of nodes which are neither i's neighbor nor j's neighbor.

In the local sensitivity, when edge (i,j) is deleted, there are three cases of change among

entries of P_3.

Firstly, some non-triangle triples will no longer form three-sized subgraphs. Each of such triple involves i,j and one node is $S1$. They are used to be counted in $P_{3<}(d_i, d_j, d_k)$ ($k \in S1$). There are $|S1|$ of them , causing P_3 changed by $|S1|$.

Secondly, some triangle triples will become non-triangle triples. Such triangle is formed with i,j and one of their shared neighbors. They used to be counted in $P_{3\Delta}(d_i, d_j, d_k)$ ($k \in S2$). After deleting (i,j), they are counted in $P_{3<}(d_i - 1, d_j - 1, d_k)(k \in S2)$. There are $|S2|$ of them , causing P_3 change by $2|S2|$.

Thirdly, some triples(no matter it is triangle or not) involve only one of i and j. The entries of P_3 counting them are changed since i and j jumps from the sets of respectively d_i and d_j to those of$d_i - 1$ and $d_j - 1$. The rest part of $LS^{(s)}_{P_3}(G)$ describes this amount.

Therefore we derive the local sensitivity in the form above.When the local sensitivity gets to its maximum,i.e., $d_i = d_j = \frac{1}{2}(n-2) + 1$, $|S2| = 0$, every pair of nodes in $S1$ is connected by an edge, we have

$$\begin{aligned} GS_{P_3}(G) &= 2 \times (n-2)(n-1-1) - 2C^2_{\frac{n-2}{2}} + 0 + (n-2) \\ &= \frac{3}{2}(n-2)^2 + 2(n-2). \end{aligned}$$

For the local sensitivity at distance s, every time s increases by one, we choose one node k_q from $S3$ and add one edge to connect k_q to i or j, it will cause the change of the first case by one, and that of the second case by zero, and that of the third case by two times of the number new three-sized subgraphs brought in by k_q. Thus we have the above form of the smooth sensitivity at distance s. □

Another challenge is that there is no known algorithm to generate dK-graphs for $d >= 3$ given a dK-distribution. The authors in [81] developed an algorithm, the 3K-rewire, for generating 3K-graphs. However, the idea was to modify the original graph G keeping the 3K-distribution unchanged. For private dP-graph models, we can not use the original graph to rewire since the rewired graph may contain other private information than those captured by the dK-distribution. As a result, we only conduct evaluations based on the DP-1K and the DP-2K models.

3.4 Empirical Evaluation

In this section, we conduct evaluations to compare the three graph generation models: the stochastic Kronecker graph (SKG) model, the 1K-graph model, and the 2K-graph model. For the SKG, we use Gleich's [44] and SNAP library [60]'s codes to generate the synthetic graphs with real parameters learned from the original graphs. For both 1K- and 2K-graph models, we use codes provided by [95] for dK-graph generation. We also implemented our private dK-graph models, DP-1K and DP-2K, by following Algorithms 2 and 3 respectively.

We conduct experiments on four graphs: $CA - GrQC$ (denoted as GC), AS20, Enron, and Polbooks. GC is a co-authorship network from arXiv with 5242 nodes and 14484 edges; AS20 is a technological infrastructure network with 6474 nodes and 12572 edges; these two datasets can be downloaded from SNAP[5]. Enron[6] is an email network collected and prepared by the CALO Project and it has 148 nodes and 869 edges; and Polbooks[7] is a network of books about US politics published around the time of the 2004 presidential

[5] $http://snap.stanford.edu/data/index.html$
[6] $http://www.cs.cmu.edu/\ enron/$
[7] $http://www-personal.umich.edu/\ mejn/netdata/$

election and sold by Amazon.com and it has 105 nodes and 441 edges.

3.4.1 Topology Metrics

Table 9: Scalar graph metrics notations in Chapter 3

Metric	Notation
Number of nodes	n
Number of edges	m
Average degree	$\bar{d}$
Assortativity coefficient	r
Average clustering	$\bar{C}$
Average distance	$\bar{l}$
Diameter	l_{max}
Largest eigenvalue of adjacency matrix	λ
Number of triangles	Δ
Transitivity	t
Betweenness	b

Various metrics can be used to measure the graph utility. Refer to a survey [24] for details. The used graph metrics are shown in Table 9.

- The nodes number(n), edges number(m) and average degree($\bar{d}$) describe the basic scale of the graphs.

- The asssortativity coefficient(r) describes the tendency that nodes with similar degree are connected to each other. Assortative (disassortative) networks are those where nodes with similar (dissimilar) degrees tend to be tightly interconnected. They are more (less) robust to both random and targeted removals of nodes and links.

- The betweenness (b) is a commonly used measure of centrality, i.e., topological importance, both for nodes and links. It is a weighted sum of the number of shortest paths passing through a given node or link. As such, it estimates the potential traffic load on a node or link, assuming uniformly distributed traffic following shortest

paths.

- The average distance (l) and the diameter (l_{max}) describe the separation of nodes, which are important for evaluating the performance of routing algorithms as well as of the speed with which worms spread in a network.

- The largest eigenvalue (λ) of the adjacency matrix describes the spectrum character of the graph topology. Eigenvalues provide tight bounds for a number of critical network characteristics [20, 75, 102] including network resilience and network performance like the maximum traffic throughput of the network.

- The average clustering ($\bar{C}$) is the average cluster coefficients of each nodes. The transitivity (t) and the number of triangles (Δ) give the graph level clustering characteristics of the graph.

For each graph model, we generate 100 random graphs and choose one with the largest average clustering coefficient $\bar{C}$. This strategy was adopted in previous works (e.g., [95]) since the variation of $\bar{C}$ from randomly generated graphs is often small.

3.4.2 Evaluation Result

We report all our results for four networks in Tables 10,11,12, and 13. In each table, 'Original' denotes the original graph; 'SKG', '1k', and '2k' respectively denote the graphs generated by the SKG, the 1K-graph model, and the 2K-graph model without privacy protection; '1k(20)', '1k(2)','1k(0.2)' denote the graphs generated by the private DP-1K model with $\epsilon = 20, 2, 0.2$ respectively. Similarly, '2k(2000)', '2k(200)', '2k(20)' , '2k(2)', and '2k(0.2)' denote the graphs generated by the DP-2K model with corresponding ϵ values.

For the DP-2K model, we use the same $\delta = 0.01$. We do not include the results for the private SKG model since they can be acquired from [87]. As shown in Section 3.4.2.1, the SKG model (even without privacy requirement) incurs much larger utility loss than the dK-graph models or the private DP-dK models.

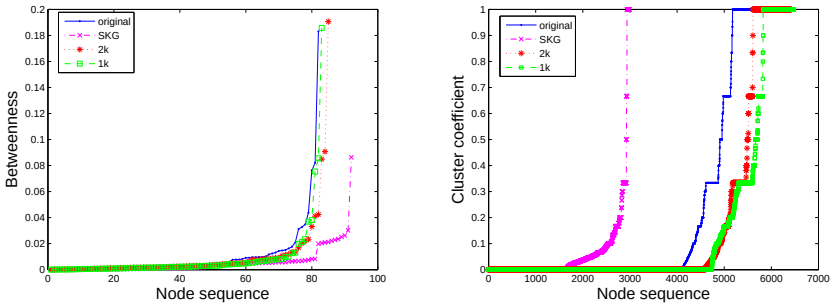
3.4.2.1 SKG Model VS. dK-Graph Model

Our experiment results show that dK-graph models (both 1K and 2K) outperform the SKG model whereas the 2K-graph model generally achieves better utility than the 1K-graph model, when privacy is not taken into account.

The dK-graph models more precisely capture most of the evaluated graph properties than the SKG model. Taking the graph AS20 (6474 nodes,12572 edges) as an example, the 2K-graph model outperforms the SKG model with nine out of the ten metrics used in our evaluation; and the 1K-graph model outperforms the SKG model with seven metrics. The first five columns of Table 11 show the detailed metric values. Compared to the original graph, the relative errors of metrics n, m and d of 1K- and 2K-graph models are around 0.1%, which indicates the dK-graph models can well capture the scale of networks. On the contrary, the SKG model generates graphs that often have different scales than the original one. For example, the relative errors of n, m and d are $54\%, 23\%, 166\%$ for the SKG model. This is because of the limitation of the SKG model that it could only generate graph with node number near 2^r (r is the iterative parameter of the model).

Apart from the global characteristics, the dK-graph models also show better performance in per-node metrics than the SKG model. As the evidence, Figure 3 shows the overlayed patterns of the distribution of the node betweenness (b) and the cluster coefficient sequence (C) of the original graph AS20 as well as its corresponding ones generated by the SKG and

dK-graph models. Figure 3(a) plots the sorted node betweenness distribution, where both the two lines representing 1K- and 2K-graph models are much similar and closer to the line representing the original AS20 graph. Figure 3(b) plots the sorted cluster coefficient sequence, where both 1K- and 2K-graph models more accurately reproduce the cluster coefficient sequence even to the positions of every turning point in the line representing the original graph. On the contrary, both the node betweenness distribution and the cluster coefficient sequence from the SKG graph are significantly different from the original graph.



(a) Betweenness Distribution (b) Cluster Coefficient Sequence

Figure 3: Overlayed patterns of real network for AS20 and generated graphs

We would also point out that neither the dK-graph models nor the SKG model could accurately capture the average cluster coefficient $\bar{C}$, number of triangles Δ, and transitivity t, where in most cases, the relative errors for them are more than 50%. For the assortativity coefficient (r), it could only be precisely captured by the 2K-graph model. We can see from Line 5 of Table 10 that, for graph GC (5242 nodes and 14484 edges), the relative error of r for the 2K-graph is 2.3% while the relative errors for those generated by the SKG and the 1K-graph model are more than 90%.

Finally, we would emphasize that the SKG model cannot achieve utility preservation as well as the dK-graph models. Even our private DP-dK models can achieve better utility

preservation than the SKG model without privacy enforcement. For instance, as shown in Table 12 for the Enron graph, the DP-1K model with strong privacy protection ($\epsilon = 2$) outperforms the SKG model in terms of utility preservation with eight metrics. The DP-2K graph model satisfying $(200, 0.01)$-differential privacy also outperforms the SKG model with eight metrics. In summary, our evaluation demonstrates that the dK-graph models (both 1K and 2K), even with acceptable amount of perturbation, would generate graphs with better utility than the SKG model.

3.4.2.2 Privacy vs. Utility of DP-dK Model

In this section, we focus on the tradeoff between utility and privacy of our private dK-graph models. As shown in Section 3.3.2, the 2K-distribution often has the large sensitivity. When enforcing strong privacy protection with small ϵ values like 0.2 or 2, the DP-2K graph model would incur significant utility loss. For example, the generated graphs often have extremely large scale and uncertain values of other graph metrics. However, for weak privacy protection, the private DP-2K graph model outperforms the DP-1K graph model, i.e., capturing more information of the original graph, especially for the assortativity coefficient (r) and the average cluster coefficient ($\bar{C}$); for other metrics like $n, m, \bar{d}, \lambda, l, l_{max}, t$, the DP-2K model shows at least the same level of accuracy as the DP-1K model.

Figure 4 shows the values of metrics $n, m, t, \bar{C}$ of the Polbooks graph and its corresponding graphs generated by the DP-2K model with varying ϵ values. Each pillar from left to right corresponds to the original graph, the 2K-graph model, and the DP-2K models with varying ϵ values from 2000 to 0.2 respectively. We observe that the magnitude of metric values changes dramatically as ϵ decreases from 20 to 0.2 in all the four metrics, which indicate the significant utility loss for the DP-2K model with strong privacy enforcement.

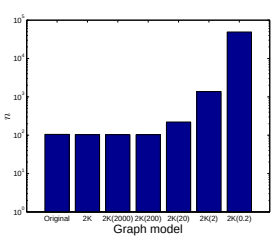
Contrastively, Figure 5 show values for the 1K-model and the DP-1K models with varying ϵ values. We can see that the DP-1K model well preserves the utility even with small ϵ values like 2 and 0.2. We can also observe from Figures 4 and 5 that the DP-2K model achieves better utility preservation than the DP-1K model under the weak privacy enforcement. For example, when $\epsilon = 200, 2000$, the DP-2K model has more accurate $\bar{C}$ than the DP-1K model (no difference for n, m, t).

Additionally, our experiment results show that the assortativity coefficient (r) can only be precisely achieved by the DP-2K model. For example, as shown in Line 5 of Table 10 for the graph GC, the DP-2K graph model with $(200, 0.01)$-differential privacy incurs much smaller loss (with the relative error of 7.8%) than both the 1K-graph model and the DP-1K graph models (with the relative errors more than 90%). The assortativity coefficient is an important metric to describe the tendency that nodes with similar degree are connected.

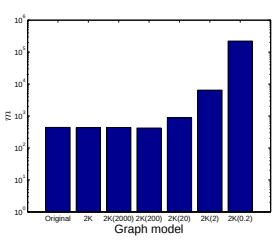
To sum up, our evaluations indicate that the DP-2K graph model generally achieves better utility than the DP-1K graph model for large ϵ values whereas the DP-1K graph model would achieve better utility for small ϵ values.

Table 10: Metrics evaluation of graph GC

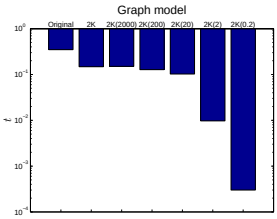
Metrics	Original	SKG	1k	2k	1k(20)	1k(2)	1k(0.2)	2k(2000)	2k(200)	2k(20)	2k(2)
n	5241	3522	5241	4581	5242	5239	5382	4585	4652	6106	153892
m	14484	13767	14484	12748	14509	14596	19430	12724	12949	24415	854067
$\bar{d}$	5.5272	7.8177	5.5272	5.5656	5.5356	5.5720	7.2204	5.5503	5.5671	7.9970	–
r	0.6593	0.0161	0.0177	0.6438	-0.0181	-0.0077	-0.0053	0.6453	0.6080	0.4275	–
$\bar{C}$	0.5297	0.0149	0.0082	0.0183	0.0078	0.0089	0.0159	0.0174	0.0156	0.0117	–
$\bar{l}$	3.8079	3.7890	4.2690	4.3144	4.2205	4.2260	3.8499	4.3149	4.2794	4.0762	–
l_{max}	17	9	11	14	13	12	10	15	16	19	–
λ	45.6167	23.0296	18.2594	41.0398	17.5947	18.1823	27.7705	40.4608	34.3474	38.7852	–
Δ	48260	1585	775	17650	628	745	3035	17149	12303	12485	–
t	0.6298	0.0171	0.0101	0.2690	0.0082	0.0094	0.0179	0.2644	0.1927	0.0646	–
Q	0.8015	0.3176	0.4074	0.5039	0.4044	0.4024	0.3236	0.5069	0.4951	0.4010	–



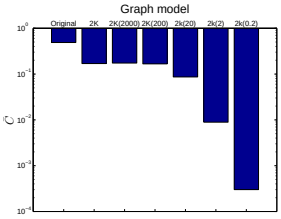
(a) Number of Nodes



(b) Number of Edges

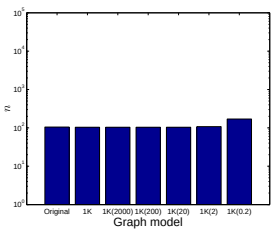


(c) Transitivity

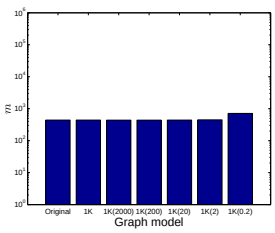


(d) Average Node Clustering

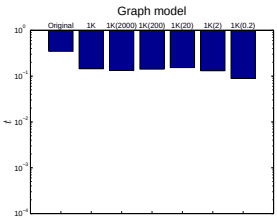
Figure 4: Utility change with varying ϵ on the DP-2K private model generated graphs for Polbooks



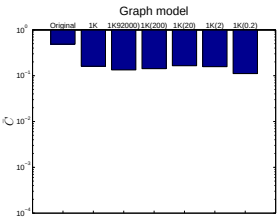
(a) Number of Nodes



(b) Number of Edges



(c) Transitivity



(d) Average Node Clustering

Figure 5: Utility change with varying ϵ on the DP-1K private model generated graphs for Polbooks

Table 11: Metrics evaluation of graph AS20

Metrics	Original	SKG	1k	2k	1k(20)	1k(2)	1k(0.2)	2k(2000)	2k(200)	2k(20)
n	6474	2987	6474	6418	6475	6491	7006	6422	9678	49788
m	12572	15456	12571	12450	12585	15705	38431	12863	30716	240675
$\bar{d}$	3.8838	10.3488	3.8835	3.8797	3.8873	4.8390	10.9709	4.00592	6.34759	9.66799
r	-0.1818	-0.1766	-0.1734	-0.1822	-0.1731	-0.3244	-0.5799	-0.1753	-0.1132	-0.0623
$\bar{C}$	0.2522	0.0811	0.1491	0.1640	0.1485	0.3386	0.5979	0.1134	0.0566	0.0109
$\bar{l}$	3.7050	3.0500	3.2215	3.4481	3.2779	2.9218	2.7337	3.5480	3.7059	4.0273
l_{max}	9	6	12	8	16	9	7	9	8	11
λ	46.3179	39.6021	49.8962	42.8135	50.3928	70.7959	161.7525	41.1401	51.4363	–
Δ	6584	7052	11732	4373	12143	42351	549789	3087	12830	–
t	0.0096	0.0271	0.0171	0.0065	0.01769	0.0303	0.1036	0.0047	0.0129	–
Q	0.6082	0.2505	0.4807	0.5133	0.4782	0.4023	0.2301	0.5053	0.3855	0.3594

Table 12: metrics evaluation of graph Enron

Metrics	Original	SKG	1k	2k	1k(20)	1k(2)	1k(0.2)	2k(2000)	2k(200)	2k(20)	2k(2)	2k(0.2)
n	148	254	148	146	147	153	281	147	146	582	6273	106976
m	869	1804	868	843	867	1024	1538	842	813	3024	29090	512785
$\bar{d}$	11.743	14.318	11.730	11.548	11.796	13.386	10.947	11.455	11.137	10.392	9.274	9.587
r	-0.146	-0.223	-0.062	-0.148	-0.083	-0.077	-0.050	-0.148	-0.145	-0.062	-0.176	-0.019
$\bar{C}$	0.5120	0.2197	0.1893	0.1992	0.1959	0.2429	0.0923	0.1986	0.1850	0.0539	0.0032	0.0020
$\bar{l}$	2.5140	2.2962	2.2951	2.2660	2.2802	2.2403	2.6143	2.2739	2.3071	3.0079	3.9928	4.9975
l_{max}	6	4	4	4	4	5	5	4	5	6	8	8
λ	17.832	22.986	17.793	16.996	17.646	21.719	16.049	16.985	16.216	19.099	19.025	20.522
Δ	1700	1687	821	684	767	1519	599	709	639	845	652	–
t	0.3441	0.1242	0.1671	0.1460	0.1559	0.2106	0.0782	0.1508	0.1502	0.0486	0.0040	–
Q	0.4170	0.1987	0.2245	0.2334	0.2168	0.1895	0.2568	0.2237	0.2575	0.3135	0.3292	0.2787

Table 13: Metrics evaluation of graph Polbooks

Metrics	Original	SKG	1k	2k	1k(20)	1k(2)	1k(0.2)	2k(2000)	2k(200)	2k(20)	2k(2)	2k(0.2)
n	105	128	104	103	104	108	170	103	103	221	1374	49089
m	441	849	440	433	440	448	712	436	421	873	6459	220939
$\bar{d}$	8.40	13.163	8.442	8.408	8.462	8.296	8.377	8.466	8.175	8.275	9.402	9.002
r	-0.128	-0.106	-0.094	-0.129	-0.028	-0.108	0.025	-0.129	-0.184	-0.023	0.029	-0.008
$\bar{C}$	0.4875	0.1773	0.1605	0.1692	0.1649	0.1574	0.1110	0.1737	0.1668	0.0865	0.0089	0.0003
$\bar{l}$	3.0787	2.1323	2.3831	2.3842	2.4099	2.3980	2.6383	2.3775	2.3859	2.7755	3.5576	4.9716
l_{max}	7	4	4	4	4	4	5	4	4	5	8	12
λ	11.9326	18.019	11.871	11.599	12.173	11.586	12.569	11.695	11.104	12.753	13.301	13.587
Δ	560	825	231	230	246	208	239	239	185	318	243	221
t	0.3484	0.1766	0.1449	0.1478	0.1538	0.1309	0.08923	0.1505	0.1273	0.1024	0.0097	0.0003
Q	0.5020	0.1999	0.2694	0.2866	0.2733	0.2706	0.2830	0.2891	0.2974	0.3361	0.3508	0.2855

3.5 Summary

In this chapter, we have presented private dK-graph generation models that enforce rigorous differential privacy while preserving utility. We have conducted theoretical analysis and empirical evaluations to show that the developed private dK-graph generation models significantly outperform the approach based on the stochastic Kronecker generation model. We have shown that the DP-2K graph model generally achieves better utility preservation than the DP-1K graph model with weak privacy enforcement whereas the DP-1K graph model would achieve better utility preservation with strong privacy enforcement. Some preliminary results of this work can be found in [107].

CHAPTER 4: DIFFERENTIAL PRIVACY PRESERVING SPECTRAL GRAPH ANALYSIS

4.1 Introduction

In this chapter, we focus on differential privacy preserving spectral graph analysis. Spectral graph analysis deals with the analysis of the spectra (eigenvalues and eigenvector components) of the graph's adjacency matrix or its variants. There is a large literature on examining the eigenvectors of the adjacency eigenspace, the graph Laplacian or normal matrix for social networks with various applications such as spectral clustering [15, 26, 49, 58, 70, 89, 91, 92, 99, 117, 119–121, 125, 127], fraud detection [118, 129], and spectral graph visualization [9, 57]. In this chapter,we develop two approaches to compute the ϵ-differential private spectra, the first k eigenvalues and the corresponding eigenvectors, from the input graph G. The first approach, denoted as *LNPP*, is based on the Laplace Mechanism [32] that calibrates Laplace noise on the eigenvalues and every entry of the eigenvectors based on their sensitivities. We derive the global sensitivities of both eigenvalues and eigenvectors based on the matrix perturbation theory [101]. Because the output eigenvectors after perturbation are no longer orthogonormal, we postprocess the output eigenvectors by using the state-of-the-art vector orthogonalization technique [41]. The second approach, denoted as *SBMF*, is based on the exponential mechanism [86] and the properties of the matrix Bingham-von Mises-Fisher density for network data spectral analysis [55]. We prove that the Gibbs sampling procedure [55] achieves differential privacy. We conduct empirical

evaluation on a real social network data and compare the two approaches in terms of utility preservation (the accuracy of spectra and the accuracy of low rank approximation) under the same differential privacy threshold. Our empirical evaluation results show that *L*NPP generally incurs smaller utility loss.

4.2 Preliminaries

We omit the formal definition and the classic Laplace mechanism of differential privacy here, since they can be referred from chapter 2. Here we introduce a general mechanism, proposed in [86] by McSherry and Talwar for differential privacy that comes with guarantees about the quality of the output, even for functions that are not robust to additive noise. The idea is to sample from the distribution specified by the exponential mechanism distribution. This mechanism skews a base measure to the largest degree possible while ensuring differential privacy, focusing probability on the outputs of highest value.

4.2.1 Differential Privacy

Exponential mechanism was proposed to achieve differential privacy for diverse functions especially those with large sensitivities [86]. The exponential mechanism is driven by a score function q that maps a pair of input(G) and output(r) from $D^n \times \boldsymbol{R}^d$ to a real valued score($q(G, r)$) which indicates the probability associated with the output. Given an input graph G, an output $r \in \boldsymbol{R}^d$ is returned such that $q(G, r)$ is approximately maximized while guaranteeing differential privacy.

Theorem 4. (The General Exponential Mechanism [86]) For any function q: $(D^n \times \mathbb{R}^d) \rightarrow \mathbb{R}$, based on a query Q with computing function $f : D^n \rightarrow \mathbb{R}^d$, and base measure μ over $\mathbb{R}^d$, the algorithm Υ which takes as input a graph G and some $\alpha > 0$ and outputs some

$r \in \mathbb{R}^d$ is defined as

$$\Upsilon_q^\alpha(G) := Choosing\ r\ with\ probability\ proportional\ to\ exp(\alpha q(G, r)) \times \mu(r).$$

$\Upsilon_q^\alpha(G)$ gives $(2\alpha\Delta q)$-differential privacy, where Δq is the largest possible difference in q when applied to two input graphs that differ only one link, for all r.

4.2.2 Spectral Analysis of Network Topologies

A graph G can be represented as a symmetric adjacent matrix $A_{n\times n}$ with $A_{i,j} = 1$ if there is an edge between nodes i and j, and $A_{i,j} = 0$ otherwise. We denote the i-th largest eigenvalue of A by λ_i and the corresponding eigenvector by u_i. The eigenvector u_i is a $n \times 1$ column vector of length 1. The matrix A can be decomposed as

$$A = \sum_{i=1}^{n} \lambda_i u_i u_i^T. \tag{21}$$

One major application of the spectral decomposition is to approximate the graph data A by a low dimension subspace A_k that captures the main information of the data, i.e., minimizes $\|A - A_k\|_F$. Given the top-k eigenvalues and corresponding eigenvectors, we have a rank-k approximation to A as

$$A_k = \sum_{i=1}^{k} \lambda_i u_i u_i^T = U_k \Lambda_k {U_k}^T, \tag{22}$$

where Λ_k is a diagonal matrix with $\Lambda_{ii} = \lambda_i$ and $U_k = (u_1, .., u_k)$.

U_k belongs to the Stiefel manifold. Denoted as $\nu_{k,n}$, the Stiefel manifold is defined as the set of rank-k $k \times n$ orthonormal matrices. One of the commonly used probability distributions on the Stiefel manifold $\nu_{k,n}$ is called the matrix Bingham-von Mises-Fisher density (Definition 9).

Definition 9. (The matrix Bingham-von Mises-Fisher density [55]) The probability density of the matrix Bingham-von Mises-Fisher distribution is given by

$$\mathbb{P}_{BMF}(X|C_1, C_2, C_3) \propto etr\{C_3^T X + C_2 X^T C_1 X\}, \tag{23}$$

where C_1 and C_2 are assumed to be symmetric and diagonal matrices, repectively.

The matrix Bingham-von Mises-Fisher density arises as a posterior distribution in latent factor models for multivariate and relational data. Recently, a Gibbs sampling scheme was developed for sampling from the matrix Bingham-von Mises-Fisher density with application of network spectral analysis [55] based on the latent factor model(Definition 10).

Definition 10. (The latent factor model for network data [55]) The network data is represented with a binary matrix A so that $A_{i,j}$ is the 0-1 indicator of a link between nodes i and j. The latent factor model with a probit link for such network data is defined as:

$$A_{i,j} = \delta_{(c,\infty)}\left(Z_{i,j}\right)$$

$$Z_{i,j} = u_i^T \Lambda u_j + e_{i,j}$$

$$Z = U\Lambda U^T + E$$

where E is modeled as a symmetric matrix of independent normal noise, Λ is a diagonal matrix and U is an element of $\nu_{k,n}$, with k generally much smaller than n. Given a uniform prior distribution for U, we have

$$\mathbb{P}(U|Z, \Lambda) \propto etr(Z^T U \Lambda U^T/2) = etr(\Lambda U^T Z U/2),$$

which is a Bingham distribution with parameters $C_1 = Z/2$, $C_2 = \Lambda$ and $C_3 = 0$.

Lemma 5. [55]A uniform prior distribution on eigenvectors U and independent normal(0,

τ^2) prior distributions for the eigenvalues Λ give

$$\mathbb{P}(\Lambda|Z,U) = \Pi_{i=1}^{k} normal(\tau^2 u_i^T Z u_i/(2+\tau^2), 2\tau^2/(2+\tau^2))$$

$$\mathbb{P}(U|Z,\Lambda) \propto etr(Z^T U \Lambda U^T/2) = etr(\Lambda U^T Z U/2),$$

where 'normal(u, σ^2)' denotes the normal density with mean u and variance σ^2.

The sampling scheme by Hoff [55] ensures Lemma 5 to approximate inferences for U and Λ for a given graph topology. As suggested in [55], the prior parameter τ^2 is usually chosen as the number of nodes n since this is roughly the variance of the eigenvalues of an $n \times n$ matrix of independent standard normal noise.

4.3 Mechanism for Spectral Differential Privacy

In this section, we present two approaches to computing the ϵ-differential private spectra: *LNPP*, which is based on the Laplace Mechanism (Theorem1), and *SBMF*, which is based on the exponential mechanism [86] and the properties of the matrix Bingham-von Mises-Fisher density for network data spectral analysis [55].

4.3.1 LNPP: Laplace Noise Perturbation With Postprocessing

In this approach, we output the first k eigenvalues, $\lambda^{(k)} = (\lambda_1, \lambda_2, ..., \lambda_k)$, and the corresponding eigenvectors, $U_k = (u_1, u_2, ..., u_k)\big)$, under ϵ-differential privacy with the given graph G and parameters k, ϵ. We first derive the sensitivities for the eigenvalues and eigenvectors in Results 5, 6. We then follow Theorem 1 to calibrate Laplace noise to the eigenvalues and eigenvectors based on the derived sensitivities and privacy parameter. Because the perturbed eigenvectors will no longer be orthogonalized to each other, we finally do a postprocess to normalize and orthogonalize the perturbed eigenvectors following Theorem

5.

Result 5. Given a graph G with its adjacent matrix A, the global sensitivity of each eigenvalue is $GS_{\lambda_i}(G) = 1$,($i \in [1, n]$); the global sensitivity of the first $k(k > 1)$ eigenvalues as a vector, $\lambda^{(k)} = (\lambda_1, \lambda_2, ..., \lambda_k)$, is $GS_{\lambda^{(k)}}(G) = \sqrt{2k}$.

Proof. We denote adding/deleting an edge between nodes i and j on the original graph G as a perturbation matrix P added to the original adjacent matrix A. $P_{n\times n}$ is a symmetric matrix where only $P_{i,j}$ and $P_{j,i}$ have value $1/-1$ and all other entries are zeros. We denote λ_i as the eigenvalue of the matrix A and $\tilde{\lambda}_i$ as that of matrix $A + P$. We have the Euclidean norm and Frobenius norm of P respectively as $\|P\|_2 = 1$ and $\|P\|_F = \sqrt{2}$. Based on the matrix perturbation theory [101](Chapter IV, Theorem 4.11), we have

$$GS_{\lambda_i}(G) \leq \max |\tilde{\lambda}_i - \lambda_i| \leq \|P\|_2 = 1$$

and

$$GS_{\lambda^{(k)}}(G) = \sum_{i=1}^{k} |\tilde{\lambda}_i - \lambda_i| \leq \sqrt{k}\sqrt{\sum_{i=1}^{k}(\tilde{\lambda}_i - \lambda_i)^2} \leq \sqrt{k}\|P\|_F = \sqrt{2k}.$$

□

Result 6. Given a graph G with its adjacent matrix A, the sensitivity of each eigenvector, $u_i(i > 1)$, is $GS_{u_i}(G) = \frac{\sqrt{n}}{\min\{|\lambda_i - \lambda_{i-1}|, |\lambda_i - \lambda_{i+1}|\}}$, where the denominator is commonly referred as the eigen-gap of λ_i. Specifically, the sensitivities of the first and last eigenvector are respectively $GS_{u_1}(G) = \frac{\sqrt{n}}{\lambda_1 - \lambda_2}$ and $GS_{u_n}(G) = \frac{\sqrt{n}}{\lambda_{n-1} - \lambda_n}$.

Proof. We define the perturbation matrix P and other terminologies the same as those in the proof of Result 5. We denote eigenvectors of matrix $A, A + P$ respectively as column

vectors u_i and $\tilde{u}_i$ ($i \in [1, k]$). Based on the matrix perturbation theory [101](Chapter V, Theorem 2.8), for each eigenvector $u_i (i > 1)$, we have

$$GS_{u_i}(G) \leq \sqrt{n}\|\tilde{u}_i - u_i\|_2 \leq \frac{\sqrt{n}\|Pu_i\|_2}{\min\{|\lambda_i - \lambda_{i-1}|, |\lambda_i - \lambda_{i+1}|\}}$$
$$\leq \frac{\sqrt{n}}{\min\{|\lambda_i - \lambda_{i-1}|, |\lambda_i - \lambda_{i+1}|\}}.$$

Specifically for $i = 1$ (similarly for $i = n$),

$$GS_{u_1}(G) \leq \sqrt{n}\|\tilde{u}_1 - u_1\|_2 \leq \frac{\sqrt{n}\|P\|_2}{\lambda_1 - \lambda_2} = \frac{\sqrt{n}}{\lambda_1 - \lambda_2}.$$

□

Theorem 5. (Orthogonalization of vectors with minimal adjustment [41]) Given a set of non-orthogononal vectors $\boldsymbol{x}_1, ..., \boldsymbol{x}_k$, we could construct components $u_1, ..., u_k$ such that $\boldsymbol{x}_i$ is close to u_i for each i, and $U^T U$ is an identity matrix where $U = (u_1, ..., u_k)$ following

$$U = XC,$$

where $X = (\boldsymbol{x}_1, ..., \boldsymbol{x}_k)$ is the set of $n \times 1$vectors and $X^T X$ is non-singular, C is the symmetric square-root of $(X^T X)^{-1}$ and is unique.

Algorithm 4 illustrates our *LNPP* approach. We output the first k eigenvalues, $\widetilde{\lambda}^{(k)} = (\tilde{\lambda}_1, \tilde{\lambda}_2, ..., \tilde{\lambda}_k)$, and the corresponding eigenvectors, $\tilde{u}_1, \tilde{u}_2, ..., \tilde{u}_k)$, under ϵ-differential privacy with the given graph topology A and parameters k, ϵ. We first compute the real values of eigenvalues $\lambda^{(k)}$ and eigenvectors $u_i (i \in [1, k])$ from the given graph adjacent matrix A (Line 1). Then we distribute the privacy parameter ϵ among $\lambda^{(k)}$ and $u_1, u_2, ..., u_k$ respectively as ϵ_0 and $\epsilon_1, \epsilon_2, ..., \epsilon_k$ where $\epsilon = \sum_{i=0}^{k} \epsilon_i$ (Line 2). With the derived the sensitivities

Input: Graph adjacent matrix A, privacy parameter ϵ and dimension parameter k
Output: The first k eigenvalues $\widetilde{\lambda}^{(k)} = (\tilde{\lambda}_1, \tilde{\lambda}_2, ..., \tilde{\lambda}_k)$ and corresponding eigenvectors $\tilde{u}_1, \tilde{u}_2, ..., \tilde{u}_k$, which satisfies ϵ-differential privacy.

1: Decomposition A to obtain the first k eigenvalues $\lambda^{(k)} = (\lambda_1, \lambda_2, ..., \lambda_k)$ and the corresponding eigenvectors $u_1, u_2, ..., u_k$;
2: Distribute ϵ into $\epsilon_0, ..., \epsilon_k$, s.t.$\epsilon = \sum_{i=0}^{k} \epsilon_i$;
3: Follow Theorem 1 to add Laplace noise to $\lambda^{(k)}$ with ϵ_0 based on $GS_{\lambda^{(k)}}(G)$ derived in Result 5 and obtain $\widetilde{\lambda}^{(k)} = (\tilde{\lambda}_1, ..., \tilde{\lambda}_k)$;
4: For i:=1 to k do
Follow Theorem 1 to add Laplace noise to u_i with ϵ_i based on $GS_{u_i}(G)$ derived in Result 6 and obtain $\tilde{\boldsymbol{x}}_i$;
Endfor
5: Normalize and orthogonalize $\tilde{x}_1, ..., \tilde{x}_k$ to obtain $\tilde{u}_1, ..., \tilde{u}_k$ following Theorem 5.
6: Output $\tilde{\lambda}_1, \tilde{\lambda}_2, ..., \tilde{\lambda}_k$ and $\tilde{u}_1, \tilde{u}_2, ..., \tilde{u}_k$

Algorithm 4: LNPP: Laplace noise calibration approach

for the eigenvalues ($GS_{\lambda^{(k)}}(G)$) and each of the k eigenvectors ($GS_{u_i}(G), i \in [1, k]$) from Results 5 and 6, next we follow Theorem 1 to calibrate Laplace noise and obtain the private answers $\widetilde{\lambda}^{(k)}$ (Line 3) and $\tilde{\boldsymbol{x}}_1, \tilde{\boldsymbol{x}}_2, ..., \tilde{\boldsymbol{x}}_k$ (Line 4). Finally we do a postprocess to normalize and orthogonalize $\tilde{\boldsymbol{x}}_1, \tilde{\boldsymbol{x}}_2, ..., \tilde{\boldsymbol{x}}_k$ into $\tilde{u}_1, \tilde{u}_2, ..., \tilde{u}_k$ following Theorem 5 (Line 5).

4.3.2 SBMF: Sampling from *BMF* Density

The *SBMF* approach to provide spectral analysis of network data is based on the sampling scheme proposed by Hoff [55] as an application of their recently-proposed technique of sampling from the matrix Bingham-von Mises-Fisher density (Definitions 9, 10). In [18], the authors investigated differentially private approximations to principle component analysis and also developed a method based on the general exponential mechanism [86]. In our work we focus on the eigen-decomposition of the 0-1 adjacency matrix (rather than the second moment matrix of the numerical data) and prove that the sampling scheme from the matrix Bingham-von Mises-Fisher density satisfies differential privacy through the general exponential mechanism (Theorem 4). The sampling scheme proposed by Hoff [55]

ensures Lemma 5, with the purpose to build the latent factor model (Definition 10) for network data, i.e, to approximate inferences for U and Λ. We derive the privacy bounds of the output eigenvalues and eigenvectors following the sampling scheme respectively in Claims 4 and 5, based on Lemma 5. Then following the Composition Theorem (Theorem 2), we come to the conclusion that the *SBMF* approach satisfies ϵ-differential privacy (Theorem 6).

Claim 4. The sampling scheme which outputs $\lambda^{(k)}$ satisfies ϵ_Λ-differential privacy where $\epsilon_\Lambda = k(\frac{2\tau^2}{2+\tau^2})^{3/2}$.

Proof. We denote A and A' as the adjacent matrix of any neighboring graph G and G'. The calibrated noise to a function f from the Gaussian distribution $normal(0, \sigma^2)$, similar as that from the Laplace distribution, provides a $2\sigma GS_f$-differential privacy [32]. Based on Lemma 5, we have for each eigenvalue λ_i, the sampling scheme satisfies

$$\begin{aligned}\epsilon_{\lambda_i} &= 2\sigma GS_{\lambda_i} = 2(\frac{2\tau^2}{2+\tau^2})^{1/2}\{\tau^2 u_i^T A u_i/(2+\tau^2) - \tau^2 u_i^T A' u_i/(2+\tau^2)\} \\ &= 2(\frac{2\tau^2}{2+\tau^2})^{1/2}\frac{\tau^2}{2+\tau^2}u_i^T(A-A')u_i \leq (\frac{2\tau^2}{2+\tau^2})^{3/2}\end{aligned}$$

where the proof of $u_i^T(A - A')u_i \leq 1$ is straightforward. With the composition theorem (Theorem 2), $\epsilon_\Lambda = \sum_{i=1}^{k} \epsilon_{\lambda_i} = k(\frac{2\tau^2}{2+\tau^2})^{3/2}$.

□

Claim 5. Given the graph G's adjacent matrix A, the sampling scheme which outputs U satisfies ϵ_U-differential privacy where $\epsilon_U = k^2\lambda_1$.

Proof. The sampling scheme for U can be considered as an instance for the exponential

mechanism(Theorem 4) with $\alpha = 1$ and $q(A, U) = tr(\Lambda U^T AU/2)$. We have

$$\Delta q(A, U) = \left|tr(\Lambda U^T AU/2) - tr(\Lambda U^T A'U/2)\right| = \frac{1}{2}\left|tr(\Lambda U^T (A - A')U)\right|$$
$$\leq \frac{1}{2}k\lambda_1 \left|tr(U^T (A - A')U)\right| \leq \frac{1}{2}k^2\lambda_1.$$

Following Theorem 4, we have $\epsilon_U = 2\alpha\Delta q(A, U) = k^2\lambda_1$. □

Theorem 6. The *SBMF* approach to computing the spectra, the first k eigenvalues and the corresponding eigenvectors of a given graph topology A satisfies $\epsilon = (\epsilon_\Lambda + \epsilon_U)$-differential privacy, where $\epsilon_\Lambda = k(\frac{2\tau^2}{2+\tau^2})^{3/2}$ and $\epsilon_U = \alpha k^2\lambda_1$.

In this work, we take the prior parameter τ^2 as n, which is suggested by Hoff [55] since this is roughly the variance of the eigenvalues of an $n \times n$ matrix of independent standard normal noise. We illustrate the *SBMF* approach in Algorithm 5. In the Algorithm, the parameter α is used to change the privacy magnitude by changing ϵ_U (Theorems 4, 6). Given the input graph topology A and dimension parameter k, we acquire the eigenvalues $\widetilde{\Lambda}_k$ and corresponding eigenvectors $\widetilde{U}_k$ from the sampler application provided by Hoff [55] with input matrix αA. The output satisfies $\epsilon = (\epsilon_\Lambda + \epsilon_U)$-differential privacy following Theorem 6.

Input: Graph adjacent matrix $A_{n\times n}$, privacy magnitude α and dimension parameter k
Output: The first k eigenvalues $\widetilde{\Lambda}_k$ and corresponding eigenvectors $\widetilde{U}_k$, which satisfies $\epsilon = (\epsilon_\Lambda + \epsilon_U)$-differential privacy.

1: Set the input matrix $Y = \alpha A$, the parameter $\tau^2 = n$ and the number of iterations t;
2: Acquire $\widetilde{\Lambda}_k$ and $\widetilde{U}_k$ from the sampler provided by Hoff [55] with the input matrix Y, the output satisfies $\epsilon = (\epsilon_\Lambda + \epsilon_U)$-differential privacy(Theorem 6);
3: Output $\widetilde{\Lambda}_k$ and $\widetilde{U}_k$

Algorithm 5: SBMF: Sampling from BMF density approach

4.4 Empirical Evaluation

We conduct experiments to compare the performance of the two approaches, *LNPP* and *SBMF*, in producing the differentially private eigenvalues and eigenvectors. For the *LNPP*, we implement Algorithm 4. For the *SBMF*, we use the R-package provided by Hoff [55]. We use 'Enron' (147 nodes, 869 edges) data set that is derived from an email network [8] collected and prepared by the CALO Project. We take the dimension $k = 5$ since it has been suggested in previous literatures [121] that the first five eigenvalues and eigenvectors are sufficient to capture the main information of this graph. The first two rows in Table 14 show the eigenvalues and their corresponding eigen-gaps (Result 6).

4.4.1 Performance Comparison with $\alpha = 1$

In this section, we compare the performance of the *LNPP* approach with that of the *SBMF* approach in three aspects: the accuracy of eigenvalues, the accuracy of eigenvectors and the accuracy of graph reconstruction with the private eigen-pairs. With $\tau^2 = n$ and $\alpha = 1$, we compute that $\epsilon_\lambda = 14$ and $\epsilon_U = 446$ following Claims 4 and 5. Therefore the *SBMF* approach satisfies $\epsilon = 460$ differential privacy following Theorem 6. On the other hand, the same ϵ is taken as the input for the *LNPP* approach. Different strategies have been proposed to address the ϵ distribution problem(Line 2 in Algorithm 4) in previous literatures [123]. In our work, we just take one simple strategy, distributing ϵ as $\epsilon_0 = 10$ to the eigenvalues and $\epsilon_i = 90, (i \in [1, k])$ equally to each eigenvector. Therefore *LNPP* approach also satisfies $\epsilon = 460$ differential privacy.

For eigenvalues, we measure the output accuracy with the absolute error defined as $E_\Lambda = |\widetilde{\lambda}^{(k)} - \lambda^{(k)}|_1 = \sum_{i=1}^{k} |\widetilde{\lambda}_i - \lambda_i|$. The absolute errors E_Λ for *LNPP* and *SBMF* are

[8] http://www.cs.cmu.edu/~enron/

Table 14: Eigenvalues comparison

	λ_1	λ_2	λ_3	λ_4	λ_5
eigenvalue	17.8317	12.7264	10.6071	9.7359	9.5528
eigen-gap	5.1053	2.1193	0.8712	0.1832	0.1832
LNPP	18.1978	13.2191	10.6030	9.7311	9.4650
SBMF	107.8450	88.9362	76.1712	76.0596	56.6721

Table 15: Eigenvector comparison

Approaches	E_U	$cos\langle\widetilde{u}_i, u_i\rangle = \widetilde{u}_i' \cdot u_i$				
		u_1	u_2	u_3	u_4	u_5
LNPP	11.9989	0.9591	0.7925	0.4786	0.1217	0.1280
SBMF	13.4224	0.6605	0.6995	0.7336	0.2921	0.4034

respectively 0.9555 and 345.2301. One sample o eigenvalues In the third and fourth rows of Table 14, we show the output eigenvalues from the *LNPP* and the *SBMF* approaches. We can see that the *LNPP* outperforms the *SBMF* in more accurately capturing the original eigenvalues.

For eigenvectors, we define the absolute error as $E_U = |\widetilde{U}_k - U_k|_1$. E_U for *LNPP* and *SBMF* approaches are respectively 11.9989 and 13.4224. We also define the cosine similarity to measure the accuracy of each private eigenvector as $cos\langle\widetilde{u}_i, u_i\rangle = \widetilde{u}_i' \cdot u_i (i \in [1, k])$. We show the detailed values of E_U and the cosine similarities in Table15. Note that the cosine value closer to 1 indicates better utility. We can see that *LNPP* generally outperforms *SBMF* in privately capturing eigenvectors that close to the original ones. Specifically, the *LNPP* approach is sensitive to eigen-gaps (second row in Table 14), i.e., it tends to show better utility when the eigen-gap is large such as for u_1 and u_2. Thus a better strategy will be distributing privacy parameter ϵ according to magnitudes of eigen-gaps, instead of the equal distribution.

The *SBMF* approach outputs much larger eigenvalues than the original ones. It does

not tend to accurately approximate anyone of the original eigenvectors either. The reason is that *SBMF* approach is designed to provide a low rank spectral model for the original graph rather than approximating of the original eigenvalues and eigenvectors.

We consider the application of graph reconstruction using the differentially private first k eigenvalues and the corresponding eigenvectors. $A_k = \sum_{i=1}^{k} \lambda_i u_i u_i^T = U_k \Lambda_k U_k^T$ is commonly used as a rank-k approximation to the original graph topology A when A is not available for privacy reasons or A's rank is too large for analysis. Since A_k is not an 0/1 matrix, We discretize A_k as $\widetilde{A}_k^1$ by choosing the largest $2m$ entries as 1 and all others as 0 (so keeping the number of edges m the same as that of the original graph). We then compare the performance of the two approaches by the absolute reconstruction error defined as $\gamma = \|A - \widetilde{A}_k^1\|_F$.The γ values for *LNPP* and *SBMF* approaches are 47.7912 and 34.1760 respectively. We can see that the result of the *SBMF* approach outperforms the *LNPP*.

4.4.2 Performance Comparison with Varying α

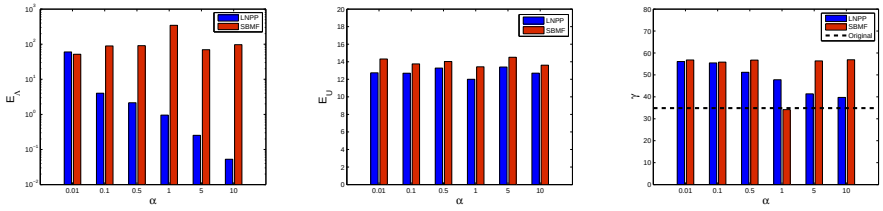
In this section, we change the privacy magnitude to additionally study the performance of the *LNPP* and *SBMF* approaches. α denotes the amplification factor of the privacy parameter ϵ used in section 4.4.1. We choose the value of α as $0.01, 0.1, 0.5, 1, 5, 10$ where the corresponding ϵ values are respectively $18.46, 58.6, 237, 460, 2244, 4474$ following Theorem 6.

We show the values of E_Λ, E_U and γ for the *LNPP* and the *SBMF* approaches in Table 16. The accuracy of the *LNPP* approach increases significantly with α for both the eigenvalues(E_Λ) and graph reconstruction (γ). Note that the greater the α, the weaker privacy protection, and hence the more utility preservation. However, the accuracy of

Table 16: Comparison of two approaches for varying privacy magnitudes

	α	0.01	0.1	0.5	1	5	10
E_Λ	*LNPP*	60.1586	4.0160	2.1452	0.9555	0.2528	0.0527
	SBMF	51.6551	89.0678	90.9442	345.2301	69.6852	96.8904
E_U	*LNPP*	12.7419	13.2455	13.9874	11.9989	13.3967	12.7033
	SBMF	14.3155	13.7518	14.0238	13.4224	14.5114	13.6087
γ	*LNPP*	56.2139	55.4617	51.1859	47.4912	41.3763	39.7492
	SBMF	56.8155	55.8211	56.7450	34.1760	56.3715	56.9210

eigenvectors measured by E_U is not changed much with α, as shown in Figure 6. This is because of the normalization of eigenvectors in the postprocess step. While the *SBMF* approach cannot accurately capture eigenvalues for any α value; as to graph reconstruction, the case of $\alpha = 1$ shows the best utility.



(a) E_Λ (b) E_U (c) γ

Figure 6: Utility comparison for varying privacy magnitude

4.5 Summary

In this chapter we have presented two approaches to enforcing differential privacy in spectral graph analysis. We apply and evaluate the Laplace Mechanism [32] and the exponential mechanism [86] on the differential privacy preserving eigen decomposition on the graph topology. Some preliminary results of this work can be found in [111].

CHAPTER 5: USING AGGREGATE HUMAN GENOME DATA FOR INDIVIDUAL IDENTIFICATION

5.1 Introduction

Genome-wide association studies (GWAS) have received intensive attention due to the rapid decrease of genotyping costs and promising potential in genetic diagnostics. GWAS typically focus on associations between single-nucleotide polymorphisms (SNPs) and human traits like common diseases. It has been shown that many chronic diseases and various cancer types have genetic disposition factors.

High-density genotyping microarrays, and recently next-generation sequencing technologies, have been developed for geneticists to identify common genetic variants that predispose an individual to diseases. In biomedical community, there is a considerable push to make experimental data publicly available so that the data can be combined with other studies or reanalyzed by other researchers. However, genotype data is classified as sensitive and should be handled by complying with specific restrictions, e.g., the Health Insurance Portability and Accountability Act of 1996 (HIPAA) that protects the privacy of individually identifiable health information in the USA. It was shown that only 30-80 out of 30 million SNPs are needed to uniquely identify an individual [77]. Therefore, in addition to the HIPPA privacy rule, the USA Genetic Information Nondiscrimination Act of 2008 (GINA) requires data collectors and supervisory organization must guarantee that the data analysts meet privacy restrictions, and organizations should protect against all forms of ge-

netic discrimination from using individuals' genetic information. Hence, genotype profiles for GWAS participants are only accessible to researchers after confidentiality agreements are signed.

However, Homer et al. [56] developed a method to determine whether a person with known genotypes at a number of markers was part of a sample from which only allele frequencies are known. They showed that GWAS statistics do not completely conceal identity since it is straightforward to assess the probability a person who participated in a particular GWAS cohort. Their study prompted concerns about the public dissemination of genotype data and aggregate statistics from GWAS. Consequently NIH regulated that the database of Genotypes and Phenotypes (dbGaP) [9] has to be accessed by controlled access.

Subsequently, another group [83] investigated the limitation of the method in [56] and analytically assessed the likelihood that an individual who is not in the mixture is mistakenly classified as being a member. These results showed that the method has a high false-positive rate in practice due to its sensitivity to underlying assumptions, limiting its utility for inferring the presence of an individual. An study in [105] proposed two attacks based on the statistics released from GWAS. The first attack extended Homer's attack [56] by utilizing a more powerful statistics (r^2) which describes the correlation among different SNPs, rather than the allele frequencies in Homer's attack. In this attack, only the information of a few hundred of SNPs are needed while in Homer's attack at least 10,000 SNPs are necessary from the target. The other attack gave a way to recover the un-released SNPs of participants by analyzing the r^2 between pairwise SNPs. In [132], the researchers studied the risk of identity inference and genotype recovery of participants by firstly examining

[9] http://www.ncbi.nlm.nih.gov/gap

the common aggregate dataand identifying the typical threat they are faced; and secondly studying possible scenario of attacks on different types of data: the identify attack and the recovery attack. In general, such attacks are hard to succeed due to the lack of enough information or high computation cost. It may still work with special reference population which leads to a small solution space of the problem. Several research works [37, 61] have been conducted for the safe release of aggregate GWAS statistics without compromising a participant's privacy. Their ideas were based on differential privacy [32]. Differential privacy is a paradigm of post-processing the output and is agnostic to auxiliary information an adversary may possess, and provides guarantees against arbitrary attacks. A differentially private algorithm provides an assurance that the output cannot be exploited by the attacker to derive whether or not any individual's record is included. In summary, all of the previous work listed above solely focused on the privacy protection of GWAS participants.

In this chapter, we investigate a related but different privacy protection problem, i.e., whether and to what extent GWAS statistics can be exploited by an attacker to learn private information of regular people (rather than those GWAS participants). Specifically, we study two potential attacks: 1) *trait inference attack* that aims to infer the probability of a target developing some private trait when the target's genotype profile is available to the attacker; and 2) *identity inference attack* that aims to infer the probability of a record in an anonymized genotype database that belongs to the target, when some traits of the target are available.

In *trait inference attack*, the attacker such as an insider from the organizations is assumed to know the whole or part of a target individual's SNP profile and aims to predict some sensitive trait (e.g., disease) of the target individual. In *identity inference attack*, we assume

the attacker such as an outsider has access to the anonymized genotype profile which contains the target individual's record and aims to identify the target individual's record from the anonymized dataset. We also assume the attacker knows some traits of the target individual. For example, private traits and attributes of individuals can be predictable from easily accessible digital records of behavior such as Facebook Likes [68]. Other patient social networks and online communities like 'patientlikeme.com' provide a platform for users (mostly patients) to connect with others who have the same disease or condition and share their own experiences. The data generated in such process may also have potential risks in that the data can be used by the attacker to learn the private traits and attributes of individuals.

Both attacks in our study pose a serious threat to individuals when their genotype profiles are exploited by attackers. Many organizations such as biobanks, hospitals, research consortia and pharmaceutical companies collect and publish DNA sequence and genotype data. For example, the 1000 Genome Project (1000GP) [104] provides the public with free services like browsing and downloading DNA sequences, SNP genotypes and other types of data from over a thousand anonymous participants in different populations. Although without any specific traits, these data can be exploited by attackers where they can combine such public information with health records and other online information. One of such studies [48] demonstrated end-to-end identification of individuals with only public information and showed that full identities of personal genomes can be exposed via surname inference from recreational genetic genealogy databases followed by Internet searches. They considered a scenario in which the genomic data are available with the target's year of birth and state of residency, two identifiers that are not protected by HIPAA. From a different

angle but along the same line, our study here further shows that the re-identification of anonymized genotype data still hold a real threat to regular individuals who are not GWAS participants using published data.

The contributions of our study is as follows. 1) We develop a method to build a two-layer bayesian network from the released GWAS statistics. The constructed bayesian network explicitly reveals the conditional dependency between SNPs and traits, and can be used to compute the probability distribution for any subset of network variables given the values or distributions for any subset of the remaining variables. 2) We then formulate two attacks, namely trait inference attack and identity inference attack, as two inference problems based on the dependency relationship captured in the bayesian network, and develop efficient formulas and algorithms to infer the probability of attacks. 3) We conduct empirical evaluations of the proposed methods. Our results show that unexpected privacy breaches can occur because aggregation statistics (or even perturbed statistics under differential privacy) provide no explicit security guarantees and these statistics could be exploited by attackers to identify individuals or derive private information.

5.2 Background

5.2.1 GWAS Catalog and Statistics

Case-control studies under the GWAS framework are usually conducted by comparing the genotypes of two groups of participants: individual with the disease (case group) and matched individuals without the disease (control group). Each individual is genotyped by microarray or sequencing platforms. Dependent on genotyping platform, the number of SNPs genotyped in a GWAS setting typically ranges from tens of thousands to tens of millions. From genotype data, we can view that an SNP locus has two possible alleles, a

risk allele and a non-risk allele. The risk allele is the allele that is more frequent in the case group comparing with the control group.

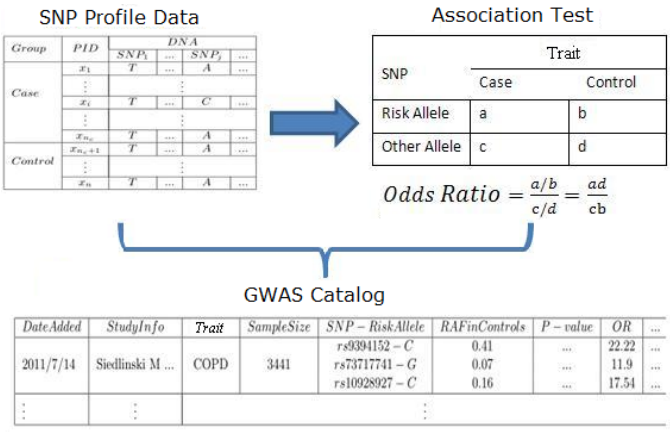


Figure 7: Typical process of a genome-wide association study

Figure 7 shows a typical GWAS process. Firstly, SNP profile data is generated by genotyping the individuals in cases and controls. Secondly, Allele frequency for each of those SNPs over the case group and the control group is calculated respectively and a statistical test is performed on a contingency table to investigate if the allele frequencies are significantly different in cases versus controls. The upper-right table in Figure 7 depicts a 2×2 contingency table for allele counts in case and control groups. The odds ratio, which is defined as the ratio of the proportion of individuals in the case group having a specific allele, and the proportion of individuals in the control group having the same allele, is often used to report the difference. When the allele frequency in the case group is much higher than in the control group, the odds ratio will be higher than 1. Additionally, a p-value for the significance of the odds ratio is typically calculated using a simple chi-squared test. Finding SNPs whose odds ratios are significantly different from 1 is the objective of the GWAS because those SNPs are associated with the trait. Finally, those SNPs that are associated

with the trait, along with the statistics (e.g. *p*-value and odds ratio) are reported. These reported SNPs, along with information about the study, the trait, specific SNP information (e.g. identifier, position, and the risk allele type), and statistics, are later collected and curated at the National Human Genome Research Institute (NHGRI) GWAS catalog [54], as shown in the lower table in Figure 7.

5.2.2 Bayesian Network Revisited

A bayesian network $G = (V, E)$ is a Directed Acyclic Graph (DAG), where the nodes in V represent the variables and the edges in E represent the dependence relationships among the variables. The dependence/independence relationships are graphically encoded by the presence or absence of direct connections between pairs of variables. Hence a Bayesian network shows the (in)dependencies between the variables qualitatively, by means of the edges, and quantitatively, by means of conditional probability distributions which specify the relationships. In general, a bayesian network represents the joint probability distribution by specifying a set of conditional independence assumptions together with sets of local conditional probabilities. The edges represent the assertion that the variable is conditionally independent of its nondescendants in the network given its immediate predecessors in the network. A conditional probability table is given for each variable, describing the probability distribution for that variable given the values of its immediate predecessors. Formally, for each variable $X_i \in V$, we have a family of conditional probability distributions $P(X_i|Parent(X_i))$, where $Parent(X_i)$ represents the parent set of the variable X_i in G. From these conditional distributions we can compute the joint probability for any desired assignment of values $< x_1, x_2, \cdots, x_n >$ to the tuple of network variables

$X_1, X_2, \cdots, X_n$ by the formula:

$$P(x_1, x_2, \cdots, x_n) = \prod_{i=1}^{n} P(x_i | Parent(X_i)) \tag{24}$$

Note the values of $P(x_i|Parent(X_i))$ are precisely the values stored in the conditional probability table associated with variable X_i. Bayesian networks can perform efficiently reasoning tasks and there are several algorithms (including exact inference methods and approximate inference methods) to compute the posterior probability for any variable given the observed values of the other variables in the graph.

5.3 Constructing a Bayesian Network from GWAS

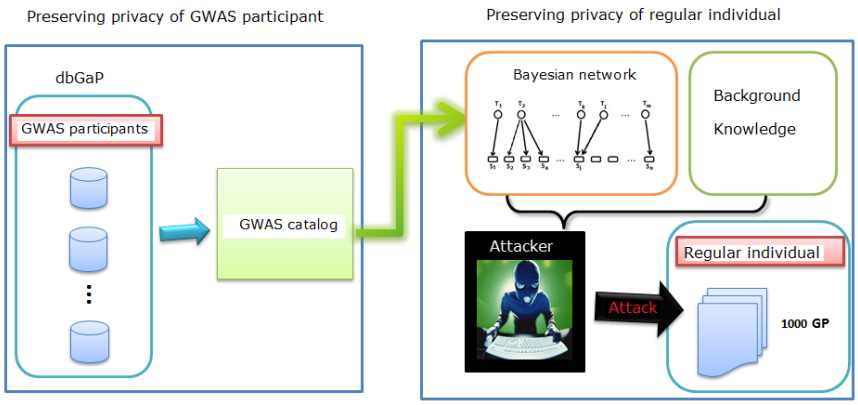


Figure 8: Infringement of individual privacy for GWAS participants and regular individuals

In this section, we describe the implementation of our methods that potentially disclose individual privacy for GWAS participants and regular individuals as well. As illustrated in Figure 8, we consider the infringement of individual privacy for both GWAS participants (e.g., whose genotypes are stored in dbGaP and under controlled access) and regular individuals (including but not limited to individuals whose genotypes are publicly available). Specifically, we extract summary statistics of risk alleles from GWAS catalog [54], build

a two-layered bayesian network from the aforementioned GWAS catalog, and present how an attacker can attack the identity of regular individuals by inferring on the constructed bayesian network with aid of publicly available background information.

We elaborate how to build a two-layered bayesian network using the statistics extracted from the aforementioned GWAS catalog. The constructed bayesian network, which explicitly captures the conditional dependency between SNPs and their associated traits, will be used as background knowledge for inference attacks.

5.3.1 Knowledge from GWAS Catalog

We use the information publicly available from the GWAS catalog [54] to construct the bayesian network. Such information includes trait/disease name, the associated SNPs and corresponding risk allele type, the risk allele frequency in control group, and statistics (e.g., odds ratio and p-value) in the association test of each SNP. Specifically, we extract the following data from the GWAS catalog: a trait set $\mathbb{T}$, which contains m traits, and an SNP set $\mathbb{S}$, which contains n SNPs. For each specific trait $T_k \in \mathbb{T}$, we have a subset of associated SNPs. For each associated SNP S_j, we can extract its corresponding risk allele type ($rSNP_{kj}$) associated trait T_k, the odds ratio O_{kj} of the association test, and the risk allele frequency in the control group f^t_{kj}.

Though not directly given in the GWAS catalog, the risk allele frequency in the case group can be derived from the corresponding odds ratio and the risk allele frequency in the control group. For an SNP S_j associated with a trait T_k, its odds ratio is

$$O_{kj} = \frac{f^c_{kj}(1 - f^t_{kj})}{f^t_{kj}(1 - f^c_{kj})} \tag{25}$$

With the released values of the odds ratio (O_{kj}) and the risk allele frequency in the control

group f_{kj}^t, the risk allele frequency in the case group f_{kj}^c can be derived as

$$f_{kj}^c = \frac{O_{kj} \cdot f_{kj}^t}{O_{kj} \cdot f_{kj}^t + 1 - f_{kj}^t} \tag{26}$$

Lemma 6. The background knowledge that an attacker can obtain from the GWAS catalog [54] includes: a trait set $\mathbb{T}$, an SNP set $\mathbb{S}$, the risk allele type ($rSNP_{kj}$), the odds ratio O_{kj}, and the risk allele frequency in the control group f_{kj}^t for each pair of trait and its associated SNPs.

5.3.2 Two-layered Bayesian Network Construction

In GWAS, we can distinguish between two different sets of variables: the set $\mathbb{T}$ of the m traits, T_k, and the set $\mathbb{S}$ of the n SNPs, S_j. Each trait T_k is a binary random variable taking values in the set $\{t_k, \bar{t}_k\}$, where $t_k(\bar{t}_k)$ stands for the presence (absence) of the trait of a participant. Similarly, each SNP S_j has its domain in the set $\{s_j, \bar{s}_j\}$, where s_j stands for the SNP has the risk allele and $\bar{s}_j$ otherwise.

We construct a bayesian network to represent the conditional dependencies between traits and SNPs, with background knowledge shown in Lemma 6. The constructed network is composed of two layers, the trait layer and the SNP layer, with edges only going from trait nodes to SNP nodes. As shown in Figure 9, in such a network, each node at the top level denotes a specific trait; while each node at the second level denotes an SNP. If an SNP(S_j) is associated with a trait(T_k), a directed edge is added from T_k to S_j. Focusing on the network structure, we consider the following guidelines to determine network topology:

- For each trait in the GWAS catalog, there is a directed link from the node representing that trait to each of its associated SNP node.

- The traits are marginally independent from each other, which implies that there are no links between trait nodes. All trait nodes are root nodes.

- There are no links between the SNP nodes. The relationships between SNPs only occur through the associated traits. SNPs are conditionally independent given the traits that they are associated with.

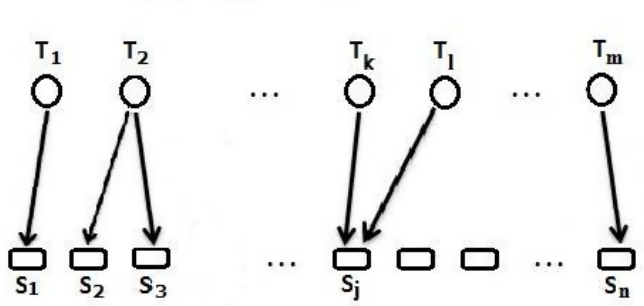


Figure 9: A two-layered Bayesian network of traits and associated SNPs

The next step to completely specify a bayesian network is to determine the conditional probability table stored at each node. Firstly, we need to acquire the prior probability of each trait node at the top level of the network. The prevalence of a trait T_k in the population can be obtained from the literature or internet. We treat it as the prior probability that one individual has this trait, denoted as $P(t_k)$. Secondly, we need to determine the conditional probability table of each SNP node at the second level. There are two types of SNP nodes in terms of dependency relationship with traits: a) SNP nodes which have a single parent trait node, e.g., S_1, S_2 in Figure 9; and b) SNP nodes which have more than one parent trait node, e.g., S_j in Figure 9.

Figure 10 shows the histogram distribution of SNPs which are associated with more than one trait in the GWAS catalog. Specifically, among the $10,027$ unique SNPs, there

Table 17: Probability function for SNPs with single parent trait

$P(S_1\|T_1)$		T_1	
		t_1	$\bar{t}_1$
S_1	s_1	f_{11}^c	f_{11}^t
	$\bar{s}_1$	$1 - f_{11}^c$	$1 - f_{11}^t$

are $9,447$ SNPs associated to a single trait, 368 SNPs associated with two traits, and at the most, two SNPs associated with 11 traits.

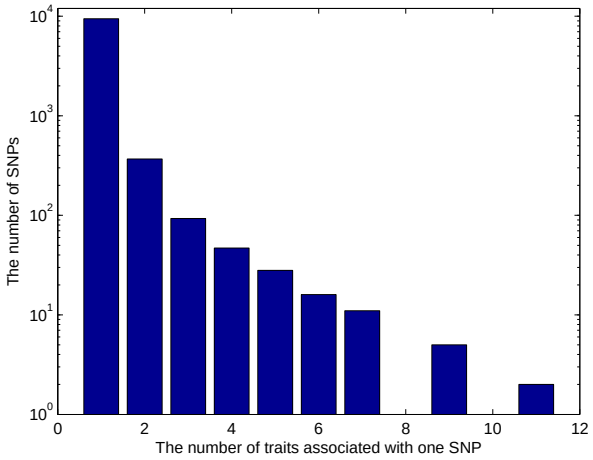


Figure 10: Distribution of SNP-trait associations

For those SNP nodes with a single parent trait node, we can specify the values of the conditional probability table associated with each SNP node by its risk allele frequency in the control group and the risk allele frequency in the case group. An example for node $S1$ is shown in Table 17. The probability of the risk allele of SNP $S1$ given the presence of the trait $T1$, $P(s_1|t_1)$ equals the risk allele frequency in the case group f_{kj}^c. Note that the conditional probability table of S_2 can be specified in the same way as S_1, although S_2 shares the parent trait node T_2 with other SNP nodes (i.e., S_3 and S_4).

For the SNP nodes with multiple parent trait nodes, the conditional probability table cannot be built from the GWAS catalog directly. Instead, the value of each cell in the con-

Table 18: Probability function for an SNP with two parent traits

$P(S_j\|T_k,T_l)$		T_k,T_l			
		t_k,t_l	$t_k,\bar{t}_l$	$\bar{t}_k,t_l$	$\bar{t}_k,\bar{t}_l$
S_j	s_j	$\frac{f^c_{kj}f^c_{lj}}{f_{s_j}}$	$\frac{f^c_{kj}f^t_{lj}}{f_{s_j}}$	$\frac{f^t_{kj}f^c_{lj}}{f_{s_j}}$	$\frac{f^t_{kj}f^t_{lj}}{f_{s_j}}$
	$\bar{s}_j$	$\frac{(1-f^c_{kj})(1-f^c_{lj})}{1-f_{s_j}}$	$\frac{(1-f^c_{kj})(1-f^t_{lj})}{1-f_{s_j}}$	$\frac{(1-f^t_{kj})(1-f^c_{lj})}{1-f_{s_j}}$	$\frac{(1-f^t_{kj})(1-f^t_{lj})}{1-f_{s_j}}$

ditional probability table, which represents one of the possible combinations of its parent nodes being true or false, can be calculated based from Lemma 7.

Lemma 7. (Conditional probability for SNPs with multiple parent trait nodes) For a specific risk SNP S_j associated with a subset of traits $Parent(S_j)$, we have

$$P(S_j|Parent(S_j)) = \frac{\prod_{T_k \in Parent(S_j)} P(S_j|T_k)}{P^{q-1}(S_j)}, \tag{27}$$

where q is the number of elements in $Parent(S_j)$. Specifically, when S_j is associated with two traits T_k, T_l, we have

$$P(S_j|T_k,T_l) = \frac{P(S_j|T_k)\cdot P(S_j|T_l)}{P(S_j)} \tag{28}$$

Proof.

$$\begin{aligned}
P(S_j|T_k,T_l) &= \frac{P(T_k,T_l|S_j)\cdot P(S_j)}{P(T_k,T_l)}(Bayes'sLaw)\\
&= \frac{P(T_k|S_j)\cdot P(T_l|S_j)\cdot P(S_j)}{P(T_k)P(T_l)}(Conditionalindependence)\\
&= \frac{\frac{P(S_j|T_k)P(T_k)}{P(S_j)}\cdot\frac{P(S_j|T_l)P(T_l)}{P(S_j)}\cdot P(S_j)}{P(T_k)P(T_l)} \quad (Bayes'sLaw)\\
&= \frac{P(S_j|T_k)\cdot P(S_j|T_l)}{P(S_j)}
\end{aligned}$$

The proof for the general case is straightforward. □

Table 18 shows an example for node S_j which has two parent trait nodes, T_k and T_l. Note that values in Table 18 are suitable for the situation where the risk allele type of S_j

for T_l is the same as that for T_k; otherwise we should exchange all the f^c_{lj} with $1 - f^c_{lj}$ in each cell of the table. When we calculate the first cell value in the conditional probability table, $P(s_j|t_k, t_l)$, we simply set $P(s_j|t_k) = f^c_{kj}$, $P(s_j|t_l) = f^c_{lj}$, and $P(s_j) = f_{s_j}$ where f^c_{kj} and f^c_{lj} denote the risk allele frequency in each case group and f_{s_j} denotes the probability of T_j's risk allele appearing in the population. In Lemma 7, we assume that the traits are conditionally independent with each other, given the SNPs they are both associated with. The probability that one allele of an SNP appears in the population, f_{s_j}, can be found from an NCBI website[10].

With the bayesian network constructed from the GWAS catalog, we can calculate the joint probability for any desired assignment of values to variables sets $\mathbb{S}$ (SNPs), $\mathbb{T}$ (traits), for example $< s_1, s_2, ..., s_{|\mathbb{S}|}, t_1, t_2, ...t_{|\mathbb{T}|} >$, following Lemma 8. In the original reasoning process in the bayesian network, we need to involve all of the other variables to calculate $P(\mathbb{S}, \mathbb{T})$. By marginalization of summing out 'irrelevant' variables, we achieve the form in Lemma 8.

Lemma 8. The joint probability for any desired assignment of values to variables sets $\mathbb{S}$ (SNPs), $\mathbb{T}$ (traits), for example $< s_1, s_2, ..., s_{|\mathbb{S}|}, t_1, t_2, ...t_{|\mathbb{T}|} >$, can be calculated using Equation 29. Note that we can apply this equation to any other assignment of values, simply changing s_i/t_j to $\bar{s}_i$/$\bar{t}_j$ accordingly.

$$\begin{aligned} P(\mathbb{S}, \mathbb{T}) &= P(s_1, s_2, ..., s_{|\mathbb{S}|}, t_1, t_2, ...t_{|\mathbb{T}|}) \\ &= \sum_{T_k \in \{t_k, \bar{t}_k\}} (\prod_{i=1}^{|\mathbb{S}|} P(s_i|Parent(S_i)) \prod_{j=1}^{|\mathbb{T}|} P(t_j) \prod_{k=1}^{|\mathbb{T}'|} P(T_k)) \end{aligned} \tag{29}$$

where $\mathbb{T}'$ denotes the set of all the other parent traits of the SNPs in $\mathbb{S}$ except for those

[10] $http://www.ncbi.nlm.nih.gov/snp/$

already contained in $\mathbb{T}$.

Additionally, we can calculate the conditional joint probability for any *desired* assignment of values to variables sets $\mathbb{S}_x, \mathbb{T}_x$ given the *observed* assignment of variables sets $\mathbb{S}_y, \mathbb{T}_y$ following Theorem 7. Note that $\mathbb{S}_x$ and $\mathbb{S}_y$ denote the set of SNPs; while $\mathbb{T}_x$, $\mathbb{T}_y$ denote the set of traits.

Theorem 7. (Inference via a GWAS Bayesian Network) The joint probability for any desired assignment of values to variables in $\mathbb{S}_x, \mathbb{T}_x$ given the (observed) assignment of values to variables in $\mathbb{S}_y, \mathbb{T}_y$ can be derived from Equation 30.

$$P(\mathbb{S}_x, \mathbb{T}_x | \mathbb{S}_y, \mathbb{T}_y) = \frac{P(\mathbb{S}_x, \mathbb{T}_x, \mathbb{S}_y, \mathbb{T}_y)}{P(\mathbb{S}_y, \mathbb{T}_y)} \tag{30}$$

where the joint probability $P(\mathbb{S}_x, \mathbb{T}_x, \mathbb{S}_y, \mathbb{T}_y)$ and $P(\mathbb{S}_y, \mathbb{T}_y)$ can be calculated following Lemma 8.

5.4 Inference Attacks based on a Two-layered Bayesian Network

5.4.1 Trait Inference Attack

In this attack scenario, we assume that an attacker has stolen genotype profile of the target and aims to derive the probabilities that the victim has specific traits using the constructed bayesian network. Formally, we represent the genotype of a target v as a vector, $r_v = (r_{v1}, r_{v2}, \cdots, r_{vn})$, with each entry r_{vj} denoting the allele type of SNP j. The attacker aims to learn the posteriori probability $P(t_k|r_v)$ that the target has a specific trait T_k given the target's genotype profile r_v using the constructed bayesian network.

Definition 11. (Trait Inference Attack) Assume that the attacker has the genotype profile r_v of the target v. The attacker aims to learn the posteriori probability $P(t_k|r_v)$ that the target

has a specific trait T_k given the target's genotype profile r_v using the constructed bayesian network.

The probability of the prevalence of a specific trait, which is retrievable from the literature or the internet, is used as the prior probability that the target has the specific trait. The attacker can improve his/her guess by calculating the posterior probability of the target having the trait by inferring from with the target's genotypes.

The attacker can calculate the posterior probability of the target having a particular trait ($P(t_k|r_v)$), using the victim's genotype information (r_v) and Lemma 9.

Lemma 9. (Trait Development Risk Estimation With Several Related SNPs) The posteriori probability $P(t_k|r_v)$ can be calculated following Equation 30, specifically with $\mathbb{S}_x, \mathbb{T}_y = \emptyset$, $\mathbb{T}_x = T_k = t_k$. Based on conditional independence, we have $\mathbb{S}_y$ that contains only the SNPs associated with T_k, where the value assignment of SNP genotypes is equal to the corresponding genotype record of the target individual.

In the attack scenario described in Algorithm 6, the attacker intends to find out the possibility that the victim has certain trait ,according to his/her genotype and the GWAS catalog information.

5.4.2 Identity Inference Attack

In identity inference attack, we assume that the attacker has access to an anonymized genotype dataset that contains the target's genotype record and the attacker knows a subset of traits the target has. Formally, we denote the anonymized genotype profile dataset as r, where each record $r_i = (r_{i1}, r_{i2}, \cdots, r_{in})$ represents the genotype profile of an anonymized individual i. We assume that the genotype profile of the target r_v is contained in r, and the

Input: The genotype profile r_v of an individual v, the GWAS Bayesian Network G, the trait set $\mathbb{T}$
Output: The probability $P(T_k|r_v)$ that the individual has any trait in $\mathbb{T}$

1: FOR each trait T_k in $\mathbb{T}$
2: Search G for T_k and obtain the associated SNPs $\{S_j\}$ (j=1..m) and corresponding risk allele type;
3: Extract the subgraph of T_k, SNP set $\{S_j\}$ (j=1..m) and all the other parent traits of these SNPs from the constructed bayesian network.
4: Obtain the binary values of r_{vj} for each j from 1 to m according to whether the victim has the risk allele type of each S_j in r_v;
5: Calculate $P(T_k|r_v)$ following Lemma 9.
6: ENDFOR

Algorithm 6: Trait Inference

attacker knows $\mathbb{T}_S$, a subset of traits the target has.

Definition 12. (Identity Inference Attack) Given the anonymized genotype profile dataset r which contains the target's genotype record r_v, and a subset of the target's traits, $\mathbb{T}_S$, the attacker aims to learn the posteriori probability $P(r_i = r_v|\mathbb{T}_S)$ that the genotype record r_i corresponds to the target using the constructed bayesian network.

With the bayesian network constructed in the previous section, we can naturally acquire the probability that an individual has a specific allele type for an SNP given his/her associated trait information. Lemma 10 shows how to calculate the possibility that a record in the dataset belongs to the target given his specific traits. The proof is straightforward based on Theorem 7.

Lemma 10. For each genotype record, the probability that r_i belongs to the target v is

$$P(r_i = r_v|\mathbb{T}_S) = \frac{\prod_{j=1}^{|r_i|} P(r_{ij}|\mathbb{T}_{S_j})}{\sum_{i=1}^{|r|} \prod_{j=1}^{|r_i|} P(r_{ij}|\mathbb{T}_{S_j})} \tag{31}$$

where $\mathbb{T}_{S_j}$ denotes the parent trait nodes of S_j in the bayesian network. $P(r_{ij}|\mathbb{T}_{S_j})$ can be

acquired from the bayesian network.

Input: The genotype profile dataset $r = \{r_1, r_2, ..., r_n\}$ containing the target individual's genotype record(r_v), the trait set $\{T_1, T_2, ..., T_l\}$ that the target individual has, the GWAS catalog bayesian network G.
Output: The probability of each record in R belonging to the target individual $P(r_i = r_v)$.

1: FOR each trait T_k in set $\{T_1, T_2, ..., T_l\}$
2: Search G for T_k and obtain the associated SNPs $S_j (j \in [1, m])$ and the corresponding risk allele type;
3: ENDFOR
4: FOR each record r_i in r
5: Calculate the probability that r_i belongs to the target individual following Lemma 10.
6: ENDFOR

Algorithm 7: Identity Inference

Algorithm 7 describes a possible approach an attacker could take to identify the target individual's record in the dataset. Based on this approach, the attacker can also infer other private information of the target individual. For example, after deriving the probability that each record in the genotype dataset belongs to the target individual, the attacker can further derive any other trait that the target may have, based on the genotype information contained in the dataset. We formalize such new trait inference in Lemma 11.

Lemma 11. Assume that the genotype profile of the target, r_v, is contained in a genotype profile dataset r. The attacker has access to r where each record $r_i = (r_{i1}, r_{i2}, \cdots, r_{in})$ denotes the genotype profile of an individual i. The attacker also knows the target individual has a subset of traits, $\mathbb{T}_S$. The probability that the target has a new trait T_{new} can be derived as

$$\begin{aligned} P(T_{new}|r_v \in R, \mathbb{T}_S) &= \sum_{i=1}^{|r|} P(r_i = r_v) \times P(T_{new}|r_i) \\ &= \sum_{i=1}^{|r|} P(r_i = r_v|\mathbb{T}_S) \times P(T_{new}|r_i) \end{aligned} \tag{32}$$

where $P(T_{new}|r_i)$ can be derived following Lemma 9 and $P(r_i = r_v|\mathbb{T}_S)$ can be derived following Lemma 10.

5.5 Differentially Private GWAS Statistics

As introduced in previous chapters, differential privacy [32] is a paradigm of post-processing the output of queries and is agnostic to auxiliary information an adversary may possess, and provides guarantees against arbitrary attacks. In prior work on differential privacy, a database is treated as a collection of *rows*, with each row corresponding to the data of a different individual. Here we focus on how to compute GWAS statistics under differential privacy. The aim here is to ensure that the inclusion or exclusion of an individual in the GWAS dataset makes no statistical difference to the results found. We also study how differential privacy protection may prevent the proposed attack. We omit the formal definition of differential privacy and the classic Laplace mechanism to achieve differential privacy here, since they can be referred from chapter 2.

Differential privacy research has been significantly studied from the theoretical perspective, e.g., [11, 17, 53, 65, 71, 130]. The applicability of enforcing differential privacy in genomic data has been recently studied in [37, 61] where statistics (e.g., the allele frequencies of cases and controls, chi-square statistic and p-values) and logistic regression were explored on GWAS data.

We use $x = \{x_1, x_2, ..., x_{n_c+n_t}\}$ to denote a SNP data set that contains n_c cases and n_t controls. Each SNP profile x_i contains N SNPs. The purpose of a typical GWAS study is to discover K SNPs that are most significantly related with the trait under study. For each SNP, we can easily derive that the risk allele frequency in the case (control) group

f^c (f^t) has a global sensitivity of $\frac{1}{n_c}$ ($\frac{1}{n_t}$) where $n_c(n_t)$ is the number of individuals in the case (control) group. The sensitivity of various statistics used for statistical tests between a given SNP and the trait can also be derived straightforwardly. For example, the sensitivity values of chi-square statistic and p-value were derived in [37] and those sensitivity values are small. For those statistics with large sensitivity values (e.g., the sensitivity of odds ratio is infinity), we can use perturbed risk allele frequencies to indirectly calculate them.

One naive approach for differentially private releasing K most significant SNPs based on a given statistics Φ (e.g., chi-square statistic) is to add the Laplace noise $Lap(\frac{N}{\epsilon}GS_\Phi)$ to the true statistic value of each of N SNPs and then output K SNPs with most significant perturbed statistics values. The approach achieves ϵ differential privacy based on Theorem 2. However, this naive approach is infeasible in GWAS because the noise magnitude of $Lap(\frac{N}{\epsilon}GS_\Phi)$ is very large due to the large number of SNPs (N).

In [11], the authors developed an effective differential privacy preserving method on how release the most significant patterns together with their frequencies in the context of frequent pattern mining. The authors in [37] adapted this method to GWAS. We summarize their algorithm below.

1. Compute the sufficient statistics Φ for each of the N SNPs and perturb each real value with the Laplace noise of mean zero and magnitude of $Lap(\frac{4K}{\epsilon}GS_\Phi)$.

2. Pick K most relevant SNPs in terms of the noisy values and denote the corresponding set of SNPs by Γ.

3. Perturb the true Φ value of each SNP in Γ with the new Laplace noise with mean zero and magnitude of $Lap(\frac{2K}{\epsilon}GS_\Phi)$ and output Γ as well as their frequencies.

The algorithm still achieves ϵ differential privacy. However, the magnitude of the added noise is proportional to K rather than N. Note than the number of significant SNPs (K) is much smaller than the number of total SNPs (N). In our evaluation, we are not able to access the raw SNP data and the set of significant SNPs Γ is given in the released GWAS catalog. Thus we add the Laplace noise directly to the statistics of those SNPs Γ. In particular, for each significant SNP, we add the Laplace noise of mean zero and magnitude of $Lap(\frac{2K}{\epsilon n_c})$ ($Lap(\frac{2K}{\epsilon n_t})$) to the risk allele frequency in the case group f^c (in the control group f^t), and then use the perturbed frequencies to calculate the odds ratio. Recall that the risk allele frequency in the case (control) group f^c (f^t) has a global sensitivity of $\frac{1}{n_c}$ ($\frac{1}{n_t}$). The perturbed odds ratio values are used to construct the two-layered bayesian network. In the next section, we will show how the perturbed statistics affect the performance of both the trait inference attack and the identity inference attack.

5.6 Evaluation

We first discuss our evaluation setup in Section 5.6.1. We then evaluate the performance of our proposed trait inference attack in Section 5.6.2 and that of the identity inference attack in Section 5.6.3 using the directly released statistics in GWAS catalog. In Section 5.6.4, we evaluate how the differential privacy preserving GWAS statistics affect the performance of the above two attacks.

5.6.1 Experimental setup

We evaluate our methods using data extracted from the online NHGRI GWAS catalog [54] as of May 21, 2013. This version of the GWAS catalog includes 1,607 publications and 12,520 records about 10,133 SNPs associated with 834 traits. Publications included in such a catalog are limited to those attempted to assay at least 100,000 SNPs in the initial

stage. SNP-trait associations listed are limited to those with p-values less than 10^{-5}. Out of the 12,520 records, there are 9,461 records with exact odds ratio values and the remaining ones with odds ratio entry listed as 'pending' or 'NR'.

It is worth pointing out here that the *risk* allele type for a specific SNP is defined to indicate that this allele type appears more frequently in the case group than in the control group, therefore the corresponding odds ratio should be greater than 1. As described in the GWAS catalog website, odds ratio values less than 1 in the original publications are converted to odds ratio values greater than 1 for the alternate alleles. However, we identify that 4,469 records of SNP-trait associations with odds ratios less than 1 in this catalog; thus we treat them in our evaluation in the same way as claimed by the GWAS catalog, referring the alternate allele as the risk allele with an odds ratio converted to be greater than 1. In this chapter, we preprocess all of the 9,461 records with exact odds ratio values to build a knowledge database about traits and the associated *risk* SNP allele types. To evaluate the performance of the trait inference attack and the identity inference attack, we use the genotype profiles in the 1000 Genomes Project [104]. In our experiment, we extract two datasets. The first dataset, referred as 'CEU', consists of the 85 HapMap individuals from Utah residents with Northern and Western European ancestry (CEU) in the 1000 Genomes Project. The second dataset, referred as 'Random', consists of 85 randomly selected individuals from the 1,092 samples in the 1000 Genomes Project.

5.6.2 Trait Inference Attack

Table 19 shows the information and statistics of a snapshot of our constructed two-layer bayesian network from the GWAS catalog. There are 6 traits and 9 associated SNPs, which were reported from from six previous GWAS publications. For each SNP-trait pair, the risk

Table 19: Attack background information

Index	$Trait$	$S_j - s_j$	f_{kj}^t	O_{kj}	f_{kj}^c	$P(t_k)$
1	Chronic obstructive pulmonary disease	$rs9394152 - C$	0.41	22.22	0.9392	0.05
2		$rs73717741 - G$	0.07	11.9	0.4725	
3		$rs10928927 - C$	0.16	17.54	0.7696	
4	Drug-induced liver injury (flucloxacillin)	$rs2395029 - G$	0.05	45	0.703	0.00008
5	Jaw Osteonecrosis	$rs1934951 - T$	0.12	12.75	0.63	0.056
6	Osteoarthritis	$rs12982744 - C$	0.61	11.11	0.9456	0.036
7	Height(taller than 90% of population)	$rs12982744 - G$	0.4	33.33	0.9569	0.10
8		$rs7853377 - G$	0.23	50.0	0.9372	
9		$rs7567288 - C$	0.2	33.33	0.8929	
10	Eye color(Green)	$rs12913832 - A$	0.23	8.43	0.7158	0.16

allele type, risk allele in the control group and the odds ratio are shown in Columns 3-5. We calculate the risk allele frequency in the case group for each SNP-trait pair and show the result in Column 6. Note that there is a big gap (around 0.5) between the risk allele frequency in the case group and that in the control group. We also acquire from the original studies or Wikipedia the prior probability (prevalence) of each trait, $P(t_k)$, and show the result in Column 7.

Note that the SNP, $rs12982744$, is related to two traits with row indices of 6 and 7 in Table 19. Since it is unavailable of the detailed probability that the alleles of this SNP appear in people who have these two traits, we follow Lemma 7 to calculate the conditional probability table. The prior probabilities that alleles C and G appear in the population, are 0.79 and 0.21 respectively.

With the constructed bayesian network, the attacker can then run the trait inference attack(Algorithm 6) to calculate the posterior probability that the target individual has a trait given his/her genotype profile.

Table 20 shows the estimation results calculated from Lemma 9. Each row in Table 20 corresponds to the row with the same index in Table 19. Column $\overline{P}(t_k|r_{ij})$ gives the av-

Table 20: Posterior probability of certain trait considering one SNP

Index	n_1	$P(T\|r_{ij} = s_j)$	n_0	$P(T\|r_{ij} = \bar{s}_j)$	$\overline{P}(t_k\|r_{ij})$
1	58	0.1076	27	0.0054	0.0751
2	15	0.2621	70	0.0290	0.0701
3	20	0.2020	65	0.0142	0.0584
4	10	0.0011	75	$2.5E-5$	$1.5E-4$
5	27	0.2389	58	0.0240	0.0923
6	28	0.0546	57	0.0203	0.0316
7	57	0.1744	28	0.0078	0.1195
8	6	0.3117	79	0.0100	0.0313
9	3	0.3316	82	0.0175	0.0286
10	37	0.3721	48	0.0657	0.1991

erage probability that the 85 CEU participants from the 1000 Genomes Project has each trait. We can see that most of the average probabilities (with bold font) are higher than the corresponding prior probability of having a trait. Columns n_1 and n_0 respectively represent the number of individuals who have and do not have the risk allele type listed in the corresponding row of Table 19. Columns $P(t_k|r_{ij} = s_j)$ and $P(t_k|r_{ij} = \bar{s}_j)$ respectively represent the the posterior probability of one individual has a trait if he/she has the risk allele type, or does not have the risk allele type of one specific SNP corresponding to the trait.

5.6.3 Identity Inference Attack

For the identity inference attack, we use both 'CEU' and 'Random' datasets. In our experiment, we assume that the target has the trait of green eyes, top 10% height, and all other traits listed in Table 19. That is to say, the trait set for the target has six elements.

We randomly generate the genotype record for the target individual. The generating strategy is that for each SNP S_j associated with one trait T_k, we generate $r_{ij} = s_j$ with the probability $P(s_j|t_k)$. Particularly, SNP $rs12982744$ is associated with both osteoarthritis and height where the allele C is risky for osteoarthritis and allele G is positively associated

with height. We refer to allele C as s_j when considering them together, and then generate the target individual's allele type $r_{ij} = s_j$ with the probability calculated following Lemma 7. Next we blend the generated record into the 'CEU' dataset (containing the genotype records of the 85 CEU individuals) and the 'Random' dataset(containing the randomly selected 85 records) respectively. Finally, we calculate the probability that the generated record is correctly identified as belonging to the target individual, given the background trait information. We also compare the inference capability with different amount of background knowledge, i.e., with the size of trait set ranging from one to six.

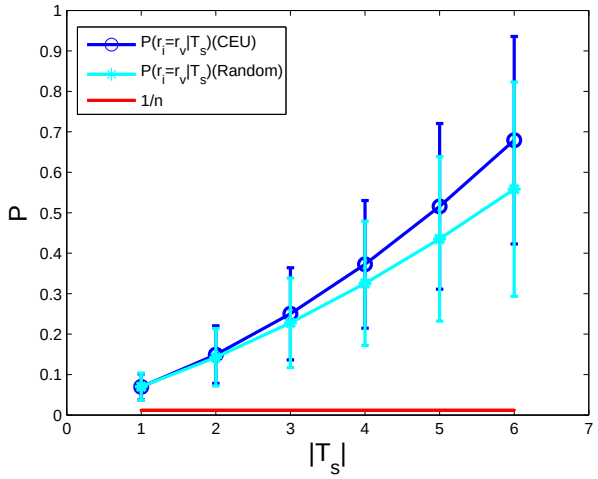


Figure 11: Average probability of identity inference attack with different amount of background knowledge

We run this whole process for 10,000 times for each dataset. Figure 11 shows the average value of the resulted probabilities. As shown in Figure 11, the red line is the baseline representing the probability (1/86) that the generated record is inferred as belonging to the target individual without any background knowledge, whereas the blue (cyan) line represents the resulted probability on 'CEU' ('Random') dataset. The first point in the blue (cyan) line

represents the average probability that the generated record is correctly referred given any one of the six traits on ‘CEU’ (‘Random’) dataset. Similarly, the second point represents the average probability that the generated record is correctly referred given any two traits from all the six traits, and so on. We can see that in general, for both the ‘CEU’ and ‘Random’ datasets, the probability of correctly identifying the target individual increases as the background knowledge increases, while the inference probability given only one trait is much larger than that of the situation without any background knowledge. The bar at each point shows the standard deviation of the resulting probabilities of 10,000 times of test.

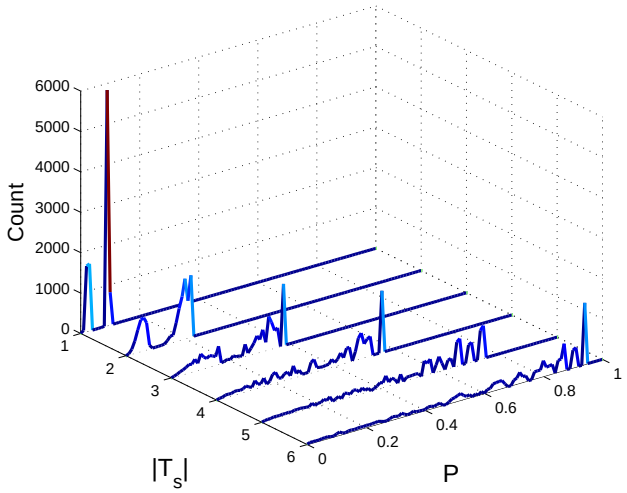


Figure 12: Probability distribution of identity inference attack on ‘CEU’ dataset with different amount of background knowledge

Figure 12 shows the distribution of the inference probability among the 10,000 times of identity inference test for the ‘CEU’ dataset and Figure 13 shows the similar distribution for the ‘Random’ dataset. As the amount of background traits increases, the peaks of the process count would be located at positions with larger identifying probabilities. This indicates that in general, the more background knowledge we have, the more probably that

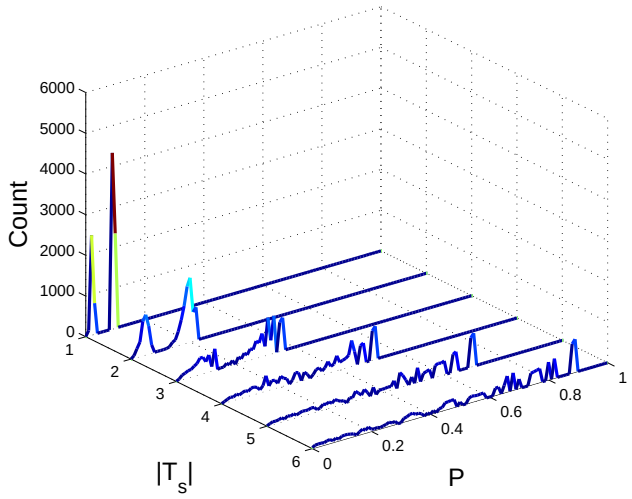


Figure 13: Probability distribution of identity inference attack on 'Random' dataset with different amount of background knowledge

the target individual's record is correctly identified. Specifically, the highest peak in the wave of $|T_s| = 6$ locates near the line of $P = 1$, which represents that when knowing all six traits, the attacker could successfully identify the target individual with a confidence of more than 90% in most times of test. On the other hand, multiple peaks in each line represent the different identifying probabilities due to the different combinations of background traits as well as the different possible genotype records being randomly generated.

We also notice that the overall inference probability on the 'Random' dataset is slightly lower than that on the 'CEU' dataset. This is due to the different risk allele frequencies between the 'CEU' dataset and the 'Random' dataset. As in 'CEU' dataset, we have known more specifics about the population background, while 'ALL' dataset includes individuals from other populations and the population pool is larger. Figure 14 shows the risk allele frequencies for each of the nine involved SNPs between the 'CEU' dataset and the whole dataset (referred as 'ALL') of 1092 individuals from the 1000 Genome Project.The blue

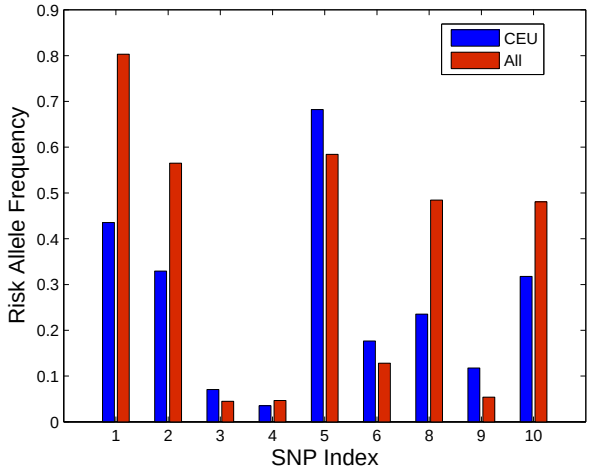


Figure 14: Risk allele frequency in the subset of 85 CEU individuals and in whole dataset of 1092 individuals

(red) bars represent the risk allele frequencies in the 'CEU' ('ALL') dataset. The SNP index corresponds to the index in Table 19. From Figure 14, we can see that for four out of the nine associated SNPs, the risk allele frequencies in the whole dataset of 1092 individuals are much higher than those in the subset of 'CEU' records. For other SNPs, there is no remarkable difference of the risk allele frequency in the two datasets. Thus in the 'Random' dataset, some risk alleles are more frequently occurred than that in 'CEU' dataset, making the probability of inferring the target individual slightly lower. However, the results demonstrate that even in a dataset with no race information available, the attacker can still correctly infer the target individual with high confidence by exploiting known traits of the target.

5.6.4 Attack using differentially private GWAS statistics

We conduct experiments to evaluate how the trait inference attack and the identity inference attack work based on the bayesian network constructed from the differentially private

Table 21: Differential private posterior probability of certain trait considering one SNP

Index	$P(t_k)$	$\overline{P}(t_k\|r_{ij})$	$\overline{P}(t_k\|r_{ij})(\epsilon = 2)$	$\overline{P}(t_k\|r_{ij})(\epsilon = 0.2)$
1		0.0751	0.0749	0.0749
2	0.05	0.0701	0.0670	0.0679
3		0.0584	0.0581	0.0571
4	8E-5	$1.54E-4$	$1.59E-4$	$1.49E-4$
5	0.056	0.0923	0.0934	0.2637
6	0.036	0.023	0.023	0.023
7		0.2031	0.2054	0.2055
8	0.10	0.0303	0.0360	0.0360
9		0.0258	0.0300	0.0301
10	0.16	0.1991	0.1992	0.1986

statistics. Our evaluation is still based on the 85 CEU participants from the 1000 Genomes Project. In our experiments, we choose two privacy threshold values, $\epsilon = 2$ and $\epsilon = 0.2$, which represent two settings for reasonable privacy preservation in GWAS. For each ϵ, we follow the procedure in Section 5.5 to derive the differential privacy preserving statistics and then construct the bayesian network.

Table 21 shows the comparison of the trait inference attack. Each row in Table 21 corresponds to one trait-SNP pair with the same index in Table 19. Column $P(t_k)$ shows the prevalence of the trait in population. Columns $\overline{P}(t_k|r_{ij})$, $\overline{P}(t_k|r_{ij})(\epsilon = 2)$, and $\overline{P}(t_k|r_{ij})(\epsilon = 0.2)$ show the average probability that the 85 CEU participants from the 1000 Genomes Project has each trait under three compared scenarios, using directly released GWAS statistics, 2-differentially private statistics, and 0.2-differentially private statistics, respectively. We can see that most of the average probabilities (with bold font) are higher than the corresponding prior probability of having a trait. We are interested in how the derived posterior probabilities using perturbed statistics are different from those using the original statistics. We define the average absolute relative error as $\gamma(\epsilon) = \frac{1}{K}\sum_{j=1}^{K}\frac{|\overline{P}(t_k|r_{ij}) - \overline{P}_\epsilon(t_k|r_{ij})|}{\overline{P}(t_k|r_{ij})}$. Our results show $\gamma(2) = 0.0408$ and $\gamma(0.2) = 0.2282$, which indicate the more rigorous privacy

Table 22: Average probability of identity inference attack with different amount of background knowledge

$\lvert\mathbb{T}_S\rvert$	$\overline{P}(r_i = r_v \lvert \mathbb{T}_S)$					
	Original		$\epsilon = 2$		$\epsilon = 0.2$	
	ave	std	ave	std	ave	std
1	0.0697	0.0321	0.0645	0.0275	0.0325	0.0075
2	0.1493	0.0312	0.1320	0.0576	0.0646	0.0229
3	0.2503	0.1138	0.2118	0.0916	0.0978	0.0497
4	0.3725	0.1578	0.3032	0.1348	0.1230	0.0923
5	0.5158	0.2047	0.4079	0.1911	0.1360	0.1484
6	0.6792	0.2565	0.5323	0.2657	0.1340	0.2200

protection incurs more loss of attack performance in terms of accuracy.

We also use the differentially private statistics to run the identity inference attack again on 'CEU' dataset. In Table 22, each row corresponds to some certain number of traits the target individual has. The columns under label 'Original', '$\epsilon = 2$' and '$\epsilon = 0.2$' denote the average probability of correctly identifying the target individual $\overline{P}(r_i = r_v|\mathbb{T}_S)$ calculated with original value of GWAS statistics, the 2-differentially private GWAS statistics, and the 0.2-differentially private GWAS statistics respectively. For each scenario, we use 'ave' and 'std' to denote the mean and the standard deviation. We can easily observe that the average probability of correctly identifying the target individual $\overline{P}(r_i = r_v|\mathbb{T}_S)$ increases as the number of known traits increases under three scenarios. This observation shows that the more background knowledge the attacker has, the more likely the target individual can be identified. We are interested in how the performance of the identity inference attack is affected by the perturbed GWAS statistics. We can see that the attack performance is significantly decreased when GWAS statistics are distorted under rigorous privacy protection. For example, as the last row shows, when $|\mathbb{T}_S| = 6$, the accuracy of the attack decreases from 0.6792 to 0.5323 ($\epsilon = 2$), and further to 0.1340 ($\epsilon = 0.2$). However, the probability

(0.1340) that the target individual being correctly identified under $\epsilon = 0.2$ is still an order high than the probability of random guess (0.0116).

5.7 Summary

In summary, we studied whether and to what extent GWAS statistics can be exploited by an attacker to learn private information of general population, not limited to GWAS participants. We developed two potential attacks, *trait inference attack* and *identity inference attack*. Both attacks exploit the released GWAS statistics about the associations between SNP genotypes and human traits. Our evaluations showed that the proposed attacks (even with perturbed GWAS statistics under differential privacy) have made re-identification of anonymized genotype data a real threat. In this work and our early conference version [109], we were focused on a different and new privacy protection problem not covered by previous research, i.e., whether and to what extent GWAS statistics can be exploited by an attacker to learn private traits of regular people rather than those GWAS participants. The technical report version of this work can be found in [110].

CHAPTER 6: REGRESSION MODEL FITTING UNDER DIFFERENTIAL PRIVACY AND MODEL INVERSION ATTACK

6.1 Introduction

In many applications, sensitive datasets such as financial transactions, medical records, or genetic information about individuals are often only disclosed to authorized users, yet the models learned from them are made public. The released models may be exploited by an adversary to breach privacy of both participant individuals in the datasets and regular non-participant individuals.

In [38], the authors developed a model inversion attack where an adversary uses the released model to make predictions of sensitive attributes (used as input to the model) of a target individual when some background information about the target individual is available. The authors showed that differential privacy mechanisms prevent model inversion attacks only when the privacy budget is very small. However, for privacy budgets effective at preventing attacks, the utility in terms of performing simulated clinical trials is significantly lost.

Hence it is imperative to develop mechanisms to achieve differential privacy protection for participants and prevent model inversion attacks while retaining the utility of the released models. In this chapter, we focus on regression models which have been widely applied in many applications. Regression studies often involve continuous data (e.g., blood lipid levels or heights) in addition to categorical attributes (e.g., gender, race and disease).

Various regression models including linear regression, logistic regression, and lasso models have been developed. There are generally two approaches to derive differential privacy preserving regression models. The first approach is to directly perturb the output coefficients of the regression models. However, this approach requires an explicit sensitivity analysis of the regression models, which is often infeasible. The second approach, called the functional mechanism, is to add noise to the coefficients of the objective function used to derive regression models [17, 131]. Deriving a bound on the amount of noise needed for the functional mechanism involves a fairly simple calculation on the object function.

Differential privacy preserving regression models based on the functional mechanism [131] guarantee protection against attempts to infer whether a subject was included in the training set used to derive a model. It is not effective to protect attribute privacy, which is the target of the model inversion attacks. This is because in the functional mechanism regression coefficients corresponding to different attributes are perturbed equally. However, model inversion attacks seek to exploit correlation between the target sensitive attributes, known non-sensitive attributes and the model output. In this chapter, we will develop a new approach to learn differential privacy preserving regression models which effectively prevent model inversion attacks and retain the model utility. Our approach leverages the functional mechanism but effectively balances the privacy budget for sensitive and non-sensitive attributes in learning the differential privacy preserving regression models.

6.1.1 Problem Formalization

Let D be a data set that contains n tuples $t_1, t_2, ..., t_n$ regarding d explanatory attributes $X_1, X_2, \cdots, X_d$ and one response attribute Y. The explanatory attributes can be divided into two groups: non-sensitive attributes and sensitive attributes. For simplicity, we con-

Table 23: Notations used in Chapter 6

Symbol	Definition
$t_i = (\mathbf{x}_i, y_i)$	the i-th tuple
ω	the parameter vector of the regression model
$\rho(\omega)$	the released regression mode
$f(t_i, \omega)$	the cost function on tuple t_i
$f_D(\omega)$	$f_D(\omega) = \sum_{t_i \in D} f(t_i, \omega)$
ω^*	$\omega^* = \arg\min_\omega f_D(\omega)$
$\phi(\omega)$	a product of one or more elements in $\omega_1, \omega_2, ..., \omega_d$
Φ_j	the set of all products of all possible ϕ of order j
$\lambda_{\phi t_i}$	the polynomial coefficient of ϕ in $f(t_i, \omega)$
$\epsilon, \epsilon_s, \epsilon_n$	privacy for all, sensitive, non-sensitive attributes

sider there is only one sensitive attribute X_s and all remaining ones are non-sensitive. Our analysis can be straightforwardly extended to multiple sensitive attributes. For each explanatory attribute X_i, without loss of generality, we assume its domain $\mathcal{X}_i$ in the range of $[-1, 1]$. Similarly, we denote $\mathcal{Y}$ as the domain of the response attribute Y, which could be $[-1, 1]$ for linear regression or $\{0, 1\}$ for logistic regression. We denote each tuple t_i as $(\mathbf{x}_i, y_i)$ where $\mathbf{x}_i = (x_{i1}, x_{i2}, ..., x_{id})$. Throughout this chapter, we use bold lower-case variables, e.g., $\mathbf{x}_i$, to represent vectors; upper-case alphabets, e.g., X_i, to denote an attribute; calligraphic upper-case alphabets, e.g., $\mathcal{X}_i$, to denote the domain of attribute X_i. $\mathbf{x}^T$ refers the transpose of vector $\mathbf{x}$. Table 23 summarizes the notation used in this chapter.

The data mining task is to release a regression model from D to predict the attribute value of Y given the corresponding attribute value of $X_1, ..., X_d$. That is to say, we are to release a function ρ parameterized with a real number vector $\omega = (\omega_1, ..., \omega_d)$. The model takes $\mathbf{x}_i$ as input and output the corresponding prediction for y_i as $\hat{y}_i = \rho(\mathbf{x}_i, \omega)$. Most regression analytical methods often iteratively optimize some objective functions with various constraints. A cost function f is often chosen to measure the difference between the original and predicted values based on specific ω. The optimal model parameter ω^* is

defined as the one that minimizes the loss function.

$$\omega^* = \arg\min_{\omega} f_D(\omega) = \arg\min_{\omega} \sum_{i=1}^{n} f(t_i, \omega). \tag{33}$$

In this chapter, we consider two commonly used regression models, linear regression and logistic regression.

Definition 13. (Linear Regression) Assume without loss of generality that Y has a domain of $[-1, 1]$. A linear regression on D returns a prediction function $\hat{y}_i = \rho(\mathbf{x}_i, \omega^*) = \mathbf{x}_i^T \omega^*$, where ω^* is a d-dimensional real vector that minimizes the following cost function.

$$\omega^* = \arg\min_{\omega} f_D(\omega) = \arg\min_{\omega} \sum_{i=1}^{n} (y_i - \mathbf{x}_i^T \omega)^2. \tag{34}$$

Definition 14. (Logistic Regression) Assume Y has a domain of $\{0, 1\}$. A logistic regression on D returns a prediction function which returns $\hat{y}_i = 1$ with the probability $P(\hat{y}_i = 1 | \mathbf{x}_i, \omega^*) = exp(\mathbf{x}_i^T \omega^*)/(1 + exp(\mathbf{x}_i^T \omega^*))$, where ω^* is a d-dimensional real vector that minimizes the following cost function.

$$\begin{aligned} \omega^* &= \arg\min_{\omega} f_D(\omega) \\ &= \arg\min_{\omega} \sum_{i=1}^{n} (log(1 + exp(\mathbf{x}_i^T \omega^*)) - y_i \mathbf{x}_i^T \omega). \end{aligned} \tag{35}$$

Releasing the regression model under differential privacy requires noise injection to the model parameter ω^*. Adding noise to ω^* involves the derivation of the sensitivity of ω^*, which is rather challenging. In this chapter, we apply the *functional mechanism* proposed in [131], which perturbs the objective function of the regression models. However, the release model parameter ω^* or its perturbed one $\bar{\omega}$ could be exploited by the adversary to predict the value of sensitive input attributes $x_{\alpha s}$ for a target individual α when some

background information about the target individual is available. Formally, the adversary has access to the regression model with parameters ω^*, the domain value and marginal probability of the model attributes, accuracy metrics of the model like the confusion matrix, in addition to some background knowledge of the target including the value of a subset of those non-sensitive input attributes and the value of output attribute of the model y_α.

Our research problem is how to derive the perturbed regression model parameter $\bar{\omega}$ such that we achieve differential privacy protection for participants and prevent model inversion attacks on regular individuals while retaining the utility of the regression model.

6.2 Background

We omit the formal definition of differential privacy and the classic Laplace mechanism to achieve differential privacy here, since they can be referred from chapter 2.

6.2.1 Model Inversion

Model inversion attack [38] leverages the released regression model $y = \rho(\mathbf{x}, \omega^*)$ trained from a dataset D which contains a sensitive attribute X_s. An adversary could exploit the released model to predict the sensitive input attribute value of the target individual based on some of the target individual's background (values of some non-sensitive input attributes, e.g. demographic information, for the model) and the observed response attribute value. Such attack seeks to take advantage of correlation between the target sensitive attribute, observed non-sensitive attributes and the model output.

The model inversion attack algorithm works as follows. The adversary has access to the regression model ρ with parameter ω^* trained over a dataset D drawn i.i.d from an unknown prior distribution p. Recall that D has input domain $\mathcal{X}_1 \times \cdots \times \mathcal{X}_d$ and output domain $\mathcal{Y}$. The target individual is represented by $t_\alpha = (\mathbf{x}_\alpha, y_\alpha)$. The adversary is assumed

to know values of some (or all) input attributes of the target individual except the sensitive one, i.e., $S \subseteq X \backslash X_s$, and the output value y_α. The sensitive attribute value the adversary wants to learn is referred to as $x_{\alpha s}$. Note that the target individual t_α is not necessary to be in D. In addition to the released model $\rho(\mathbf{x}, \omega^*)$, the adversary also has access to marginal $p_1, ..., p_d, p_y$ of the joint prior p, the input domain and the output domain, the information π about the model prediction performance where $\pi(y, y') = Pr(y_i = y | \rho(x_i, \omega^*) = y')$. The algorithm makes prediction by estimating the probability of a potential target attribute value given the available information of the target individual and the model.

- Find the feasible set $\hat{\mathcal{X}} \subseteq \mathcal{X}$, i.e., for $\forall x \in \hat{\mathcal{X}}$, x matches $\mathbf{x}_\alpha$ on each known attribute in S.

- if $\hat{\mathcal{X}} = 0$, return null; otherwise, return $\hat{x}_{\alpha s} = z$ that maximizes $\sum_{x \in \mathcal{X} : x_s = z} \pi_{y_\alpha, \rho(x_\alpha, \omega)} \prod_{1 \leq j \leq d} p_j(x_j)$.

In step 1, the algorithm filters the domain space using the known attribute values of the target individual. In step 2, the algorithm calculates weight to each candidate row in the domain space based on known priors and how well the model's output on that row coincides with the target individual's model output value. It then returns the value of the target sensitive attribute with the largest weight computed by marginalizing the other attributes. The model inversion algorithm is optimal as it minimizes the expected misclassification rate on the maximum-entropy prior given the model and marginals. It was demonstrated in [38] that the value of the sensitive attribute is predicted with significantly better accuracy than guessing based on marginal distributions.

6.3 Our Approach

It is concluded in [38] that differential privacy mechanisms can prevent model inversion attacks only when privacy budget is very small, but in those cases, the private model usually does not simultaneously retain desirable efficacy. In clinical trials, such lack of efficacy may put patients in increased risk of health problems.

To tackle such problem, in this section, we propose a new approach to provide regression models under differential privacy and model inversion attacks. Our approach aims to improve privacy specifically for sensitive attributes while retaining the efficacy of the released regression model by balancing the privacy budget for sensitive and non-sensitive attributes. Our approach leverages the *functional mechanism* proposed in [131] but perturbs the coefficients of the objective function with different magnitudes of noise.

6.3.1 Functional Mechanism Revisited

Functional mechanism achieves ϵ-differential privacy by perturbing the objective function $f_D(\omega)$ and then releasing the model parameter $\bar{\omega}$ that minimizes the perturbed objective function $\bar{f}_D(\omega)$ instead of the original one. Because $f_D(\omega)$ is a complicated function of ω, the functional mechanism exploits the polynomial representation of $f_D(\omega)$.

The model parameter ω is a vector that contains d values $\omega_1, \omega_2, ..., \omega_d$. Let $\phi(\omega)$ denote a product of $\omega_1, \omega_2, ..., \omega_d$, namely, $\phi(\omega) = \omega_1^{c_1} \cdot \omega_2^{c_2} ... \omega_d^{c_d}$ for some $c_1, c_2, ..., c_d \in N$. Let $\Phi_j (j \in N)$ denote the set of all products of $\omega_1, \omega_2, ..., \omega_d$ with degree j, i.e.,

$$\Phi_j = \{\omega_1^{c_1} \cdot \omega_2^{c_2} ... \omega_d^{c_d} | \sum_{l=1}^{d} c_l = j\}. \tag{36}$$

By the Stone-Weierstrass Theorem, any continuous and differentiable $f(t_i, \omega)$ can al-

ways be written as a polynomial of $\omega_1, \omega_2, ..., \omega_d$, i.e., for some $J \in [0, \infty]$, we have

$$f(t_i, \omega) = \sum_{j=0}^{J} \sum_{\phi \in \Phi_j} \lambda_{\phi t_i} \phi(\omega), \tag{37}$$

where $\lambda_{\phi t_i} \in R$ denotes the coefficient of $\phi(\omega)$ in the polynomial. Similarly, $f_D(\omega)$ can also be expressed as a polynomial of $\omega_1, ..., \omega_d$.

For example, the expansion of the linear regression is as follows.

$$\begin{aligned} f_D(\omega) &= \sum_{t_i \in D} (y_i - \mathbf{x}_i^T \omega)^2 \\ &= \sum_{t_i \in D} y_i^2 - \sum_{j=1}^{d} (2 \sum_{t_i \in D} y_i x_{ij}) \omega_j \\ &+ \sum_{1 \le j,l \le d} (\sum_{t_i \in D} x_{ij} x_{il}) \omega_j \omega_l \end{aligned} \tag{38}$$

We can see that $f_D(\omega)$ only involves monomials in $\Phi_0 = \{1\}$ (corresponding to $\lambda_{\phi t_i} = y_i^2$), $\Phi_1 = \{\omega_1, \omega_2, ..., \omega_d\}$ (for a specific ω_j, $\lambda_{\phi_{t_i}} = 2 y_i x_{ij}$), and $\Phi_2 = \{\omega_i \omega_j | i, j \in [1, d]\}$.

$f_D(\omega)$ is perturbed by injecting Laplace noise into its polynomial coefficients $\lambda_{\phi t_i}$, and then the model parameter $\bar{\omega}$ is derived to minimize the perturbed function $\bar{f}_D(\omega)$. Each polynomial coefficient $\lambda_{\phi t_i}$ is perturbed by adding Laplace noise $Lap(\frac{\Delta}{\epsilon})$, where $\Delta = 2 max_t \sum_{j=1}^{J} \sum_{\phi \in \Phi_j} ||\lambda_{\phi t}||_1$, according to the following Lemma 12.

Lemma 12. [131] Let D and D' be any two neighboring datasets. Let $f_D(\omega)$ and $f_{D'}(\omega)$ be the objective functions of regression analysis on D and D', respectively, and denote their polynomial representations as follows:

$$f_D(\omega) = \sum_{j=1}^{J} \sum_{\phi \in \Phi_j} \sum_{t_i \in D} \lambda_{\phi t_i} \phi(\omega),$$

$$f_{D'}(\omega) = \sum_{j=1}^{J} \sum_{\phi \in \Phi_j} \sum_{t'_i \in D'} \lambda_{\phi t'_i} \phi(\omega).$$

Then, we have the following inequality

$$\sum_{j=1}^{J} \sum_{\phi \in \Phi_j} || \sum_{t_i \in D} \lambda_{\phi t_i} - \sum_{t'_i \in D'} \lambda_{\phi t'_i} ||_1 \leq 2max_t \sum_{j=1}^{J} \sum_{\phi \in \Phi_j} ||\lambda_{\phi t}||_1. \tag{39}$$

where t_i, t'_i or t is an arbitrary tuple.

When the polynomial form of an objective function (e.g., logistic regression objective function) contains terms with unbounded degrees, [131] develops an approximation polynomial form based on Taylor expansion. Functional mechanism is effective as it does not need to derive the sensitivity of ω, which is rather challenging. The perturbation method based on the functional mechanism [131] also removes the requirement, i.e., the convexity of the objective function, from the original function perturbation approach [17].

6.3.2 Improved Perturbation of Objective Function

To better optimize the balancing between privacy and the regression model' efficacy, we propose a new algorithm based on the functional mechanism to improve privacy specifically for sensitive attributes. To improve the privacy on X_s, we aim to weaken the correlation between X_s and the model output Y by perturbing the corresponding ω_s more intensely. In other words, we need to add noise with larger magnitude to the coefficients of the monomials involving ω_s and add noise with smaller magnitude to the other coefficients. As a result, we expect to retain the utility of the released regression model while achieving differential privacy for participants and preventing model inversion attacks.

In general, a database can contain more than one sensitive attributes. We allocate privacy budget ϵ_n for non-sensitive attributes and ϵ_s for sensitive ones. We introduce a ratio

parameter, γ such that $\epsilon_s = \gamma\epsilon_n$ and $0 < \gamma \leq 1$. The smaller the γ, the more noise added to the sensitive attributes.

Input: Database D, objective function $f_D(\omega)$, privacy threshold ϵ, privacy budget ratio γ
Output: $\bar{\omega}$

1: Set $\Phi_n = \{\}, \Phi_s = \{\}$;
2: FOR each $1 \leq j \leq J$
3: FOR each $\phi \in \Phi_j$
4: IF ϕ does not contain ω_s from any sensitive attribute
5: Add ϕ into Φ_n;
6: ELSE
7: Add ϕ into Φ_s;
8: ENDIF
9: ENDFOR
10: ENDFOR
11: Set $\Delta = 2max_t \sum_{j=1}^{J} \sum_{\phi \in \Phi_j} ||\lambda_{\phi t}||_1$;
12: Set $\beta_1 = 2max_t \sum_{\phi \in \Phi_n} ||\lambda_{\phi t}||_1 / \Delta$;
13: Set $\beta_2 = 2max_t \sum_{\phi \in \Phi_s} ||\lambda_{\phi t}||_1 / \Delta$; $(\beta_1 + \beta_2 = 1)$
14: Set $\epsilon_n = \frac{1}{\beta_1 + \gamma\beta_2}\epsilon$, $\epsilon_s = \frac{\gamma}{\beta_1 + \gamma\beta_2}\epsilon$;
15: FOR each $1 \leq j \leq J$
16: FOR each $\phi \in \Phi_j$
17: IF $\phi \in \Phi_n$
18: set $\lambda_\phi = \sum_{t_i \in D} \lambda_{\phi t_i} + Lap(\frac{\Delta}{\epsilon_n})$;
19: ELSE
20: set $\lambda_\phi = \sum_{t_i \in D} \lambda_{\phi t_i} + Lap(\frac{\Delta}{\epsilon_s})$;
21: ENDIF
22: ENDFOR
23: ENDFOR
24: Let $\bar{f}_D(\omega) = \sum_{j=1}^{J} \sum_{\phi \in \Phi_j} \lambda_\phi \phi(\omega)$;
25: Compute $\bar{\omega} = argmin_\omega \bar{f}_D(\omega)$;
26: RETURN $\bar{\omega}$;

Algorithm 8: Functional Mechanism with Different Perturbation of Coefficients

Specifically, we divide all ϕs into two subsets Φ_n and Φ_s based on whether they involve any sensitive attribute, as shown in Lines 1-10 of Algorithm 8. Secondly, we determine the privacy budget according to the given ϵ and the privacy budget ratio γ. In Line 11, we set Δ based on the maximum length of all the coefficients $\lambda_{\phi t}$ of $\phi(\omega)$ in the polynomial. Accordingly, β_1 and β_2 can be considered as the fraction of contributions to Δ from coef-

ficients corresponding to elements in Φ_n and that in Φ_s. We will derive formula of Δ, β_1 and β_2 for linear regression and logistic regression and further prove they will not disclose any private information about dataset D in Results 7 and 8 respectively. Thirdly, we add noise to polynomial coefficients of $\phi \in \Phi_n$ with ϵ_n and to those of $\phi \in \Phi_s$ with ϵ_s, to derive the differentially private objective function $\bar{f}_D(\omega)$. Finally, we calculate and output the optimized ω^* according to $\bar{f}_D(\omega)$. Overall, our algorithm can achieve ϵ-differential privacy.

Theorem 8. Algorithm 8 satisfies ϵ-differential privacy.

Proof. Assume D and D' are two neighbouring datasets. Without loss of generality, D and D' differ in the last row t_n and $t_n^{'}$. Δ is calculated as Line 1 of Algorithm 8, and $\bar{f}(\omega)$ is the output of Line 24. Φ_s (Φ_n) denotes the set of ϕ that does (does not) contain sensitive attribute ω_s. We denote the coefficient in front of each ϕ into a vector. The maximum difference of the objective function on D and D' is then the maximum difference of such vector introduced by t_n and $t_n^{'}$, which is the sensitivity of such coefficient vector. Adding Laplace noise to this coefficient vector would produce the differentially private objective function. Specifically, we can add different magnitudes of noise to the vector entries corresponding to $\phi \in \Phi_n$ and those corresponding to $\phi \in \Phi_s$. γ is pre-determined as the ratio of such difference of noise magnitude. Formally, we have

$$\begin{aligned} Pr(\bar{f}(\omega|D)) = \prod_{\phi \in \Phi_n} exp(\frac{\epsilon_n || \sum\limits_{t_i \in D} \lambda_{\phi t_i} - \lambda_\phi ||_1}{\Delta}) \\ \prod_{\phi \in \Phi_s} exp(\frac{\epsilon_s || \sum\limits_{t_i \in D} \lambda_{\phi t_i} - \lambda_\phi ||_1}{\Delta}) \end{aligned} \tag{40}$$

Similarly, we have the formula for $Pr(\bar{f}(\omega|D'))$.

$$
\begin{aligned}
&\frac{Pr(\bar{f}(\omega|D))}{Pr(\bar{f}(\omega|D'))} \\
&\leq \prod_{\phi\in\Phi_n} exp(\frac{\epsilon_n}{\Delta}||\sum_{t_i\in D}\lambda_{\phi t_i} - \sum_{t'_i\in D'}\lambda_{\phi t'_i}||_1) \\
&\qquad \prod_{\phi\in\Phi_s} exp(\frac{\epsilon_s}{\Delta}||\sum_{t_i\in D}\lambda_{\phi t_i} - \sum_{t'_i\in D'}\lambda_{\phi t'_i}||_1) \\
&= \prod_{\phi\in\Phi_n} exp(\frac{\epsilon_n}{\Delta}||\lambda_{\phi t_n} - \lambda_{\phi t'_n}||_1) \prod_{\phi\in\Phi_s} exp(\frac{\epsilon_s}{\Delta}||\lambda_{\phi t_n} - \lambda_{\phi t'_n}||_1) \\
&\leq \prod_{\phi\in\Phi_n} exp(\frac{\epsilon_n}{\Delta}2\max_t||\lambda_{\phi t}||_1) \prod_{\phi\in\Phi_s} exp(\frac{\epsilon_s}{\Delta}2\max_t||\lambda_{\phi t}||_1) \\
&= exp(\epsilon_n\beta_1 + \epsilon_s\beta_2) \\
&= exp(\frac{\beta_1}{\beta_1+\gamma\beta_2}\epsilon + \frac{\gamma\beta_2}{\beta_1+\gamma\beta_2}\epsilon) = exp(\epsilon)
\end{aligned}
\tag{41}
$$

□

Our algorithm needs to derive Δ, β_1 and β_2 to add noise with different magnitudes to the polynomial coefficients of sensitive attributes and non-sensitive attributes. Result 7 shows their derived formulas for linear regression and Result 7 shows for logistic regression. We can see they only involve the number of attributes d and the number of sensitive attributes k. As a result, they do not disclose any private information of the dataset D, which guarantees the rigorous ϵ-differential privacy. Due to space limitations, we only give the proof for linear regression in Result 7 and skip the proof for logistic regression in Result 8.

Result 7. For linear regression defined in Definition 13, assume there are k sensitive attributes among the total d input attributes. We have in Algorithm 8, $\Delta = 2(d^2 + 2d)$; $\beta_1 = \frac{d-k}{d}$ and $\beta_2 = \frac{k}{d}$.

Proof. According to Equation 38, we have

$$\begin{aligned}\Delta &= 2\max_{t=(x,y)}\sum_{j=1}^{J}\sum_{\phi\in\Phi_j}||\lambda_{\phi t}||_1 \\ &\leq 2\max_{t=(x,y)}(2\sum_{j=1}^{d}yx_{(j)}+\sum_{1\leq j,l\leq d}x_{(j)}x_{(l)}) \\ &= 2(2d+d^2)\end{aligned} \tag{42}$$

where $x_{(j)}$ denotes the jth entry in vector x, which satisfies $|x_{(j)}| \leq 1$ Similarly, for the coefficients related to k sensitive attributes, we have

$$2\max_{t=(x,y)}\sum_{j=1}^{J}\sum_{\phi\in\Phi_s}||\lambda_{\phi t}||_1 = 2(2k+kd) \tag{43}$$

Thus $\beta_2 = \frac{2(2k+kd)}{2(2d+d^2)} = \frac{k}{d}$. Similarly we have $\beta_1 = \frac{d-k}{d}$. □

Result 8. For logistic regression with database D defined in Definition 14, Algorithm 8 can be applied with the input objective function being $\sum_{i=1}^{n}\frac{1}{8}(x_i^T\omega)^2 + \sum_{i=1}^{n}(\frac{1}{2}-y_i)x_i^T\omega$, $\Delta = \frac{d^2}{4}+3d$, where d is the dimension of vector x_i. Assume there are k sensitive attributes, we have $\beta_1 = \frac{d-k}{d}, \beta_2 = \frac{k}{d}$.

Proof. The calculation result of Δ, as well as the approximated objective function, was provided before in [131]. For the coefficients related to k sensitive attributes, we have

$$2\max_{t=(x,y)}\sum_{j=1}^{J}\sum_{\phi\in\Phi_s}||\lambda_{\phi t}||_1 = \frac{kd}{4}+3k \tag{44}$$

Thus $\beta_2 = \frac{k}{d}$. Similarly we have $\beta_1 = \frac{d-k}{d}$. □

6.4 Evaluation

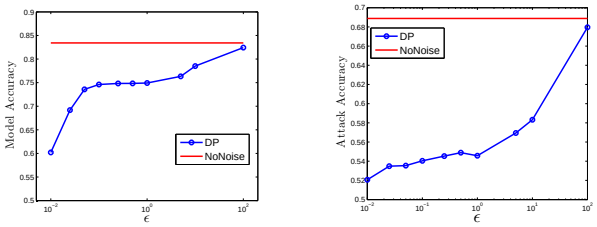
In our experiments, we mainly focus on the problem of releasing the logistic regression model which is defined in Definition 14 under differential privacy against model inversion

attack. We use the Adult dataset from the UCI Machine Learning Repository to evaluate the performance of Algorithm 8. In our experiment, we apply five-fold cross validation for all the accuracy calculation.

The Adult dataset contains census information of 30,175 individuals with 14 attributes such as age, workclass, education, marital-status, hours-per-week and so on. The regression task is to predict whether the income of an individual is greater than 50K. Among the 13 input attributes, we pick "Marital status" as the sensitive attribute which the model inversion attack would target. The marginal probability for status "married" ("unmarried") is 0.466 (0.534).

Figure 15 shows how the accuracy of the released model and the accuracy of the model inversion attack are affected by different ϵ values varying from 0.01 to 100. In this experiment, we do not differentiate the privacy budget for sensitive attribute and non-sensitive attribute. We can see from Figure 15(a) that the prediction accuracy (on income) of the regression model increases as ϵ increases and from Figure 15(b) the accuracy of the model inversion attack on marital status also increases as as ϵ increases. This is not surprising because the larger ϵ is, the less noise introduced in the released model. Hence, the model has high utility but also incurs high risk under model inversion attacks. When $\epsilon < 0.05$, the accuracy of model inversion attacks can hardly beat random guessing based on the marginal probability whereas the prediction accuracy of the released model also drops rapidly. On the contrary, when $\epsilon \geq 5$, both model prediction and model inversion attack show significantly higher accuracy values.

Our next experiment focuses on the situation that we set the privacy threshold for non-sensitive attributes, $\epsilon_n = 5$, and change the privacy threshold for sensitive attribute ϵ_s from



(a) Model Accuracy (b) Attack Accuracy

Figure 15: Accuracy of logistic regression model and that of model inversion attack vs. varying ϵ

$\{0.05, 0.1, 0.25, 0.5, 1, 5\}$.

From Result 8, we can easily derive $\epsilon = (\beta_1 + \gamma\beta_2)\epsilon_n = \epsilon_n - (1-\gamma)\beta_2\epsilon_n$. Thus ϵ values corresponding to all pairs of ϵ_n, ϵ_s are within the range of $\{4.6, 5\}$. Figure 16 shows the accuracy trend of the model prediction task and the model inversion attack. We can see that the prediction accuracy of the release model generally stays stable. However, the accuracy of the model inversion attack on marital-status shows a clear trend of decreasing as ϵ_s goes small. We can conclude that our approach can decrease the privacy risk due to the model inversion attack by adding more noise to polynomial coefficients involving sensitive attributes while retaining the utility of the released regression model.

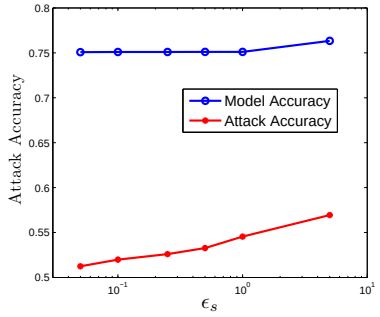


Figure 16: Accuracy of logistic regression and that of model inversion attack vs. varying ϵ_s when $\epsilon_n = 5$

6.5 Related Work

Differential privacy research has been significantly studied from the theoretical perspective, e.g., [17, 53, 65, 71, 130]. There are also studies on the applicability of enforcing differential privacy in real world applications, e.g., collaborative recommendation [85], logistic regression [17, 131], publishing contingency tables [8, 124] or data cubes [25], privacy preserving integrated queries [84], computing graph properties such as degree distributions [51] and clustering coefficient [93]in social network analysis. The mechanisms of achieving differential privacy mainly include the classic approach of adding Laplacian noise [32], the exponential mechanism based on the smooth sensitivity [85], and the functional mechanism by perturbing the objective function [17, 131].

There are several studies that showed differential privacy still could leak various type of private information. In [65], the authors showed that when rows in a database are correlated, or when previous exact statistics for a dataset have been released, differential privacy cannot achieve the ultimate privacy goal – nearly all evidence of an individual's participation should be removed. The authors in [22] showed that if one is allowed to pose certain queries relating sensitive attributes to quasi-identifiers, it is possible to build a differentially-private Naive Bayes classifier that accurately predicts the sensitive attribute.

6.6 Summary

Recent work [38] showed that the existing differential privacy mechanisms cannot prevent model inversion attacks while retaining desirable model efficacy. In this chapter, we have developed an effective approach which simultaneously protects differential privacy of participants and prevents sensitive attribute disclosure of regular individuals due to model

inversion attacks while retaining the efficacy of released regression models. Leveraging the functional mechanism [131], our approach rewrites the objective function in its polynomial representation and adds more (less) noise to the polynomial coefficients with (w/o) sensitive attributes. Our approach can effectively weaken the correlation between the sensitive attributes with the output to prevent model inversion attacks whereas retaining the utility of the released model by decreasing the perturbation effect on non-sensitive attributes. As a result, we still achieve ϵ-differential privacy for participants. Technical report version of this work can be found in [106].

CHAPTER 7: USING RANDOMIZED RESPONSE FOR DIFFERENTIAL PRIVACY PRESERVING DATA COLLECTION

7.1 Introduction

There are roughly two scenarios in the data privacy protection. One is the privacy preserving data publishing scenario, as in which a trusted server releases datasets of individual information or answers queries on such datasets. The other one is the data collection scenario, as in which an untrusted server collects personal information from individuals. We have mostly focused on problems in the first scenario in previous chapters.

In this chapter, we study how to protect privacy in the data collection scenario by using randomized response, a surveying technique for learning statistics on individuals' sensitive attribute information such as whether the survey respondent has cheated in an exam. Randomized response is purely a client-based privacy solution. It does not rely upon a trusted third-party server and puts control over data back to clients. Given a client's value x, the randomized algorithm executed by the client reports to the untrusted server a perturbed value y. The parameters of the randomized algorithm are chosen in such a way so that to limit the server's ability to learn with confidence what value x was. For example, the survey respondent can flip a biased coin, in secret, and answer the truth if it comes up head, but tell the opposite answer if it comes up tail. Using this procedure, the respondent retains confidentiality of their true value due to coin randomness.

In our analysis, we still adopt the rigorous differential privacy, which has been widely

studied in the data publishing or query answering scenario [29] to ensure that the output of the algorithm does not significantly depend on any particular individual's data and to ensures that an adversary should not be able to confidently infer whether a particular individual is present in a database even with access to every other entry in the database and an unbounded computational power. In the data collection scenario, the inference is in terms of the sensitive value of one individual. In particular, we study how to derive the optimal distortion matrix used in the randomized response given a differential privacy threshold.

Differential privacy of each individual value can also be achieved by using the classic Laplace mechanism [32], which is based on query-output independent adding of Laplace noise. We study the relationship between the randomized response and the Laplace mechanism and compare their performance in terms of utility preservation under the same privacy threshold. Our research starts from the simple case of data collection with one single binary attribute and extends to the general case with multiple polychotomous attributes. We evaluate utility preservation in terms of individual value estimate, proportion estimate, and various derived statistics (e.g., entropy and χ^2). Existing works on investigating the accuracy-privacy tradeoff in differentia privacy often define the accuracy in terms of the variance, or magnitude expectation of the noise added to the query output [50, 74]. For example, the authors [74] studied how to optimize linear counting queries under differential privacy and defined the error as the mean squared error of query output estimates, which corresponds to the variance of the noise added to the query output to preserve differential privacy. In this chapter we also measure the utility in terms of the mean squared error of the estimate when randomized response is applied. In particular, we theoretically derive the explicit formula of the mean squared error of various derived statistics based on the

randomized response theory.

We conduct our empirical evaluation on a biomarker dataset and a physical activity social network extracted from from the YesiWell pilot study about health. We compare the performance of the randomized response and that of the Laplace mechanism and report their estimates and standard deviations. One advantage of the use of the randomized response in the data collection scenario is that the collected data can be released for as much analysis as needed without worrying further privacy disclosure. This is different from the output perturbation where each additional analysis consumes further privacy budget. Moreover, the use of the randomized response for collecting data incurs less utility loss than the output perturbation when the sensitivity of functions is high, as demonstrated in our experiment where we calculate the number of triangles in the social network while preserving differential privacy.

The rest of this chapter is organized as follows. In Section 7.2 we present preliminaries of randomized response and differential privacy. In Section 7.3 we focus on the scenario with one single binary attribute and consider two types of queries, individual value estimate and proportion estimate. We compare the utility preservation under the same privacy threshold between the randomized response approach and the Laplace mechanism. In Section 7.4, we extend to a sensitive polychotomous attribute with multiple mutually exclusive and exhaustive classes. In Section 7.5, we further extend to multiple polychotomous attributes and examine the accuracy of derived statistics from the randomized data under both the randomized response approach and the Laplace mechanism. We conduct our empirical evaluation using the real dataset from a health study including a biomarker table and a physical activity social network in Section 7.6. We discuss related work in Section 7.7.

Finally we offer our concluding remarks and discuss future work in Section 7.8.

7.2 Background

7.2.1 Randomized Response

Suppose there are n individual clients $C_1, ..., C_n$; each client C_i has some private value x_i regarding a sensitive attribute X. An untrusted server needs to learn certain aggregate (statistical) properties of the individual's private data. But the clients are reluctant to disclose their personal information x_i. To ensure privacy, each client C_i only sends to the server a perturbed version y_i of x_i. The server collects the perturbed information from all individuals and then recovers the statistical properties by following some reconstruction procedures.

We assume every private value x_i about an individual belongs to the same fixed domain V_X and each x_i is chosen independently at random from the same fixed probability distribution π_X. Note that this distribution is not private and is unknown to clients. The server aims to reconstruct the distribution π_X or derive some statistical properties of this distribution. The independence assumption ensures that the private information x_j of all individuals C_j besides C_i tells nothing new about C_i's own private information x_i once the distribution π_X is learned.

To protect privacy, each individual C_i hides its own sensitive information x_i by applying a randomization algorithm. A random instance y_i is sent to the untrusted server. The domain of all possible output of y_i is denoted by V_Y. The server receives y_i from client C_i and tries to learn distribution π_X.

We omit the formal definition of differential privacy and the classic Laplace mechanism to achieve differential privacy here, since they can be referred from chapter 2.

7.3 Binary Attribute

Suppose there are n individuals $C_1, ..., C_n$ and each individual C_i has a private binary value $x_i \in \{0, 1\}$ regarding a sensitive binary attribute X. To ensure privacy, each individual C_i sends to the untrusted server a modified version y_i of x_i. Using the randomized response, the server can collect perturbed data from individuals.

7.3.1 Randomized Response

A randomized response scheme on a binary attribute X follows a 2×2 design matrix (also called distortion matrix):

$$\mathbb{P} = \begin{pmatrix} p_{00} & p_{01} \\ p_{10} & p_{11} \end{pmatrix} \tag{45}$$

where $p_{uv} = P[y_i = u | x_i = v]$ $(u, v \in \{0, 1\})$ denotes the probability that the random output is u when the real attribute value x_i for C_i is v; here $p_{uv} \in (0, 1)$. In the design matrix, the sum of probabilities of each column is 1.

In this section, we focus on two types of classic queries in the data collection scenario.

- Q1: what is the probability of correctly estimating x_i of individual C_i corresponding to the sensitive binary attribute X?
- Q2: what is the proportion of $X = 1$ $(X = 0)$?

For Q1, the original value $x_i = v (\in \{0, 1\})$ is outputted as $y_i = u (\in \{0, 1\})$ with probability p_{uv} from the design matrix $\mathbb{P}$ in Equation 45. Let $Pr(x_i = v \rightarrow \hat{x}_i = v)$ denote the probability of correctly reconstructing the individual's value as v from the perturbed data, given that the original value x_i is v where $v \in \{0, 1\}$. This reconstruction probability

implies how much information is preserved in the randomization process.

$$Pr(x_i = v \rightarrow \hat{x}_i = v) = \sum_{u=0,1} P(y_i = u|x_i = v)P(\hat{x}_i = v|y_i = u) \tag{46}$$

Q2 aims to learn the population distribution based on the collected randomized dataset. We use π_0 (π_1) to denote the true proportion of value 0 (1) to be estimated in the original population. The observed proportion of value 0 (1) in the collected dataset is denoted as $\lambda_0(\lambda_1)$. We denote the unbiased estimator for π_0, π_1 respectively as $\hat{\pi}_0, \hat{\pi}_1$.

Lemma 13. (Chapter 1.2 [16]) Given the design matrix $\mathbb{P}$ and the observed proportion of value $b(\in \{0,1\})$ in randomized dataset $\hat{D}_{rr}$, an unbiased estimator of the fraction of records whose attribute value is $b(\in \{0,1\})$ is

$$\hat{\pi}_b = \frac{p_{bb} - 1}{2p_{bb} - 1} + \frac{\lambda_b}{2p_{bb} - 1}, \tag{47}$$

where $p_{bb} \neq 0.5$ and $0 < p_{bb} < 1$. Since the observed number of records whose attribute value equals b follows binomial distribution, the variance of $\hat{\pi}_b$ is

$$var(\hat{\pi}_b) = \frac{\hat{\pi}_b(1 - \hat{\pi}_b)}{n - 1} + \frac{1}{n - 1}[\frac{1}{16(p_{bb} - 0.5)^2} - \frac{1}{4}] \tag{48}$$

which is the expected error for the estimator $\hat{\pi}_b$.

7.3.2 Randomized Response vs. Laplace Mechanism

The values in each row u ($u \in \{0,1\}$) of the design matrix denote the probability that the random output is u. For example, p_{00} (p_{01}) denotes the distortion probability that the random output value is 0 when the real individual value is 0 (1). Without loss of generality, we assume the randomized response still favors the true value, i.e., $p_{00}, p_{11} > 0.5$. Differ-

ential privacy requires that $p_{00}/p_{01} \leq e^{\epsilon}$. Thus we show how the randomized response will achieve differential privacy in the following result. In addition, we also give the form of the design matrix that is expected to achieve the optimal utility while satisfying the given ϵ-differential privacy.

Result 9. For a given differential privacy parameter ϵ, the randomized response scheme following the design matrix $\mathbb{P}$ in Equation 45 satisfies ϵ-differential privacy if $\max\{\frac{p_{00}}{p_{01}}, \frac{p_{11}}{p_{10}}\} \leq e^{\epsilon}$.

In order to maximize $p_{00} + p_{11}$ while satisfying ϵ-differential privacy, the design matrix should have the following pattern,

$$\mathbb{P}_{rr} = \begin{pmatrix} \frac{e^{\epsilon}}{1+e^{\epsilon}} & \frac{1}{1+e^{\epsilon}} \\ \frac{1}{1+e^{\epsilon}} & \frac{e^{\epsilon}}{1+e^{\epsilon}} \end{pmatrix} \tag{49}$$

Proof. Assume $\frac{p_{00}}{p_{01}} = p$,$\frac{p_{11}}{p_{10}} = q$. In order to satisfy ϵ-differential privacy, we have $1 < p \leq e^{\epsilon}$ and $1 < q \leq e^{\epsilon}$. In this case, the distortion matrix will have the general form:

$$\mathbb{P}_{rr} = \begin{pmatrix} \frac{p(q-1)}{pq-1} & \frac{q-1}{pq-1} \\ \frac{p-1}{pq-1} & \frac{(p-1)q}{pq-1} \end{pmatrix}$$

We denote

$$func(p, q) = \mathbb{P}_{rr}(1, 1) + \mathbb{P}_{rr}(2, 2) = \frac{p(q-1)}{pq-1} + \frac{(p-1)q}{pq-1}.$$

Since $\frac{\partial func}{\partial p} = \frac{(q-1)^2}{(pq-1)^2} > 0$ and $\frac{\partial func}{\partial q} = \frac{(p-1)^2}{(pq-1)^2} > 0$ when $p, q \in (1, e^{\epsilon}]$, thus $func$ will achieve its maximum value if and only if $p = q = e^{\epsilon}$. In this way, we get the form in Equation 49. □

Similarly, individual C_i can achieve differential privacy by using the Laplace mecha-

nism. The Laplace mechanism is to firstly add a random noise generated from the Laplace distribution with parameter $\frac{1}{\epsilon}$ (with a given ϵ and the global sensitivity of 1) to the true answer x_i. Since the output should be binary value, we postprocess the perturbed result by outputting 0 if the perturbed value is less than c and outputting 1 otherwise, shown in Equation 50.

$$y_i = \begin{cases} 0; \text{ if } x_i + Lap(1/\epsilon) < c \\ \\ 1; \text{ if } x_i + Lap(1/\epsilon) \geq c \end{cases} \tag{50}$$

The probability of $y_i = 0$ is $F_{x_i,1/\epsilon}(c)$ and the probability of $y_i = 1$ is $1 - F_{x_i,1/\epsilon}(c)$ where $F_{\mu,b} = \frac{1}{2} + \frac{1}{2}sgn(x-\mu)(1 - e^{(-\frac{|x-\mu|}{b})})$ denotes the cumulative distribution function of Laplace distribution $Lap(\mu, b)$ with the location parameter μ and the scale parameter b (and with the mean μ and variance $2b^2$). Thus we can map the Laplace mechanism to the randomized response with the design matrix as

$$\mathbb{P}_{lm} = \begin{pmatrix} F_{0,1/\epsilon}(c) & F_{1,1/\epsilon}(c) \\ \\ 1 - F_{0,1/\epsilon}(c) & 1 - F_{1,1/\epsilon}(c) \end{pmatrix} \tag{51}$$

For a given ϵ, the perturbed result of Laplace mechanism satisfies differential privacy because the postprocessing process does not consume any privacy budget. Thus we have the following result indicating the Laplace mechanism with the postprocessing satisfies ϵ-differential privacy. We also show that the best postprocessing strategy is to set $c = 0.5$ for Equation 50.

Result 10. For a given differential privacy parameter ϵ, the Laplace mechanism based scheme with the postprocessing strategy following Equation 50 satisfies ϵ-differential privacy.

The corresponding design matrix should have the following form,

$$\mathbb{P}_{lm} = \begin{pmatrix} 1 - \frac{1}{2}e^{-\frac{\epsilon}{2}} & \frac{1}{2}e^{-\frac{\epsilon}{2}} \\ \frac{1}{2}e^{-\frac{\epsilon}{2}} & 1 - \frac{1}{2}e^{-\frac{\epsilon}{2}} \end{pmatrix}, \tag{52}$$

in order to maximize $p_{00} + p_{11}$ while satisfying ϵ-differential privacy.

Proof. With the assumption that we need to preserve the real value with probability greater than 0.5, c in Equation 50 is in the range [0,1]. We have

$$\mathbb{P}_{lm} = \begin{pmatrix} 1 - \frac{1}{2}e^{-c\epsilon} & \frac{1}{2}e^{-(1-c)\epsilon} \\ \frac{1}{2}e^{-c\epsilon} & 1 - \frac{1}{2}e^{-(1-c)\epsilon} \end{pmatrix}.$$

We denote

$$func(c) = \mathbb{P}_{lm}(1,1) + \mathbb{P}_{lm}(2,2) = 1 - \frac{1}{2}e^{-c\epsilon} + 1 - \frac{1}{2}e^{-(1-c)\epsilon},$$

where $c \in [0,1]$. Since

$$\frac{\partial func}{\partial c} = \frac{\epsilon}{2}(e^{-c\epsilon} - e^{-(1-c)\epsilon}),$$

we have

$$\frac{\partial func}{\partial c} \begin{cases} > 0, & \text{when } c \in [0, 0.5) \\ < 0, & \text{when } c \in (0.5, 1]. \end{cases}$$

Thus the maximum value for $func(c)$ is achieved when $c = 0.5$. □

7.3.3 Utility Comparison

In this chapter we measure the utility in terms of the mean squared error of the estimate for x_i given a randomized mechanism A.

$$Error_A(\hat{x}_i) = \mathbb{E}[(\hat{x}_i - x_i)^2] \tag{53}$$

Note that by replacing $P(y_i = u|x_i = v)$ in Equation 46 with values in $\mathbb{P}_{rr}$ of the randomized response and those in $\mathbb{P}_{lm}$ of the Laplace mechanism, we can calculate the estimates $\hat{x}$ respectively. We can then compare the utility of the randomized response with that of the Laplace mechanism based on Equation 53.

Intuitively, under the same privacy standard, the mechanism with larger diagonal elements in the corresponding design matrix tends to achieve better utility. And the diagonal elements in Equation 49 is larger than those in Equation 52. Based on such tuition, we can prove that the randomized response actually can achieve better utility than the classic Laplace mechanism in the scenario of binary data collection.

Theorem 9. Given ϵ, for the randomized response scheme with $\mathbb{P}_{rr}$ and the Laplace mechanism based on $\mathbb{P}_{lm}$, we have $E_{RROR_{rr}}(\hat{x}_i) \leq E_{RROR_{lm}}(\hat{x}_i)$.

Proof. We have $(\hat{x}_i - x_i)^2 = 0$ with probability $Pr(x_i = v \rightarrow \hat{x}_i = v)$ in Equation 46; and $(\hat{x}_i - x_i)^2 = 1$ with probability $1 - Pr(x_i = v \rightarrow \hat{x}_i = v)$. So

$$
\begin{aligned}
&E_{RROR_A}(\hat{x}_i) \\
&= 0 \times Pr(x_i = v \rightarrow \hat{x}_i = v) + 1 \times (1 - Pr(x_i = v \rightarrow \hat{x}_i = v)) \\
&= 1 - Pr(x_i = v \rightarrow \hat{x}_i = v).
\end{aligned}
$$

Without loss of generality, assume $v = 1$, we denote the prior probability of $x_i = 1$ as π_1. According to Bayes's theorem, we have

$$
E_{RROR_A}(\hat{x}_i) = 1 - \left(\frac{p_{11}^2 \pi_1}{p_{11}\pi_1 + p_{10}(1 - \pi_1)} + \frac{p_{01}^2 \pi_1}{p_{01}\pi_1 + p_{00}(1 - \pi_1)}\right).
$$

For $E_{RROR_{lm}}(\hat{x}_i)$, we have $p_{11}^{lm} = p_{00}^{lm} = 1 - \frac{1}{2}e^{-0.5\epsilon}$. For $E_{RROR_{rr}}(\hat{x}_i)$, we have $p_{11}^{rr} =$

$p_{00}^{rr} = \frac{e^{\epsilon}}{e^{\epsilon}+1}$. Now we are to prove: for $\pi_1 \in [0,1]$,

$$func(\pi_1) = Error_{lm}(\hat{x}_i) - Error_{rr}(\hat{x}_i) \geq 0.$$

For a given $\epsilon > 0$, since $func(\pi_1)$ is continual and it has only two roots for the parameter range $\pi_1 \in [0,1]$. Respectively they are $\pi_1 = 0$ and $\pi_1 = 1$. It indicates that all $\pi_1 \in (0,1)$, the output of $func(\pi_1)$ has the same sign. Thus, we only need to prove for one specific π_1, say $\pi_1 = 0.5$, that $func(\pi_1) > 0$. Then the same result holds for all $\pi_1 \in (0,1)$.

In this case, we have

$$Error_A(\hat{x}_i) = 1 - (p_{11}^2 + (1 - p_{11}^2)).$$

Since $p_{11}^{rr} > p_{11}^{lm} > 0.5$ for all $\epsilon > 0$, we have $func(0.5) > 0$. Thus for all $\pi_1 \in (0,1)$ we have $func(\pi_1) > 0$. The same idea can be applied for the situation of $v = 0$. So for $\pi_1 \in [0,1]$, we have $Error_{rr}(\hat{x}_i) \leq Error_{lm}(\hat{x}_i)$. □

Using either the randomized response or the Laplace mechanism based approach, the server can collect private data from individuals. Both collected datasets satisfy ϵ-differential privacy (rather than $n\epsilon$-differential privacy) according to the independence assumption. Formally, $\hat{D}_{rr}$ denotes the dataset generated by the randomized response following the design matrix $\mathbb{P}_{rr}$ as in Equation 49. Similarly, $\hat{D}_{lm}$ denotes the dataset generated by the Laplace mechanism. We define the expected error of the estimator $\hat{\pi}_b$ as its variance, $Error_A(\hat{\pi}_b) = var(\hat{\pi}_b)$.

Theorem 10. Given ϵ, for the randomized response scheme with $\mathbb{P}_{rr}$ and the Laplace mechanism based on $\mathbb{P}_{lm}$, we have $Error_{rr}(\hat{\pi}_b) \leq Error_{lm}(\hat{\pi}_b)$.

Proof. From Equation 48, we see comparing the utility of estimation from $\hat{D}_{rr}$ and $\hat{D}_{lm}$

relies only on p_{bb} in the distortion matrix $\mathbb{P}$. For the Laplace mechanism, we have $p_{00}^{lm} = p_{11}^{lm} = 1 - \frac{1}{2}e^{-\frac{\epsilon}{2}}$; For the randomized response, we have $p_{11}^{rr} = p_{00}^{rr} = \frac{e^{\epsilon}}{e^{\epsilon}+1}$. Since $p_{11}^{rr} > p_{11}^{lm} > 0.5$ for all $\epsilon > 0$, we have $E_{RROR_{rr}}(\hat{\pi}_b) \leq E_{RROR_{lm}}(\hat{\pi}_b)$, according to Equation 48. □

7.4 Polychotomous Attribute

In the previous section, we compared the Laplace mechanism and the randomized response approach in collecting information about one private binary attribute. In this section, we extend to a sensitive polychotomous attribute with $t(t \geq 2)$ mutually exclusive and exhaustive classes.

7.4.1 Randomized Response

The corresponding unknown proportions to be estimated are denoted as $\pi_1, ..., \pi_t$. The randomization device is such that an individual belonging to the vth category($v = 1, ..., t$) reports a random value u ($u = 1, ..., t$) with probability p_{uv} and $\Sigma_{u=1}^{t} p_{uv} = 1$ for all $v = 1, ..., t$.

The matrix $\mathbb{P} = \{p_{uv}\}$ is called the design matrix, where the sum of each column in $\mathbb{P}$ is 1.

$$\mathbb{P} = \begin{pmatrix} p_{11} & p_{12} & \cdots & p_{1v} & \cdots & p_{1t} \\ p_{21} & p_{22} & \cdots & p_{2v} & \cdots & p_{2t} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ p_{u1} & p_{u2} & \cdots & p_{uv} & \cdots & p_{ut} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ p_{t1} & p_{t2} & \cdots & p_{tv} & \cdots & p_{tt} \end{pmatrix} \tag{54}$$

Similarly we have two types of classic queries in the data collection scenario.

- Q1: what is the probability of correctly estimating x_i of individual C_i corresponding to the sensitive attribute X?

- Q2: what is the proportion of $X = 1, \cdots, t$?

Let $Pr(x_i = v \rightarrow \hat{x}_i = v)$ denote the probability of correctly reconstructing the individual's value as v from the perturbed data, given that the original value x_i is v where $v \in \{1, \cdots, t\}$. This reconstruction probability implies how much information is preserved in the randomization process.

$$\begin{aligned} &Pr(x_i = v \rightarrow \hat{x}_i = v) = \\ &\sum_{u=1}^{t} P(y_i = u | x_i = v) P(\hat{x}_i = v | y_i = u) \end{aligned} \tag{55}$$

The probability λ_u of the (randomized) response u is given by

$$\lambda_u = \Sigma_{v=1}^{t} p_{uv} \pi_v \qquad (u = 1, ..., t) \tag{56}$$

Defining $\lambda = (\lambda_1, ..., \lambda_t)'$, $\pi = (\pi_1, ..., \pi_t)'$, we obtain in matrix notation

$$\lambda = \mathbb{P}\pi \tag{57}$$

Lemma 14. (Chapter 3.3 [16])With a simple random sample with replacement of size n, let $\hat{\lambda}$ be the vector of sample proportions corresponding to λ. Then assuming the nonsingularity of the design matrix $\mathbb{P}$. An unbiased estimator of π emerges as

$$\hat{\pi} = \mathbb{P}^{-1}\hat{\lambda}. \tag{58}$$

An unbiased estimator of the dispersion matrix is given by

$$disp(\hat{\pi}) = (n-1)^{-1}\mathbb{P}^{-1}(\hat{\lambda}^{\delta} - \hat{\lambda}\hat{\lambda}')\mathbb{P}'^{-1}, \tag{59}$$

where $\hat{\lambda}^{\delta}$ is a diagonal matrix with the same diagonal elements as those of $\hat{\lambda}$

7.4.2 Randomized Response vs. Laplace Mechanism

Similar as the binary case, the values in each row u ($u \in \{1, 2, ..., t\}$ of the design matrix denote the probability that the random output is u. Differential privacy requires that the maximum value difference in each row is bounded by e^{ϵ}. Thus we have the following result.

Result 11. The randomized response is ϵ-differentially private if $\epsilon \geq ln \max_{u=1..t} \frac{\max_{v=1..t} p_{uv}}{\min_{v=1..t} p_{uv}}$.

In order to maximize the sum of the diagonal elements,the design matrix for randomized response $\mathbb{P}_{rr} = \{p_{uv}\}$ should be in the following form,

$$p_{uv} = \begin{cases} \frac{e^{\epsilon}}{t-1+e^{\epsilon}}; \text{ if } u = v \\ \\ \frac{1}{t-1+e^{\epsilon}}; \text{ if } u \neq v \end{cases} \tag{60}$$

Proof. For a given predefined distortion matrix $\mathbb{P} = \{p_{uv}\}$, following the randomized strategy with $\mathbb{P}$, called R, to collect data, for an arbitrary randomized response $u(\in 1, 2, ..., t)$ corresponding to input value of $v, v' \in 1, 2, ..., t$, we have

$$\frac{P(R(v) = u)}{P(R(v') = u)} = \frac{p_{uv}}{p_{uv'}} \leq \max_{u=1..t} \frac{\max_{v=1..t} p_{uv}}{\min_{v=1..t} p_{uv}} \leq e^{\epsilon} \tag{61}$$

Similarly as the binary case, in order to maximize the sum of the diagonal elements, the distortion matrix for randomized response $\mathbb{P}_{rr} = \{p_{uv}\}$ should be in the form in Equation 60. □

In other words, in the optimal form of the design matrix, all diagonal entries are set as $\frac{e^{\epsilon}}{t-1+e^{\epsilon}}$ and all off-diagonal entries are set as $\frac{1}{t-1+e^{\epsilon}}$. We can also achieve differential privacy by adding Laplace noise. The global sensitivity is $t - 1$. So the Laplace noise is

generated from the distribution $Lap(\frac{t-1}{\epsilon})$. Because the perturbed outputs are numerical, we postprocess to map them to an index value from 1 to t as shown in Equation 62.

$$y_i = \begin{cases} 1;\ \text{if } x_i + Lap((t-1)/\lambda) \in (-\infty, c_1] \\ 2;\ \text{if } x_i + Lap((t-1)/\lambda) \in (c_1, c_2] \\ ... \\ u;\ \text{if } x_i + Lap((t-1)/\lambda) \in (c_{u-1}, c_u] \\ ... \\ t;\ \text{if } x_i + Lap((t-1)/\lambda) \in (c_{t-1}, \infty) \end{cases} \tag{62}$$

where c_u is in the range $[u, u+1]$.

Note that in this scenario, the strategy of perturbation by Laplace mechanism is also a special case of the randomized response strategy. We give the form of the corresponding design matrix in Equation 64. The following result shows such Laplace mechanism with postprocessing satisfies ϵ-differential privacy. We also give the best postprocessing strategy with the corresponding design matrix.

Result 12. The Laplace mechanism of adding random noise from distribution $Lap(\frac{t-1}{\epsilon})$, with postprocessing strategy following Equation 62 is ϵ- differentially private.

In order to maximize the sum of the diagonal elements in the corresponding design matrix for Laplace mechanism, $\mathbb{P}_{lm}$, we have $c_u = u + 0.5$ for $u \in \{1, 2, ..t-1\}$ in

equation 62. The corresponding design matrix $\mathbb{P}_{lm} = \{p_{uv}\}$ has the following form,

$$p_{uv} = \begin{cases} F_{v,\frac{\epsilon}{t-1}}(1.5); \text{ if } u = 1 \\ 1 - F_{v,\frac{\epsilon}{t-1}}(t - 0.5); \text{ if } u = t \\ F_{v,\frac{\epsilon}{t-1}}(u + 0.5) - F_{v,\frac{\epsilon}{t-1}}(u - 0.5); \text{ otherwise} \end{cases} \tag{63}$$

where $F_{v,\frac{\epsilon}{t-1}}$ is the cumulative distribution function of Laplace distribution with mean value of v ($v \in \{1, 2, ..., t\}$), variance of $2\lambda^2$ and $\lambda = (t-1)/\epsilon$.

Proof. The distortion matrix for the Laplace mechanism with the postprocessing strategy following Equation 62 has the following general form, where $u, v \in \{1, 2, ..., t\}$

$$\mathbf{P_{lm}} = \begin{pmatrix} F_1(c_1) & ... & F_v(c_1) & ... & F_t(c_1) \\ F_1(c_2) - F_1(c_1) & ... & F_v(c_2) - F_v(c_1) & ... & F_t(c_2) - F_t(c_1) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ F_1(c_u) - F_1(c_{u-1}) & ... & F_v(c_u) - F_v(c_{u-1}) & ... & F_t(c_u) - F_t(c_{u-1}) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 - F_1(c_{t-1}) & ... & 1 - F_v(c_{t-1}) & ... & 1 - F_t(c_{t-1}) \end{pmatrix} \tag{64}$$

We denote the function of the sum of the diagonal elements in the distortion matrix for the Laplace mechanism $\mathbb{P}_{lm}$ as

$$func(c_1, c_2, ..., c_{t-1}) = \sum_{i=1}^{t} \mathbb{P}_{lm}(i, i).$$

We have

$$\begin{aligned} func(c_1, c_2, ..., c_{t-1}) &= F_1(c_1) + (F_2(c_2) - F_2(c_1)) + ... + (F_v(c_v) - F_v(c_{v-1})) \\ &\quad + ... + (F_{t-1}(c_{t-1}) - F_{t-1}(c_{t-2})) + (1 - F_t(c_{t-1})) \\ &= (F_1(c_1) - F_2(c_1)) + (F_2(c_2) - F_3(c_2)) + ... \\ &\quad + (F_v(c_v) - F_{v+1}(c_v)) + ... + (F_{t-1}(c_{t-1}) - F_t(c_{t-1})) + 1 \end{aligned} \tag{65}$$

In order to maximize $func(c_1, c_2, ..., c_{t-1})$, we need to find each c_v for $v \in \{1, 2, ..., t-1\}$ that maximize $F_v(c_v) - F_{v+1}(c_v)$. Since $v \leq c_v \leq v + 1$, we have

$$F_v(c_v) - F_{v+1}(c_v) = 1 - \frac{1}{2}e^{-\frac{(c_v - v)\epsilon}{t-1}} - \frac{1}{2}e^{-\frac{(v+1-c_v)\epsilon}{t-1}},$$

which is maximized when $c_v = v + 0.5$. Thus $func(c_1, c_2, ..., c_{t-1})$ is maximized when $c_v = v + 0.5$ for all $v \in \{1, 2, ..., t-1\}$. □

7.4.3 Utility Comparison

Intuitively, the utility depends on the diagonal elements in the design matrix. The Laplace mechanism based approach degrades the utility by favoring the values near the correct value. The sum of diagonal elements in $\mathbb{P}_{lm}$ is actually smaller than that in $\mathbb{P}_{rr}$(Details shown in the proof of Theorem 11). In consistence with the binary case, the randomized response achieves better utility than the classic Laplace mechanism in data collection scenario.

Theorem 11. Given ϵ, for the randomized response scheme with $\mathbb{P}_{rr}$ and the Laplace mechanism based on $\mathbb{P}_{lm}$, we have $E_{RROR_{rr}}(\hat{x}_i) \leq E_{RROR_{lm}}(\hat{x}_i)$.

Proof. Intuitively, the utility depends on the diagonal elements in the distortion matrix. The Laplace mechanism based approach will degrade the utility by favoring the values

near the correct value. We first give the proof that the sum of diagonal elements in $\mathbb{P}_{lm}$ is smaller than that in $\mathbb{P}_{rr}$. Here for the Laplace mechanism, we have $c_v = v + 0.5$ for all $v \in \{1, 2, ..., t-1\}$. We denote the sum of diagonal elements as

$$Sum_{lm} = t - (t-1)e^{\frac{\epsilon}{2(t-1)}}.$$

For randomized response mechanism, we have

$$Sum_{rr} = \frac{te^{\epsilon}}{t-1+e^{\epsilon}}.$$

For $\epsilon > 0, t > 2$, we have

$$\begin{aligned} & \frac{te^{\epsilon}}{t-1+e^{\epsilon}} > t - (t-1)e^{\frac{\epsilon}{2(t-1)}} \\ \Leftrightarrow\ & \frac{(\tau+1)m}{\tau+m} > \tau + 1 - \tau m^{-\frac{1}{2\tau}} \ \ (\tau = t-1, m = e^{\epsilon}) \\ \Leftrightarrow\ & \quad 0 > (\tau+1)m^{\frac{1}{2\tau}} - (\tau+m) \end{aligned} \tag{66}$$

We denote, for $m > 1, \tau > 1$,

$$func(m, \tau) = (\tau+1)m^{\frac{1}{2\tau}} - (\tau+m).$$

Since $\frac{\partial func}{\partial m} < 0$ and $func(1, \tau) = 0$ for all τ values, we have $func(m, \tau) < 0$ holds for all $m > 1, \tau > 1$. Thus we have Equation 66 holds for any $\epsilon > 0$ and $t > 2$. Then similar as the binary case, we can see that the utility of the randomized response scheme is better than that of the Laplace mechanism with respect to Q1. □

Similarly as the binary case, we define the expected error of the estimator $\hat{\pi}_v$ for the proportion of category v ($v \in \{1, 2, ..., t\}$) as its variance, the diagonal element in the unbiased estimate of dispersion matrix $disp(\hat{\pi})$ following the randomized mechanism A.

We have $E_{RROR_A}(\hat{\pi}_v) = disp(\hat{\pi})_{vv}$ where $v \in \{1, 2, ..., t\}$.

However, it is intractable to directly prove that the randomized response strategy following the design matrix in Equation 60 could achieve lower expected error of the estimator $\hat{\pi}_v$ than the Laplace mechanism based approach following Equation 64 does. But intuitively we can see that the Laplace mechanism based approach will degrade the utility by favoring the values near the correct value. As shown in the proof of Theorem 11, the sum of the diagonal elements in $\mathbb{P}_{lm}$ is smaller than that in $\mathbb{P}_{rr}$, which indicates that the estimation based on the randomized response mechanism following $\mathbb{P}_{rr}$ is expected to achieve smaller error than that based on the Laplace mechanism following $\mathbb{P}_{lm}$.

7.5 Accuracy Analysis of Randomized Dataset

7.5.1 Multiple Attributes

To be consistent with notations, we denote the set of variables by $\mathcal{X} = \{X_1, \cdots, X_s\}$. Note that, for ease of presentation, we use the terms "attribute" and "variable" interchangeably. Each variable X_u has d_u mutually exclusive and exhaustive categories. We use $i_u = 1, \cdots, d_u$ to denote the index of its categories. For each data record, we apply the randomized response model independently on each sensitive variable X_u using different settings of distortion.

Formally, let $\pi_{i_1,\cdots,i_s}$ denote the true proportion corresponding to the categorical combination of s variables $(X_{1i_1}, \cdots, X_{si_s})$ in the original data, where $i_u = 1, \cdots, d_u$ ($u = 1, \cdots, s$), and X_{1i_1} denotes the i_1th category of attribute X_1. Let π be a vector with elements $\pi_{i_1,\cdots,i_s}$ arranged in a fixed order. The combination vector corresponds to a fixed order of cell entries in the contingency table formed by these s variables. Similarly, we denote $\lambda_{i_1,\cdots,i_s}$ as the expected proportion in the randomized data.

For the case of s multi-variables, we denote $\lambda_{\mu_1,\cdots,\mu_s}$ as the expected probability of getting a response $(X_{1\mu_1},\cdots,X_{s\mu_s})$ and λ the vector with elements $\lambda_{\mu_1,\cdots,\mu_s}$ arranged in a fixed order. For example, given a dataset with two variables, *Gender* with domain values {male, female} and *Race* with domain values {black, white,asian}, we have $d_1 = 2$ and $d_2 = 3$. The vector $\pi = (\pi_{11}, \pi_{12}, \pi_{13}, \pi_{21}, \pi_{22}, \pi_{23})'$ corresponds to a fixed order of cell entries π_{ij} in the 2×3 contingency table. π_{12} denotes the proportion of records with *male* and *white*.

Let $P = P_1 \times \cdots \times P_s$, we can obtain

$$\lambda = P\pi = (P_1 \times \cdots \times P_s)\pi \tag{67}$$

where $\times$ stands for the Kronecker product [11].

The original database $\mathcal{D}$ is changed to $\mathcal{D}_{rr}$ after randomization. An unbiased estimate of π based on one given realization $\mathcal{D}_{rr}$ follows as

$$\hat{\pi} = P^{-1}\hat{\lambda} = (P_1^{-1} \times \cdots \times P_s^{-1})\hat{\lambda} \tag{68}$$

where $\hat{\lambda}$ is the vector of proportions calculated from $\mathcal{D}_{rr}$ corresponding to λ and P_u^{-1} denotes the inverse of the matrix P_u.

7.5.2 Variance of Derived Measure

Many measures (including entropy, mutual information, Pearson Correlation, G^2-likelihood) can be expressed as one derived random variable (or function) from the observed variable π. Similarly, its estimate from the randomized data can be considered as another derived random variable from the input variable $\hat{\pi}$. One natural question is how to calculate the

[11] Kronecker product is an operation on two matrices, an m-by-n matrix A and a p-by-q matrix B, resulting in the mp-by-nq block matrix

variance of those estimates. In the following, we introduce the use of the delta method [64] to derive the variance of various measures [46].

Let Z be a random variable derived from the observed random variables X_i ($i = 1, \cdots, k$): $Z = g(X_1, X_2, ..., X_k)$. According to the delta method, a Taylor approximation of the variance of a function with multiple variables can be expanded as

$$\begin{aligned} var\{g(X_1, X_2, ..., X_k)\} = \sum_{i=1}^{k} \{g_i'(\theta)\}^2 var(X_i) \\ + \sum_{i \neq j=1}^{k} \sum g_i'(\theta) g_j'(\theta) cov(X_i, X_j) + o(n^{-r}) \end{aligned} \tag{69}$$

where θ_i is the mean of x_i, $g_i'(\theta)$ is the $\frac{\partial g(X_1, X_2, ..., X_k)}{\partial X_i}$ evaluated at $\theta_1, \theta_2, \cdots, \theta_k$.

We use the entropy function as an example. The entropy function from information theory is defined as follows:

$$H(X) = -\sum_{j \in Range(X)} \pi_j log_2 \pi_j \tag{70}$$

We can estimate the entropy of the discrete random variable X with possible values $\{1, 2, .., t\}$ in the original dataset using the estimator of the distribution π and the estimator of the dispersion matrix $disp(\pi)$ calculated following Equations 58 and 59.

Result 13. The variance of the estimated entropy can be computed following Equation 69 where $i, j \in \{1, 2, ..., t\}$, $k = t$, $X_i = \hat{\pi}_i$, $\frac{\partial g(X_1, X_2, ..., X_k)}{\partial X_i} = log_2 \hat{\pi}_i + \frac{1}{ln2}$ and $var(X_i) = disp(\hat{\pi})(i, i)$, $cov_{i \neq j}(X_i, X_j) = disp(\hat{\pi})(i, j)$.

Different from the entropy which involves only one variable, some measures such as chi-square statistics involve multiple variables.

$$\hat{\chi}^2 = n \sum_i \sum_j \frac{\{\pi_{ij} - \pi_{i+}\pi_{+j}\}^2}{\pi_{i+}\pi_{+j}} \tag{71}$$

It is easy to see $\hat{\chi}^2$ can be considered as one derived variable from the observed elements $\hat{\pi}_{X_1 \cdots X_s}$ and the marginal totals of the contingency table. Following the same delta method, we can derive its variance.

7.6 Empirical Evaluation

7.6.1 YesiWell Data

We conduct our empirical evaluation using the real dataset collected from the YesiWell pilot study. The study was conducted in 2010-2011 as the collaboration among several health laboratories and universities to help people maintain active lifestyles and lose weight. Data gained from this study includes information of various domains such as biomarker, biometrics, social activities and so on.

We conduct experiments on a chosen table which contains 248 individualss' biomarker information. In particular, we focus on two sensitive attributes: LDL cholesterol (LDL) with six domain levels and Total cholesterol (TC) with three domain levels. Under each differential privacy threshold ϵ, we compare the performance of the randomized response (with the corresponding derived design matrix $\mathbb{P}_{rr}$) and that of the Laplace mechanism (with the corresponding derived design matrix $\mathbb{P}_{lm}$) from the utility preservation perspective. We focus on proportion estimates of categories based on LDL levels, the derived entropy of LDL, and the χ^2 statistics of LDL and TC. For each statistics, we report their estimate values and derive standard deviations for two strategies: randomized response and Laplacian mechanism.

We also conduct experiments on the YesiWell physical activity social network which contains 185 users and 684 interactions. Each interaction, represented as an edge between two user nodes, is considered sensitive in our context. We study how to enforce edge

differential privacy in our social network, i.e., the inclusion or exclusion of a link between two individuals from the graph makes no statistical difference to the results found. We focus on two classic graph features: the degree sequence $D = \{d_i\}$ where each entry represents the degree of node i, and the number of triangle sequence $N_\Delta = \{N_\Delta(i)\}$ where each entry represents the number of triangles involving node i. We compare the performance of the randomized response and that of the Laplace mechanism and report their estimates and standard deviations for the above two graph statistics.

In addition to our above study in the data collection scenario, we also compare our randomized response with two mechanisms, Laplacian mechanism and smooth sensitivity, in the data query answering scenario where the trusted server keeps all unperturbed values and returns differential privacy preserving query answers.

7.6.2 Proportion Estimate

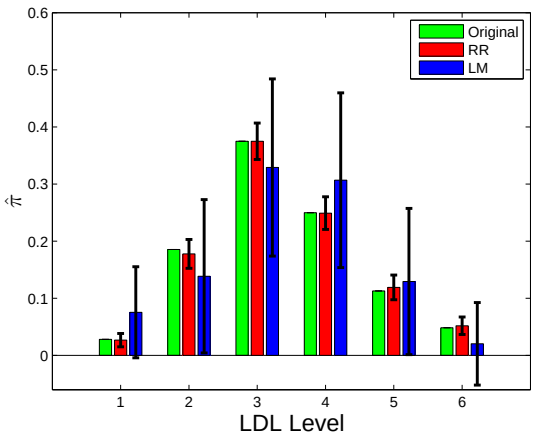


Figure 17: The estimation of the distribution of each LDL level when $\epsilon = 5$

Figure 17 shows the estimation result for each of six LDL levels when $\epsilon = 5$. For each level, the green bar represents the original proportion value, the red bar shows the estimated proportion value from the randomized response, and the blue bar shows the es-

timated proportion value from the Laplace mechanism. For each estimate, we also report its standard deviation. We can easily observe that the randomized response achieves better utility preservation for each level (with more accurate estimate and smaller standard deviation) than the Laplace mechanism.

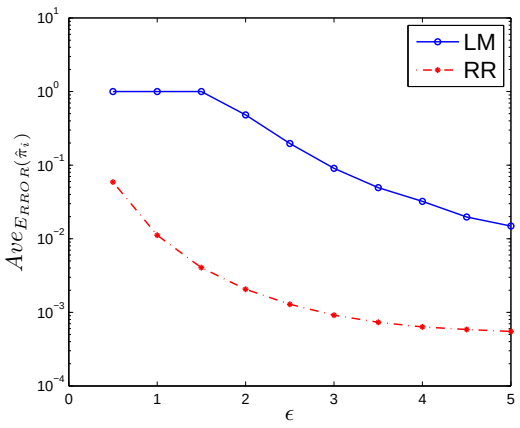


Figure 18: Average mean squared error for estimation of the distribution of the six different LDL levels vs. varying ϵ

Figure 18 shows the estimation results in terms of the average mean squared error of proportion estimates of the six different levels between the randomized response and the Laplace mechanism given varying ϵ values. We can easily observe the averaged estimation error of the randomized response is two-three orders lower than that of the Laplace mechanism and the randomized response shows more superiority than the Laplace mechanism when ϵ is small.

7.6.3 Derived Measures

We calculate the estimates of the entropy of the LDL. Figure 19 shows the estimation results of the calculated entropy values from two approaches with varying ϵ. We can see that the red line (corresponding to the randomized response) is more close to the green line (corresponding to the real entropy value) than the blue line (corresponding to the Laplace

mechanism). The bar values (corresponding to their standard deviation values) also clearly show the superiority of the randomize response.

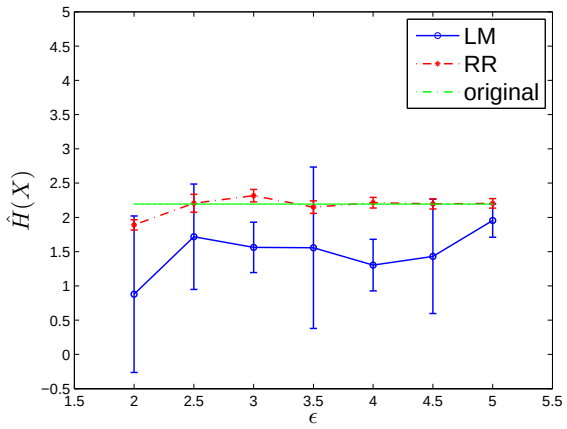


Figure 19: Estimation of the entropy of LDL vs. varying ϵ

We calculate the estimates of the chi-square statistics between the LDL and TC. Figure 20 shows the estimation results of the χ^2 statistics from two approaches with varying ϵ. We can see that the red line (corresponding to the randomized response) generally lies more close to the green line (corresponding to the real entropy value) than the blue line (corresponding to the Laplace mechanism) with varying ϵ values. The randomized response also has much smaller standard deviation values than the Laplace mechanism, which also indicates better utility preservation.

7.6.4 Graph Statistics

The graph of the YesiWell social network contains 185 nodes and 684 edges. In the data collection scenario, the untrusted server collects the link relationship information from users. The link relationship between two users is sensitive and should be protected. The collected social network data with n users and m relationships can be represented as an adjacency matrix $A_{n\times n}$ with $2m$ non-zero entries where $A_{ij} = 1$ denotes the presence

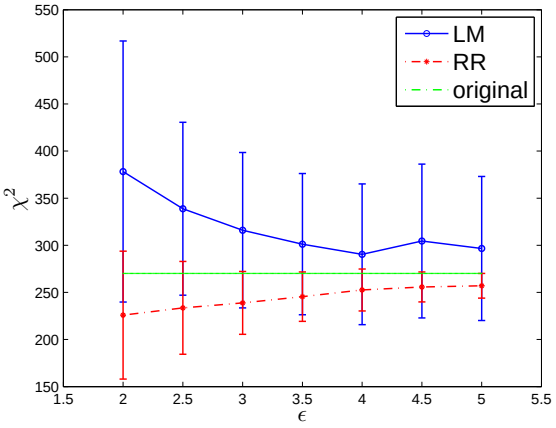


Figure 20: Estimation of chi-square statistics between LDL and TC vs. varying ϵ

of an relationship between user i and user j, and $A_{ij} = 0$ otherwise. In our setting, for A_{ij}, the client C_i applies the randomzied response (or the Laplace mechanism) to send the server a randomized output $Y_{ij} \in 0, 1$. After collecting all randomized relationships, the server then applies the reconstruction process and generates one instance of the social network with $2m$ non-zero entries (denoted as $\hat{A}$). The generated graph instance satisfies ϵ differential privacy and can be released for any analysis. In this experiment, we conduct performance comparison between the randomzied response and the Laplace mechanism using two graph features, the degree sequence $D = \{d_i\}$ and the number of triangle sequence $N_\Delta = \{N_\Delta(i)\}$. Figure 21 and Figure 22 show comparison results in terms of the degree sequence and the number of triangle sequence respectively.

Figure 21 shows the average entrywise error of the degree sequence calculated from different approaches with varying ϵ. In the figure, we denote the Laplace mechanism as LM and the randomzied response as RR, each of which uses its randomized graph topology respectively. We also report the comparison with the output perturbation method, LM-global, which adds the Laplace noise directly to the query output. Note that LM-global

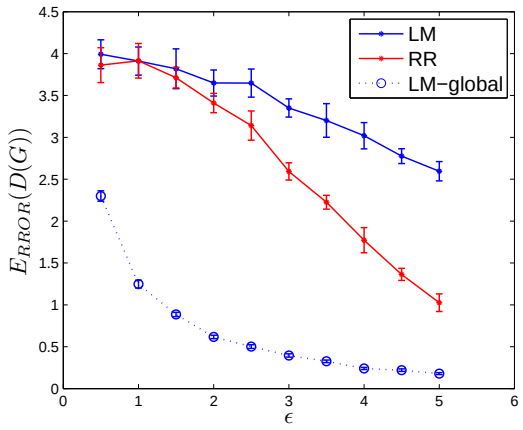


Figure 21: The average entrywise error of the degree sequence vs. varying ϵ

is used in the data query answering scenario where the server is assumed to have all the true unperturbed data. However, any differential privacy preservation query consumes a separate privacy budget. On the contrary, the randomized data collected from LM and RR can be released for any analysis with the same privacy threshold. We can observe in Figure 21 that the randomized response achieves better utility preservation than the Laplace method in the data collection scenario and the LM-global incurs less estimation error than both RR and LM (due to its small global sensitivity value $GS_D = 2$).

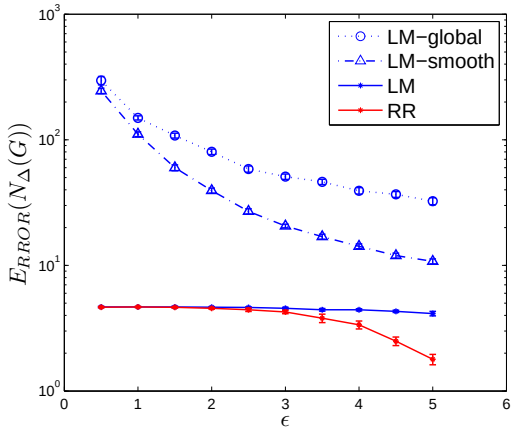


Figure 22: The average entrywise error of the number of triangle sequence vs. varying ϵ; LM-global denotes the global sensitivity based Laplace mechanism and LM-smooth denotes the smooth sensitivity based mechanism.

Figure 22 shows the average entrywise error of the number of triangle sequence calculated from different approaches with varying ϵ. Note that the global sensitivity of N_Δ is $3(n-2)$. We denote the approach of directly adding the Laplace noise based on the global sensitivity as LM-global. In Chapter 2, we presented an approach to adding Laplace noise based on the the smooth sensitivity [90], which can significantly reduce the magnitude of the injected noise. We denote the approach of adding the Laplace noise based on the smooth sensitivity as LM-smooth. As above, we denote the Laplace mechanism in our data collection scenario as LM and the randomized response as RR. We can observe that the average entrywise error of RR and LM is lower than that either of LM-global or LM-smooth, indicating the local differential privacy preserving data collection could be a better choice than output perturbation for queries or analysis with very large sensitive values. It is unsurprise that RR achieves the best utility preservation.

7.7 Related Work

Randomized response techniques have been extensively investigated in statistics (e.g., see a book [16]). Previous work on privacy preservation using the randomized response model mainly focused on evaluating the trade-off between privacy preservation and utility loss of the reconstructed data (e.g., [2, 46, 47, 94]). For example, the authors in [2, 94] focused on measuring privacy from the attacker's view when the distorted records of individuals and distortion parameters are available. The authors in [46, 47] studied utility preservation of various statistics (e.g., correlation) and data mining tasks with the unknown distortion parameters.

Some research studied the problem of determining the optimal distortion parameters to achieve good performance (e.g., [45, 59]). Specifically, the authors in [45] studied how to

choose distortion parameters such that certain chosen marginal distributions in the original data are left invariant in expectation of the randomized data. The authors in [59] developed a heuristic method of searching of optimal distortion parameters to balance privacy and utility.

The authors in [36] first presented the notation of privacy breaches based on *amplification* where it provides guarantee limits on privacy breaches without any knowledge of the distribution of the original data. Local differential privacy was formally proposed in [28,62] as a strong measure of privacy under the data collection scenario, where individual clients are willing to share their data but are concerned about revealing sensitive information. The authors studied the problem of utility maximization under local differential privacy and developed a family of extremal mechanisms called the staircase mechanisms and showed that two simple staircase mechanisms (the binary and randomized response mechanisms) are optimal in the high and low privacy regimes. In [62], the author mainly studies the tradeoff between local privacy and utility in hypothesis testing. In [28], the authors studied the tradeoff between privacy guarantees and the utility of mean estimation in location family model and convex risk minimization. Most recently, the authors introduced Randomized Aggregatable Privacy-Preserving Ordinal Response, or RAPPOR [35] for handling multiple data collections from the same client by providing longitudinal differential privacy guarantees. In this chapter, we studied the commonly used calculations including individual value estimate, proportion estimate, and the estimates of various derived statistics with multiple polychotomous attributes. We derived the explicit formula of the mean squared error of those estimates and prove that the randomized response outperforms the Laplace mechanism which is more frequently applied for privacy preservation data mining. Espe-

cially, our evaluation also indicates that for social network mining tasks that have large function sensitivity, our proposed randomized response strategy can incur much less utility loss than the traditionally used output perturbation based on Laplace mechanism with either global sensitivity or smooth sensitivity does.

7.8 Summary

In this chapter we study local privacy preservation where an untrusted server wishes to learn statistics or publish the collected client data while guaranteeing the privacy of each contributing individual. In this case, the privacy is ensured by each client individually without a need for a trusted server. The client does not have sufficient information about the data distribution which is required in the biased sampling of the exponential mechanism. The use of the randomize response is a feasible option. Moreover, the randomized response enables easy estimations of accurate population statistics while preserving the privacy of the individual respondents. We have conducted theoretical analysis and empirical evaluations to show that the developed randomize response significantly outperforms the approach of directly adding the Laplace noise in the data collection scenario in terms of utility preservation. This result is especially promising for data mining or exploration tasks with interactive processes, in which a user can adaptively query the system about the data. The user now has options of using the released data rather than submitting to the server new queries that incur further consumption of privacy budget. Technical report version of this work can be found in [108].

CHAPTER 8: CONCLUSIONS AND FUTURE WORK

Enabling accurate analysis of sensitive data while preserving differential privacy is of great importance. Great challenges have posed due to potential high global sensitivity of various mining functions, unintended disclosure risk from released aggregate statistics or data models and even the lack of trusted third party data collection and storage platform. In this dissertation, we have developed several techniques to meet such challenges.

First, we have presented a divide and conquer approach that can be used to enforce differential privacy for calculating complex mining functions. We have conducted theoretical analysis and extensive empirical evaluations to show that the developed divide and conquer approach generally outperforms the approach of directly enforcing differential privacy in terms of utility preservation. This result is especially promising for data mining or exploration tasks with interactive processes, in which a user can adaptively query the system about the data. The user now has options of reusing previous intermediate query results rather than submitting to the system "new" queries that can be expressed by previous ones. There are some other aspects of this work that merit further research. Among them, we will continue the line of this research by investigating how to enforce differential privacy for other complex functions or social network analysis tasks. For functions that we cannot compute the smooth sensitivity efficiently or explicitly, Nissim et al. proposed an approximation method that computes the β-smooth upper bound on the local sensitivity of these functions and developed a sample-aggregation framework for a large class of

functions [90]. We will evaluate those functions based on the sample-aggregation framework. We will exploit the use of correlations among unit computations to further reduce noise and enhance accuracies of computation outputs. Our goal is to identify (optimal) decomposition strategies and (optimal) budget privacy distribution.

We have then presented private dK-graph generation models that enforce rigorous differential privacy while preserving utility. We have conducted theoretical analysis and empirical evaluations to show that the developed private dK-graph generation models significantly outperform the approach based on the stochastic Kronecker generation model. We have shown that the DP-2K graph model generally achieves better utility preservation than the DP-1K graph model with weak privacy enforcement whereas the DP-1K graph model would achieve better utility preservation with strong privacy enforcement. We will study non-interactive graph data release mechanisms, i.e., we use the derived differentially private graph statistics to generate synthetic graphs for release. We are interested in how to preserve some known graph metrics (e.g., modularity) in addition to degree correlations in the dK-graph generation process. We will continue the line of this research by investigating how to enforce differential privacy on other graph generation models (e.g., the class of exponential random graph models [4]) and comparing various models in terms of the tradeoff between utility and privacy. We will also conduct more evaluations by evaluating more metrics and using large social networks.

We have also presented two approaches to enforcing differential privacy in spectral graph analysis. We have applied and evaluated the Laplace Mechanism [32] and the exponential mechanism [86] on the differential privacy preserving eigen decomposition on the graph topology. In our future work, we will investigate how to enforce differential privacy for

other spectral graph analysis tasks (e.g., spectral clustering based on graph's Laplacian and normal matrices). Nissim et al. [90] introduced a framework that calibrates the instance-specific noise with smaller magnitude than the worst-case noise based on the global sensitivity. We will study the use of smooth sensitivity and explore how to better distribute privacy budget in the proposed *LNPP* approach. We will also study how different sampling strategies in the proposed *SBMF* approach may affect the utility preservation.

Next, we have studied whether and to what extent GWAS statistics can be exploited by an attacker to learn private information of general population, not limited to GWAS participants. We have developed two potential attacks, *trait inference attack* and *identity inference attack*. Both attacks exploit the released GWAS statistics about the associations between SNP genotypes and human traits. Our evaluations have shown that the proposed attacks have made re-identification of anonymized genotype data a real threat. In our future work, we will study how to extend our two-layered bayesian network to capture trait-trait associations and/or SNP-SNP correlations. We will study how to formalize various types of background knowledge that an attacker may have in practice and evaluate how well data perturbation and agglomeration techniques with background knowledge [19, 27, 82] can protect privacy when releasing GWAS statistics. Our goal is to develop methods to enable researchers to safely release aggregate GWAS data without compromising the anonymity of both study participants and non-participants.

We then have developed an effective approach which simultaneously protects differential privacy of participants and prevents sensitive attribute disclosure of regular individuals due to model inversion attacks while retaining the efficacy of released regression models. Leveraging the functional mechanism [131], our approach rewrites the objective function

in its polynomial representation and adds more (less) noise to the polynomial coefficients with (w/o) sensitive attributes. Our approach can effectively weaken the correlation between the sensitive attributes with the output to prevent model inversion attacks whereas retaining the utility of the released model by decreasing the perturbation effect on non-sensitive attributes. As a result, we still achieve ϵ-differential privacy for participants. In our future work, we will evaluate our research on real world applications such as clinical study which involves genetic privacy. We will theoretically analyze applicability of model inversion attacks under different background knowledge. We will explore optimal perturbation strategies, e.g., what perturbations could incur least potential utility loss in terms of accuracy of analysis output under differential privacy and model inversion attacks.

The last but not the least, we have studied local privacy preservation where an untrusted server wishes to learn statistics or publish the collected client data while guaranteeing the privacy of each contributing individual. In this case, the privacy is ensured by each client individually without a need for a trusted server. The client does not have sufficient information about the data distribution which is required in the biased sampling of the exponential mechanism. The use of the randomize response is a feasible option. Moreover, the randomized response enables easy estimations of accurate population statistics while preserving the privacy of the individual respondents. We have conducted theoretical analysis and empirical evaluations to show that the developed randomize response significantly outperforms the approach of directly adding the Laplace noise in the data collection scenario in terms of utility preservation. This result is especially promising for data mining or exploration tasks with interactive processes, in which a user can adaptively query the system about the data. The user now has options of using the released data rather than

submitting to the server new queries that incur further consumption of privacy budget. For future work, we will continue the line of investigating how to enforce differential privacy for other complex functions or social network analysis tasks. We will study the potential use of randomized response on numerical data. In this dissertation, we measure the utility preservation in terms of the variance. Several theoretical works on the privacy mechanism design (e.g., [28]) proposed the use of a general utility-maximization framework under differential privacy where the utility function can be a general function depending on the noise added to the query output. We will explore the use of the general function to measure the utility.

Generally speaking, there are three aspects of this dissertation that merit future research. First, we will consider more complex data analysis tasks involving more algorithms with the goal of maximizing the accuracy of analysis result with acceptable privacy budget. Second, we will extend and incorporate our developed approaches in those data analysis systems containing data with large 'Volume' and 'Velocity' except for large 'Variety' which is the current focus of this dissertation. Third, we will explore different privacy protection requirements and techniques for different areas in the future environment of big data analysis.

REFERENCES

[1] L. A. Adamic and N. Glance. The political blogosphere and the 2004 us election. In *WWW-2005 Workshop on the Weblogging Ecosystem*, 2005.

[2] S. Agrawal and J. R. Haritsa. A framework for high-accuracy privacy-preserving mining. In *Data Engineering, 2005. ICDE 2005. Proceedings. 21st International Conference on*, pages 193–204. IEEE, 2005.

[3] W. Aiello, F. Chung, and L. Lu. A random graph model for massive graphs. In *Proceedings of the thirty-second annual ACM symposium on Theory of computing*, pages 171–180. Acm, 2000.

[4] C. J. Anderson, S. Wasserman, and B. Crouch. A p* primer: Logit models for social networks. *Social networks*, 21(1):37–66, 1999.

[5] L. Backstrom, C. Dwork, and J. Kleinberg. Wherefore art thou r3579x?: anonymized social networks, hidden patterns, and structural steganography. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 181–190, New York, NY, USA, 2007. ACM Press.

[6] L. Backstrom, D. Huttenlocher, J. Kleinberg, and X. Lan. Group formation in large social networks: membership, growth, and evolution. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 44–54. ACM, 2006.

[7] A. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

[8] B. Barak, K. Chaudhuri, C. Dwork, S. Kale, F. McSherry, and K. Talwar. Privacy, accuracy, and consistency too: a holistic solution to contingency table release. In *Proceedings of the twenty-sixth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pages 273–282. ACM, 2007.

[9] M. Belkin and P. Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. *Advances in neural information processing systems*, 1:585–592, 2002.

[10] T. Y. Berger-Wolf and J. Saia. A framework for analysis of dynamic social networks. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 523–528. ACM, 2006.

[11] R. Bhaskar, S. Laxman, A. Smith, and A. Thakurta. Discovering frequent patterns in sensitive data. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 503–512. ACM, 2010.

[12] A. Blum, C. Dwork, F. McSherry, and K. Nissim. Practical privacy: the SuLQ framework. In *Proceedings of the twenty-fourth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pages 128–138. ACM, 2005.

[13] A. Blum, K. Ligett, and A. Roth. A learning theory approach to non-interactive database privacy. In *Proceedings of the 40th annual ACM symposium on Theory of computing*, pages 609–618. ACM, 2008.

[14] M. Boguñá and R. Pastor-Satorras. Class of correlated random networks with hidden variables. *Physical Review E*, 68(3):036112, 2003.

[15] P. K. Chan, M. D. F. Schlag, and J. Y. Zien. Spectral k-way ratio-cut partitioning and clustering. In *Proceedings of the 30th international Design Automation Conference*, pages 749–754. ACM, 1993.

[16] A. Chaudhuri and R. Mukerjee. *Randomized response: Theory and techniques*. Marcel Dekker New York, 1988.

[17] K. Chaudhuri and C. Monteleoni. Privacy-preserving logistic regression. In *Proceedings of the Twenty-Second Annual Conference on Neural Information Processing Systems (NIPS)*, pages 289–296. Citeseer, 2008.

[18] K. Chaudhuri, A. Sarwate, and K. Sinha. Near-optimal algorithms for differentially-private principal components. *arXiv preprint arXiv:1207.2812*, 2012.

[19] B.-C. Chen, K. LeFevre, and R. Ramakrishnan. Privacy skyline: Privacy with multidimensional adversarial knowledge. In *Proceedings of the 33rd international conference on Very large data bases*, pages 770–781. VLDB Endowment, 2007.

[20] F. Chung. *Spectral graph theory*. Number 92. Amer Mathematical Society, 1997.

[21] F. Chung and L. Lu. Connected components in random graphs with given expected degree sequences. *Annals of combinatorics*, 6(2):125–145, 2002.

[22] G. Cormode. Personal privacy vs population privacy: learning to attack anonymization. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1253–1261. ACM, 2011.

[23] L. Costa, F. Rodrigues, G. Travieso, and P. Boas. Characterization of complex networks: A survey of measurements. *Advances in Physics*, 56(1):167–242, 2007.

[24] L. da F. Costa, F. A. Rodrigues, G. Travieso, and P. R. V. Boas. Characterization of complex networks: A survey of measurements. *Advances In Physics*, 56:167, 2007.

[25] B. Ding, M. Winslett, J. Han, and Z. Li. Differentially private data cubes: optimizing noise sources and consistency. In *SIGMOD Conference*, pages 217–228, 2011.

[26] C. Ding, X. He, H. Zha, M. Gu, and H. Simon. A min-max cut algorithm for graph partitioning and data clustering. In *ICDM*, 2001.

[27] W. Du, Z. Teng, and Z. Zhu. Privacy-MaxEnt: integrating background knowledge in privacy quantification. In *ACM SIGMOD 2008*.

[28] J. C. Duchi, M. I. Jordan, and M. J. Wainwright. Local privacy and statistical minimax rates. In *Foundations of Computer Science (FOCS), 2013 IEEE 54th Annual Symposium on*, pages 429–438. IEEE, 2013.

[29] C. Dwork. A firm foundation for private data analysis. *Communications of the ACM*, 54(1):86–95, 2011.

[30] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor. Our data, ourselves: Privacy via distributed noise generation. *Advances in Cryptology-EUROCRYPT 2006*, pages 486–503, 2006.

[31] C. Dwork and J. Lei. Differential privacy and robust statistics. In *Proceedings of the 41st annual ACM symposium on Theory of computing*, pages 371–380. ACM, 2009.

[32] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. *Theory of Cryptography*, pages 265–284, 2006.

[33] C. Dwork and A. Smith. Differential privacy for statistics: What we know and what we want to learn. *Journal of Privacy and Confidentiality*, 1(2):2, 2010.

[34] P. ERDdS and A. R&WI. On random graphs i. *Publ. Math. Debrecen*, 6:290–297, 1959.

[35] Ú. Erlingsson, V. Pihur, and A. Korolova. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 1054–1067. ACM, 2014.

[36] A. Evfimievski, J. Gehrke, and R. Srikant. Limiting privacy breaches in privacy preserving data mining. In *Proceedings of the twenty-second ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 211–222. ACM, 2003.

[37] S. E. Fienberg, A. Slavkovic, and C. Uhler. Privacy preserving gwas data sharing. In *Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on*, pages 628–635. IEEE, 2011.

[38] M. Fredrikson, E. Lantz, S. Jha, S. Lin, D. Page, and T. Ristenpart. Privacy in pharmacogenetics: an end-to-end case study of personalized warfarin dosing. *Mortality*, 1(1.15):1–20, 2014.

[39] A. Friedman and A. Schuster. Data mining with differential privacy. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 493–502. ACM, 2010.

[40] S. Ganta, S. Kasiviswanathan, and A. Smith. Composition attacks and auxiliary information in data privacy. In *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 265–273. ACM, 2008.

[41] P. Garthwaite, F. Critchley, K. Anaya-Izquierdo, and E. Mubwandarikwa. Orthogonalization of vectors with minimal adjustment. *Biometrika*, 2012.

[42] A. Ghosh, T. Roughgarden, and M. Sundararajan. Universally utility-maximizing privacy mechanisms. In *Proceedings of the 41st annual ACM symposium on Theory of computing*, pages 351–360. ACM, 2009.

[43] M. Girvan and M. E. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, 2002.

[44] D. Gleich and A. Owen. Moment based estimation of stochastic kronecker graph parameters. *Arxiv preprint arXiv:1106.1674*, 2011.

[45] J. M. Gouweleeuw, P. Kooiman, L. C. R. J. Willenborg, and P. P. de Wolf. Post randomization for statistical disclosure control: theory and implementation. *Journal of Official Statistics*, 14(4):463–478, 1998.

[46] L. Guo, S. Guo, and X. Wu. Privacy preserving market basket data analysis. In *Knowledge Discovery in Databases: PKDD 2007*, pages 103–114. Springer, 2007.

[47] L. Guo and X. Wu. Privacy preserving categorical data analysis with unknown distortion parameters. *Transactions on Data Privacy*, 2(3):185–205, 2009.

[48] M. Gymrek, A. L. McGuire, D. Golan, E. Halperin, and Y. Erlich. Identifying personal genomes by surname inference. *Science*, 339(6117):321–324, 2013.

[49] L. Hagen and B. Kahng. New spectral methods for ratio cut partitioning and clustering. *IEEE Transactions on computer-aided design*, 11(9), 1992.

[50] M. Hardt and K. Talwar. On the geometry of differential privacy. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 705–714. ACM, 2010.

[51] M. Hay, C. Li, G. Miklau, and D. Jensen. Accurate estimation of the degree distribution of private networks. In *Data Mining, 2009. ICDM'09. Ninth IEEE International Conference on*, pages 169–178. IEEE, 2009.

[52] M. Hay, G. Miklau, D. Jensen, D. Towsley, and P. Weis. Resisting structural reidentification in anonymized social networks. *Proceedings of the VLDB Endowment archive*, 1(1):102–114, 2008.

[53] M. Hay, V. Rastogi, G. Miklau, and D. Suciu. Boosting the Accuracy of Differentially Private Histograms Through Consistency. *Proceedings of the VLDB Endowment*, 3(1), 2010.

[54] L. Hindorff, J. MacArthur, J. Morales, H. Junkins, P. Hall, A. Klemm, and T. Manolio. A catalog of published genome-wide association studies.

[55] P. Hoff. Simulation of the Matrix Bingham-von Mises-Fisher Distribution, With Applications to Multivariate and Relational Data. *Journal of Computational and Graphical Statistics*, 18(2):438–456, 2009.

[56] N. Homer, S. Szelinger, M. Redman, D. Duggan, W. Tembe, J. Muehling, J. V. Pearson, D. A. Stephan, S. F. Nelson, and D. W. Craig. Resolving individuals contributing trace amounts of dna to highly complex mixtures using high-density snp genotyping microarrays. *PLoS genetics*, 4(8):e1000167, 2008.

[57] X. Hu, A. Lu, and X. Wu. Spectrum-based network visualization for topology analysis. *IEEE Computer Graphics and Applications*, 33(1):58–68, 2013.

[58] L. Huang, D. Yan, M. Jordan, and N. Taft. Spectral clustering with perturbed data. *Advances in Neural Information Processing Systems (NIPS)*, 2008.

[59] Z. Huang and W. Du. Optrr: Optimizing randomized response schemes for privacy-preserving data mining. In *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on*, pages 705–714. IEEE, 2008.

[60] J.Leskovec. Snap: Stanford network analysis platform.

[61] A. Johnson and V. Shmatikov. Privacy-preserving data exploration in genome-wide association studies. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1079–1087. ACM, 2013.

[62] P. Kairouz, S. Oh, and P. Viswanath. Extremal mechanisms for local differential privacy. *arXiv preprint arXiv:1407.1338*, 2014.

[63] D. Kempe, J. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 137–146. ACM, 2003.

[64] M. G. Kendall and A. Stuart. The advanced theory of statistics, vol. 2: Hafner. *New York*, page 133, 1969.

[65] D. Kifer and A. Machanavajjhala. No free lunch in data privacy. In *SIGMOD Conference*, pages 193–204, 2011.

[66] J. M. Kleinberg. Challenges in mining social network data: processes, privacy, and paradoxes. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 4–5. ACM, 2007.

[67] Y. Koren, S. C. North, and C. Volinsky. Measuring and extracting proximity in networks. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 245–255. ACM, 2006.

[68] M. Kosinski, D. Stillwell, and T. Graepel. Private traits and attributes are predictable from digital records of human behavior. *Proceedings of the National Academy of Sciences*, 110(15):5802–5805, 2013.

[69] R. Kumar, J. Novak, and A. Tomkins. Structure and evolution of online social networks. In *Link mining: models, algorithms, and applications*, pages 337–357. Springer, 2010.

[70] J. Kunegis, S. Schmidt, A. Lommatzsch, J. Lerner, E. W. D. Luca, and S. Albayrak. Spectral analysis of signed graphs for clustering, prediction and visualization. In *SDM*, pages 559–, 2010.

[71] J. Lee and C. Clifton. Differential identifiability. In *KDD*, pages 1041–1049, 2012.

[72] J. Leskovec, D. Chakrabarti, J. Kleinberg, C. Faloutsos, and Z. Ghahramani. Kronecker graphs: An approach to modeling networks. *The Journal of Machine Learning Research*, 11:985–1042, 2010.

[73] J. Leskovec and C. Faloutsos. Scalable modeling of real graphs using kronecker multiplication. In *Proceedings of the 24th international conference on Machine learning*, pages 497–504. ACM, 2007.

[74] C. Li, M. Hay, V. Rastogi, G. Miklau, and A. McGregor. Optimizing linear counting queries under differential privacy. In *Proceedings of the twenty-ninth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 123–134. ACM, 2010.

[75] L. Li, D. Alderson, W. Willinger, and J. Doyle. A first-principles approach to understanding the internet's router-level topology. In *ACM SIGCOMM Computer Communication Review*, volume 34, pages 3–14. ACM, 2004.

[76] N. Li, T. Li, and S. Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In *ICDE*, 2007.

[77] Z. Lin, A. B. Owen, and R. B. Altman. Genomic research and human subject privacy. *SCIENCE-NEW YORK THEN WASHINGTON-.*, pages 183–183, 2004.

[78] Z. Luo, Y. Wang, and X. Wu. Predicting retweeting behavior based on autoregressive moving average model. In *Web Information Systems Engineering-WISE 2012*, pages 777–782. Springer, 2012.

[79] Z. Luo, Y. Wang, X. Wu, W. Cai, and T. Chen. On burst detection and prediction in retweeting sequence. In *Advances in Knowledge Discovery and Data Mining*. Springer, 2015.

[80] A. Machanavajjhala, J. Gehrke, D. Keifer, and M. Venkitasubramanian. l-diversity: privacy beyond k-anonymity. In *Proceedings of the IEEE ICDE Conference*, 2006.

[81] P. Mahadevan, D. Krioukov, K. Fall, and A. Vahdat. Systematic topology analysis and generation using degree correlations. In *ACM SIGCOMM Computer Communication Review*, volume 36, pages 135–146. ACM, 2006.

[82] D. J. Martin, D. Kifer, A. Machanavajjhala, J. Gehrke, and J. Y. Halpern. Worst-case background knowledge for privacy-preserving data publishing. In *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on*, pages 126–135. IEEE, 2007.

[83] N. Masca, P. R. Burton, and N. A. Sheehan. Participant identification in genetic association studies: improved methods and practical implications. *International journal of epidemiology*, 40(6):1629–1642, 2011.

[84] F. McSherry. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In *Proceedings of the 35th SIGMOD international conference on Management of data*, pages 19–30. ACM, 2009.

[85] F. McSherry and I. Mironov. Differentially Private Recommender Systems. In *kdd*. ACM, 2009.

[86] F. McSherry and K. Talwar. Mechanism design via differential privacy. In *Foundations of Computer Science, 2007. FOCS'07. 48th Annual IEEE Symposium on*, pages 94–103. IEEE, 2007.

[87] D. Mir and R. Wright. A differentially private graph estimator. In *Data Mining Workshops, 2009. ICDMW'09. IEEE International Conference on*, pages 122–129. IEEE, 2009.

[88] M. Molloy and B. Reed. A critical point for random graphs with a given degree sequence. *Random Structures & Algorithms*, 6(2-3):161–180, 1995.

[89] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems 14: Proceeding of the 2001 Conference*, pages 849–856, 2001.

[90] K. Nissim, S. Raskhodnikova, and A. Smith. Smooth sensitivity and sampling in private data analysis. In *Proceedings of the thirty-ninth annual ACM Symposium on Theory of Computing*, pages 75–84. ACM, 2007.

[91] A. Pothen, H. Simon, and K. Liou. Partitioning sparse matrices with eigenvectors of graphs. *SIAM Journal on Matrix Analysis and Applications*, 11:430, 1990.

[92] B. Prakash, A. Sridharan, M. Seshadri, S. Machiraju, and C. Faloutsos. EigenSpokes: Surprising Patterns and Scalable Community Chipping in Large Graphs. *Advances in Knowledge Discovery and Data Mining*, pages 435–448, 2010.

[93] V. Rastogi, M. Hay, G. Miklau, and D. Suciu. Relationship privacy: Output perturbation for queries with joins. In *Proceedings of the twenty-eighth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 107–116. ACM, 2009.

[94] S. J. Rizvi and J. R. Haritsa. Maintaining data privacy in association rule mining. In *Proceedings of the 28th international conference on Very Large Data Bases*, pages 682–693. VLDB Endowment, 2002.

[95] A. Sala, X. Zhao, C. Wilson, H. Zheng, and B. Zhao. Sharing graphs using differentially private graph models. In *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*, pages 81–98. ACM, 2011.

[96] P. Samarati and L. Sweeney. Protecting privacy when disclosing information: k-anonymty and its enforcement through generalization and suppression. In *Proceedings of the IEEE Symposium on Research in Security and Privacy*, 1998.

[97] R. Sarathy and K. Muralidhar. Differential Privacy for Numeric Data. *Joint UNECE/Eurostat work session on statistical data confidentiality, Bilbao, Spain*, 2009.

[98] A. J. Seary and W. D. Richards. *Spectral methods for analyzing and visualizing networks: an introduction*. na, 2003.

[99] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000.

[100] E. Spertus, M. Sahami, and O. Buyukkokten. Evaluating similarity measures: a large-scale study in the orkut social network. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 678–684. ACM, 2005.

[101] G. Stewart and J. Sun. *Matrix perturbation theory*, volume 175. Academic press New York, 1990.

[102] H. Tangmunarunkit, R. Govindan, S. Jamin, S. Shenker, and W. Willinger. Network topology generators: Degree-based vs. structural. In *ACM SIGCOMM Computer Communication Review*, volume 32, pages 147–159. ACM, 2002.

[103] C. Tantipathananandh, T. Berger-Wolf, and D. Kempe. A framework for community identification in dynamic social networks. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 717–726. ACM, 2007.

[104] The 1000 Genomes Project Consortium. An integrated map of genetic variation from 1,092 human genomes. *Nature*, 491:1, 2012.

[105] R. Wang, Y. F. Li, X. Wang, H. Tang, and X. Zhou. Learning your identity and disease from research papers: information leaks in genome wide association study. In *Proceedings of the 16th ACM conference on Computer and communications security*, pages 534–544. ACM, 2009.

[106] Y. Wang, C. Si, and X. Wu. Regression model fitting under differential privacy and model inversion attacks. *Technical Report DPL-2015-001 University of Arkansas*, 2015.

[107] Y. Wang and X. Wu. Preserving differential privacy in degree-correlation based graph generation. *Transactions on data privacy*, 6(2):127, 2013.

[108] Y. Wang, X. Wu, and D. Hu. Using randomized response for differential privacy preserving data collection. *Technical Report DPL-2014-003 University of Arkansas*, 2014.

[109] Y. Wang, X. Wu, and X. Shi. Using aggregate human genome data for individual identification. In *Proceedings of the IEEE International Conference on Bioinformatics and Biomedicine*, pages 410–415. 2013.

[110] Y. Wang, X. Wu, and X. Shi. Infringement of individual privacy via mining gwas statistics. *Technical Report DPL-2014-004 University of Arkansas*, 2014.

[111] Y. Wang, X. Wu, and L. Wu. Differential privacy preserving spectral graph analysis. In *PAKDD (2)*, pages 329–340, 2013.

[112] Y. Wang, X. Wu, J. Zhu, and Y. Xiang. On learning cluster coefficient of private networks. In *2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 395–402. IEEE, 2012.

[113] Y. Wang, X. Wu, J. Zhu, and Y. Xiang. On learning cluster coefficient of private networks. *Social network analysis and mining*, 3(4):925–938, 2013.

[114] D. Watts and S. Strogatz. Collective dynamics of small-worldnetworks. *Nature*, 393(6684):440–442, 1998.

[115] D. Watts and S. Strogatz. The small world problem. *Collective Dynamics of Small-World Networks*, 393:440–442, 1998.

[116] S. White and P. Smyth. Algorithms for estimating relative importance in networks. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 266–275. ACM, 2003.

[117] L. Wu, X. Wu, A. Lu, and Y. Li. On spectral analysis of signed and dispute graphs. In *2014 IEEE International Conference on Data Mining, ICDM 2014, Shenzhen, China, December 14-17, 2014*, pages 1049–1054, 2014.

[118] L. Wu, X. Wu, A. Lu, and Z.-H. Zhou. A spectral approach to detecting subtle anomalies in graphs. *Journal of Intelligent Information Systems*, 41(2):313–337, 2013.

[119] L. Wu, X. Ying, X. Wu, A. Lu, and Z.-H. Zhou. Spectral analysis of k-balanced signed graphs. In *PAKDD (2)*, pages 1–12, 2011.

[120] L. Wu, X. Ying, X. Wu, A. Lu, and Z.-H. Zhou. Examining spectral space of complex networks with positive and negative links. *IJSNM*, 1(1):91–111, 2012.

[121] L. Wu, X. Ying, X. Wu, and Z.-H. Zhou. Line orthogonality in adjacency eigenspace with application to community partition. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, volume 22, page 2349, 2011.

[122] X. Wu, X. Ying, K. Liu, and L. Chen. A survey of privacy-preservation of graphs and social networks. In *Managing and mining graph data*, pages 421–453. Springer, 2010.

[123] X. Xiao, G. Bender, M. Hay, and J. Gehrke. ireduct: differential privacy with reduced relative errors. In *SIGMOD Conference*, pages 229–240, 2011.

[124] X. Xiao, G. Wang, and J. Gehrke. Differential privacy via wavelet transforms. In *Data Engineering (ICDE), 2010 IEEE 26th International Conference on*, pages 225–236. IEEE, 2010.

[125] X. Ying, L. Wu, and X. Wu. A spectrum-based framework for quantifying randomness of social networks. *Knowledge and Data Engineering, IEEE Transactions on*, 23(12):1842–1856, 2011.

[126] X. Ying and X. Wu. Graph generation with prescribed feature constraints. In *SDM*, 2009.

[127] X. Ying and X. Wu. On randomness measures for social networks. In *SDM*, 2009.

[128] X. Ying and X. Wu. On link privacy in randomizing social networks. *Knowledge and information systems*, 28(3):645–663, 2011.

[129] X. Ying, X. Wu, and D. Barbará. Spectrum based fraud detection in social networks. In *Data Engineering (ICDE), 2011 IEEE 27th International Conference on*, pages 912–923. IEEE, 2011.

[130] X. Ying, X. Wu, and Y. Wang. On linear refinement of differential privacy-preserving query answering. In *PAKDD (2)*, pages 353–364, 2013.

[131] J. Zhang, Z. Zhang, X. Xiao, Y. Yang, and M. Winslett. Functional mechanism: regression analysis under differential privacy. *Proceedings of the VLDB Endowment*, 5(11):1364–1375, 2012.

[132] X. Zhou, B. Peng, Y. F. Li, Y. Chen, H. Tang, and X. Wang. To release or not to release: evaluating information leaks in aggregate human-genome data. In *Computer Security–ESORICS 2011*, pages 607–627. Springer, New York, 2011.