Compressive Mechanism: Utilizing Sparse Representation in Differential Privacy

Yang D. Li[1]   Zhenjie Zhang[1]   Marianne Winslett[1,2]   Yin Yang[1]

[1]Advanced Digital Sciences Center
Illinois at Singapore Pte.
{daniel.li, zhenjie, yin.yang}@adsc.com.sg

[2]Department of Computer Science
University of Illinois at Urbana-Champaign
winslett@illinois.edu

Abstract

Differential privacy provides the first theoretical foundation with provable privacy guarantee against adversaries with arbitrary prior knowledge. The main idea to achieve differential privacy is to inject random noise into statistical query results. Besides correctness, the most important goal in the design of a differentially private mechanism is to reduce the effect of random noise, ensuring that the noisy results can still be useful.

This paper proposes the *compressive mechanism*, a novel solution on the basis of state-of-the-art compression technique, called *compressive sensing*. Compressive sensing is a decent theoretical tool for compact synopsis construction, using random projections. In this paper, we show that the amount of noise is significantly reduced from $O(\sqrt{n})$ to $O(\log(n))$, when the noise insertion procedure is carried on the synopsis samples instead of the original database. As an extension, we also apply the proposed compressive mechanism to solve the problem of continual release of statistical results. Extensive experiments using real datasets justify our accuracy claims.

1 Introduction

No one shall be subjected to arbitrary interference with his privacy, family, home or correspondence, nor to attacks upon his honor and reputation.

Universal Declaration of Human Rights

Rapid advances in information technology and computational capacity are raising concerns regarding the privacy of sensitive personal information. Previous work has shown that even after removing all personal identity attributes such as names and addresses, adversaries may still be able to identify specific individuals by combining their *prior knowledge* with the published attributes in the database, e.g., age, gender and race. For example, in 2007, a team of researchers successfully reidentified two customers in the "anonymized" Netflix data set [1], based on their transaction histories with Netflix and movie comments on IMDB [37]. Moreover, recent studies show that group statistics are also vulnerable to privacy attacks due to inference techniques combining anonymized data and existing and/or public information. In Genome Wide Association Studies (GWAS), for instance, the DNA samples from participating patients are mixed to prevent disclosure of their identities, and only statistics regarding the prevalence of particular single-nucleotide polymorphisms

(SNPs) are published. However, an adversary can still verify the presence of a specific person in the mixture with high confidence, given a DNA sample from that person and a reference DNA mixture [29, 45]. Such privacy problems have become a major obstacle to biomedical research, as they make access to useful input data increasingly difficult to obtain.

To tackle these problems, numerous privacy protection frameworks have been proposed. Among these, *differential privacy* outshines others in providing strong robustness guarantee against attacks from adversaries with arbitrary prior knowledge. Simply put, a randomized mechanism (query answering method) for answering statistical queries satisfies differential privacy if and only if the query results will be almost identical after modifying or deleting one of the records in the database [21]. This requirement ensures that sensitive information in a record almost cannot be inferred, even if the adversary knows the values of all the other records in the database.

To achieve differential privacy, one basic mechanism is to add random noise to the statistical query results [21]. In particular, given the unperturbed query result and a parameter ϵ called the *privacy budget*, the mechanism randomly selects a number following a *Laplacian distribution*, with mean at the original query result and scale proportional to the *sensitivity* (discussed later) of the queries times $\frac{1}{\epsilon}$. Theoretical analysis shows that the resulting randomness in the query result fulfills the requirements of differential privacy with respect to privacy budget ϵ. While this basic mechanism handles a single query quite well, it is not effective for processing a large number of queries, as it requires a privacy budget linear in the number of queries to maintain a given level of result accuracy. Thus queries may quickly exhaust the total privacy budget assigned to a database, forcing the database to be taken offline to avoid unacceptable privacy breaches. In such situations, a better alternative is to build a compact, privacy-preserving synopsis from the data with a fixed amount of privacy budget, such that it is capable of answering all possible queries.

There have been several previous efforts to design and construct such synopses, e.g., wavelets [46], trees [28] and linear summation basis [30, 23, 12]. These synopsis structures are *lossless*, meaning that they preserve all information in the original database. In other words, the original statistics can be completely and accurately recovered by running the corresponding decoding algorithms on the synopsis structures. Consequently, the size of these synopses, as well as the privacy budget they require, grow linearly with the size of the dataset.

This paper explores a new direction: probabilistic synopses based on *compressive sensing* [17, 8, 10]. Figure 1 illustrates this new compressive sensing mechanism. Using a *sparse representation* of the original data, we use compressive sensing to encode a very small synopsis, compared to the original database size. We then add Laplacian noise to the synopsis, making it differentially private; decode the synopsis, creating a noisy version of the original data; then answer an unlimited number of queries over the decoded data, without adding additional noise. The compressive sensing mechanism allows us to use less noise than previous synopsis proposals under certain conditions, and provides much more accurate statistical query results after decoding. Unlike previous methods that focus on specific classes of queries, the compressive sensing mechanism is *universal*, supporting all possible queries on the decoded noisy data. Thus the compressive sensing mechanism can be seamlessly incorporated into any applications with privacy concerns, from GWAS analysis to user transaction history mining. We show that the compressive mechanism improves the accuracy of the result statistics by up to an order of magnitude, in both theoretical analysis as well as empirical studies.

The organization of the paper is as follows. Section 2 summarizes existing studies of differential privacy and compressive sensing. Section 3 gives an overview of differential privacy and compressive sensing. Section 4 introduces *universal mechanisms* and the basic definitions of our target problems. Section 5 presents the compressive mechanism and analyzes its error. Section 6 generalizes the compressive mechanism to a streaming environment, for continual release of statistics. Section 7
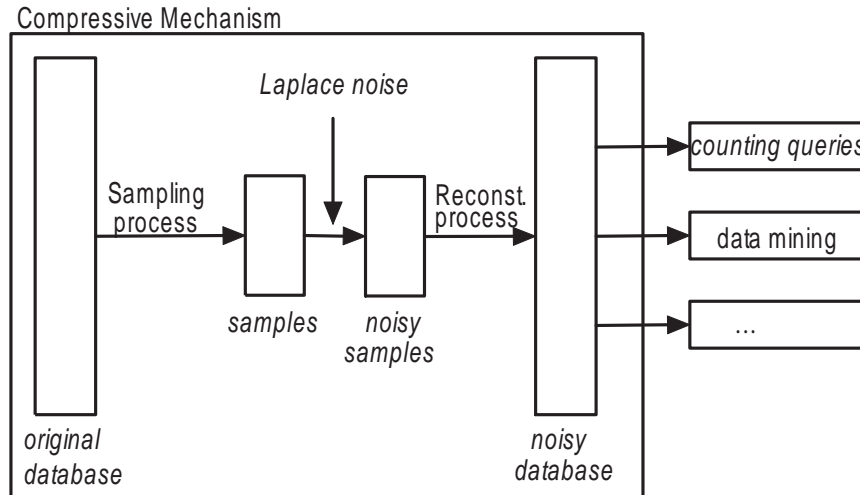
Figure 1: Compressive mechanism framework.

demonstrates the usefulness of these new methods through empirical studies with real data sets. Finally, Section 8 concludes the paper and outlines interesting research directions for future work.

2 Related Work

This section gives an overview of work on differential privacy and compressive sensing.

2.1 Differential Privacy

The notion of ϵ-differential privacy was introduced in [21, 18]. Alternative definitions soon appeared, such as (ϵ, δ)-differential privacy [20, 40], pan-privacy [24, 23] and zero-knowledge privacy [25]. (ϵ, δ)-differential privacy relaxes ϵ-differential privacy, while pan-privacy and zero-knowledge privacy are stronger than ϵ-differential privacy and apply to special circumstances. This work uses the more popular ϵ-differential privacy definition. [19] provides a comprehensive survey of the development of differential privacy.

Considerable efforts have been devoted to the design of ϵ-differentially private mechanisms, but the majority of them only deal with linear counting queries, e.g., [12, 46, 27, 30, 23, 26, 28]. In contrast, our compressive sensing mechanism can handle *arbitrary* statistical queries. Particularly, [46] uses a Haar wavelet transform and [28] designs a tree structure for answering range counting queries. However, neither the Haar wavelet transform nor tree structure can be generalized to answer other queries in a private and accurate way. [46] and [28] are merged into a unified framework of [30] for answering linear counting queries. A recent work [14] separates counting queries from arbitrary low-sensitivity queries and essentially shows that counting is much easier than arbitrary low-sensitivity queries. [39, 4] use (deterministic) Fourier transforms, for problems that are very different from ours; while we take advantage of a probabilistic transform in compressive sensing to reduce the dimension of data from $O(n)$ to $O(\log n)$.

2.2 Compressive Sensing

Compressive sensing was introduced by [9, 11, 17], and later shown to have extensive applications in imaging [41, 33, 6], signal processing [44], computational biology [42], geophysical data analysis

3

[31], communications [43] and so on. To the best of our knowledge, we are the first to apply compressive sensing to sensitive data analysis.

[48, 47, 32] apply random projections to differential privacy. They show that the compressed data can be used for certain statistical tasks, and do not consider the reconstruction process of compressive sensing. [13] reconsiders the contingency table release problem [4] for sparse data (without any transformation), without using compressive sensing. [22] is the most relevant paper in the sense that it works in the opposite direction of ours: applying privacy to compressive sensing.

3 Preliminaries

3.1 Notation

We use $\mathbb{R}$ ($\mathbb{R}^+$) to denote the set of real numbers (positive real numbers) and $[1, n]$ to represent the integer set $\{1, 2, \ldots, n\}$. For two vectors $\boldsymbol{v}, \boldsymbol{w} \in \mathbb{R}^n$, $\langle \boldsymbol{v}, \boldsymbol{w} \rangle$ means their inner product. The p-norm (p is a positive integer) of a vector $\boldsymbol{v} \in \mathbb{R}^n$ is defined to be $(\sum_{i=1}^n |\boldsymbol{v}[i]|^p)^{\frac{1}{p}}$ and is denoted by $\|\boldsymbol{v}\|_p$. For example, the 2-norm of the difference of two points gives their Euclidian distance. We write log for $\log_2$. The number e is the base of natural logarithms. We let $Lap(\lambda)$ denote the one-dimensional Laplacian distribution centered at 0 with scale λ and the corresponding density function $g(x) = \frac{1}{2\lambda} e^{-\frac{|x|}{\lambda}}$. The composition of functions g and h is denoted $g \circ h$, meaning that we first apply g to input and then h to the output of g.

We use other notation common in theoretical computer science. When describing asymptotic complexity, we use $\tilde{O}$ (pronounced soft-O) as a variant of O (pronounced big-O) that ignores logarithmic factors. For instance, if the complexity is $O(n/\log n)$, we simply write $\tilde{O}(n)$. Also, $\exp(n)$ means e^{Cn} for some constant C. *With high probability* means with probability at least 0.99.

3.2 Compressive Sensing

This section gives a brief overview of the theory of compressive sensing, and we refer the reader to an excellent survey [8] for more information. As shown in Figure 1, compressive sensing consists of a probabilistic compression procedure, also called the *sampling* process, followed by a *reconstruction* process that decodes the compressed data. The sampling process reduces the data size from $O(n)$ to $O(\log n)$. The rather complex decoding process exactly or approximately reconstructs the original data from the compressed samples. Readers who are not interested in the mathematics underlying the compressive sensing technique should skip to the next section.

In what follows, all vectors are over $\mathbb{R}^n$ unless otherwise noted. Consider a vector $\boldsymbol{D}$ that we wish to represent using an *orthonormal* basis (such as a standard basis, or a wavelet basis) $\boldsymbol{\Psi} = [\boldsymbol{\psi}_1, \ldots, \boldsymbol{\psi}_n]$. Let $\boldsymbol{x} = (\boldsymbol{x}[1], \ldots, \boldsymbol{x}[n])$ be the coefficient sequence of $\boldsymbol{D}$ under the new basis $\boldsymbol{\Psi}$. Then we have:

$$\boldsymbol{D}[j] = \sum_{i=1}^n \boldsymbol{x}[i]\boldsymbol{\psi}_i[j].$$

If we treat $\boldsymbol{\Psi}$ as an $n \times n$ matrix with $\boldsymbol{\psi}_1, \ldots, \boldsymbol{\psi}_n$ as the columns, $\boldsymbol{D}$ can be written as $\boldsymbol{\Psi}\boldsymbol{x}$, and we say that $\boldsymbol{x}$ represents $\boldsymbol{D}$ under the new basis. We call $\boldsymbol{x}$ *S-sparse* if it has at most S nonzero entries. Let $\boldsymbol{x}_S$ be obtained from $\boldsymbol{x}$ by replacing its $n - S$ coefficients with smallest absolute value by 0. Then $\boldsymbol{x}_S$ is S-sparse.

Often data are compressible. Given a constant $0 < p < 1$, we say vector $\boldsymbol{x}$ is *(p-)compressible* with magnitude R if its components taken in sorted order obey $|\boldsymbol{x}_{(i)}| \leq R \cdot i^{-1/p}, \forall i \in [1, n]$. A

compressible vector x can be well approximated by an S-sparse vector in the sense that $||\boldsymbol{x} - \boldsymbol{x}_S||_1 \leq C_p \cdot R \cdot S^{1-1/p}$ for some constant C_p. A vector $\boldsymbol{D}$ has an *(S-)sparse representation* if there is an orthonormal basis $\boldsymbol{\Psi}$ (called a *sparse basis*) where $\boldsymbol{D}$'s representation $\boldsymbol{x}$ is sparse or compressible, i.e., $\boldsymbol{x}$ is S-sparse or is compressible to an S-sparse vector $\boldsymbol{x}_S$.

The input to compressive sensing is a vector $\boldsymbol{D}$ with a sparse representation. Then there is a sampling process which can be characterized as a linear mapping. We use a matrix $\boldsymbol{\Phi} \in \mathbb{R}^{k \times n}$ to describe the sampling operator and the result is a vector $\boldsymbol{y} = \boldsymbol{\Phi} \boldsymbol{D} \in \mathbb{R}^k$. Candes and Tao [10] define the r-th *restricted isometry constant* δ_r of $\boldsymbol{\Phi}$ to be the smallest number such that

$$(1 - \delta_r)||\boldsymbol{x}||_2^2 \leq ||\boldsymbol{\Phi}\boldsymbol{x}||_2^2 \leq (1 + \delta_r)||\boldsymbol{x}||_2^2,$$

for all r-sparse vectors $\boldsymbol{x} \in \mathbb{R}^n$. The condition that $\delta_{2S} \ll 1$ implies that all pairwise distances between S-sparse signals must be well-preserved in the measurement space, i.e.,

$$(1 - \delta_{2S})||\boldsymbol{x}_1 - \boldsymbol{x}_2||_2^2 \leq ||\boldsymbol{\Phi}\boldsymbol{x}_1 - \boldsymbol{\Phi}\boldsymbol{x}_2||_2^2 \leq (1 + \delta_{2S})||\boldsymbol{x}_1 - \boldsymbol{x}_2||_2^2,$$

for all S-sparse vectors $\boldsymbol{x}_1, \boldsymbol{x}_2$. We loosely say that a matrix $\boldsymbol{\Phi}$ satisfies the *Restricted Isometry Property* (RIP) if it has $\delta_{2S} \ll 1$. Such $\boldsymbol{\Phi}$ always exists. For example, with probability $1 - \exp(-k)$, a random matrix $\boldsymbol{\Phi}$ formed by sampling independent and identically distributed (i.i.d.) entries from a symmetric Bernoulli distribution, more precisely

$$Prob(\boldsymbol{\Phi}(i,j) = \pm\frac{1}{\sqrt{k}}) = \frac{1}{2},$$

satisfies RIP, provided that

$$k \geq C \cdot S \log(n/S),$$

where C is some constant [5]. In words, every entry of $\boldsymbol{\Phi}$ equals $\frac{1}{\sqrt{k}}$ with probability $1/2$ and equals $-\frac{1}{\sqrt{k}}$ with probability $1/2$. In the following discussion, we will assume that $\boldsymbol{\Phi} \in \mathbb{R}^{k \times n}$ is in such a form. It is not hard to verify that $\boldsymbol{A} = \boldsymbol{\Phi}\boldsymbol{\Psi} \in \mathbb{R}^{k \times n}$ also satisfies RIP with overwhelming probability $1 - \exp(-k)$, where $k = \Theta(S \log(n/S))$, for any fixed orthonormal basis $\boldsymbol{\Psi} \in \mathbb{R}^{n \times n}$ [5].

Up to now, we have clarified the input and the sampling process, obtaining a sample vector $\boldsymbol{y} = \boldsymbol{\Phi}\boldsymbol{D} \in \mathbb{R}^k$ with $k = \Theta(S \log(n/S))$. The next and final step is to reconstruct the vector $\boldsymbol{D}$ from $\boldsymbol{y}$ through the sparse representation of $\boldsymbol{D}$. The samples may be contaminated with an unknown noise $\boldsymbol{e} \in \mathbb{R}^k$, and the sample vector becomes

$$\boldsymbol{y}^* = \boldsymbol{y} + \boldsymbol{e} = \boldsymbol{\Phi}\boldsymbol{D} + \boldsymbol{e} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{e},$$

where $\boldsymbol{A} = \boldsymbol{\Phi}\boldsymbol{\Psi}$ is known from the sampling process. Candes, Romberg and Tao [7] prove the remarkable and surprising result that by solving a combinatorial optimization problem, the recovered answer $\boldsymbol{x}^* \in \mathbb{R}^n$ can be close enough to $\boldsymbol{x}$ even in the presence of unknown perturbations. Needell and Tropp [38] prove essentially the same error bound using a greedy algorithm. The details of the two algorithms can be found in the appendix and the result is summarized as follows.

Lemma 1 ([7, 38]) *Suppose $\boldsymbol{A}$ satisfies RIP and $||\boldsymbol{e}||_2 \leq \theta$. Then $||\boldsymbol{x} - \boldsymbol{x}^*||_2 \leq \frac{C_2||\boldsymbol{x} - \boldsymbol{x}_S||_1}{\sqrt{S}} + C_3\theta$, for some constants C_2 and C_3.*

For a compressible vector $\boldsymbol{x} \in \mathbb{R}^n$, we mentioned that $||\boldsymbol{x} - \boldsymbol{x}_S||_1 \leq C_p \cdot R \cdot S^{1-1/p}$ for some $p \in (0, 1)$. Also note that $\boldsymbol{D}$ can be recovered by setting $\boldsymbol{D}^* = \boldsymbol{\Psi}\boldsymbol{x}^*$ and that $||\boldsymbol{D} - \boldsymbol{D}^*||_2 = ||\boldsymbol{x} - \boldsymbol{x}^*||_2$. We can derive the following corollary for our purpose.

Corollary 1 *Suppose that $\boldsymbol{A}$ satisfies RIP and $||\boldsymbol{e}||_2 \leq \theta$. Then $||\boldsymbol{D} - \boldsymbol{D}^*||_2 = O(S^{1/2-1/p} + \theta)$ for some constant $S \ll n$ and $p \in (0, 1)$.*

If the input vector $\boldsymbol{D} \in \mathbb{R}^n$ has no sparse representation, then the $\boldsymbol{D}^* \in \mathbb{R}^n$ obtained by compressive sensing could be very far away from $\boldsymbol{D}$, and that is why we require the input to have a sparse representation. Formally,

Corollary 2 *Suppose $\boldsymbol{A}$ satisfies RIP and $||\boldsymbol{e}||_2 \leq \theta$. If $\boldsymbol{D}$ has no sparse representation, then $||\boldsymbol{D} - \boldsymbol{D}^*||_2 = O(n/\sqrt{S} + \theta)$.*

Finally, we remark that the time complexity and the space complexity of compressed sensing are both $\tilde{O}(n)$.

3.3 Differential Privacy

In this paper, we represent a database as a vector $\boldsymbol{D} \in \mathbb{R}^n$. This abstraction encompasses many previous abstractions of data, such as a data distribution [40], histogram [27], contingency table [4], private bits [16], the database itself, and recommendation systems [34]. Two databases $\boldsymbol{D}_1, \boldsymbol{D}_2 \in \mathbb{R}^n$ are said to be *neighboring* iff $||\boldsymbol{D}_1 - \boldsymbol{D}_2||_1 \leq 1$. The notion of differential privacy is defined as follows.

Definition 1 ([21, 18]) *A randomized mechanism $\mathcal{K}$ provides ϵ-differential privacy if for all neighboring databases $\boldsymbol{D}_1, \boldsymbol{D}_2 \in \mathbb{R}^n$ and all $Sub \subseteq Range(\mathcal{K})$,*

$$Prob(\mathcal{K}(\boldsymbol{D}_1) \in Sub) \leq e^\epsilon \times Prob(\mathcal{K}(\boldsymbol{D}_2) \in Sub),$$

where the probability space in each case is over the coin flips of $\mathcal{K}$.

A popular mechanism for achieving ϵ-differential privacy is the *Laplacian mechanism* [21], which can be used when the output of the mechanism is numeric. McSherry and Talwar developed a technique called the *exponential mechanism* [35] for problems where the output is non-numeric.

Laplacian mechanism: The *sensitivity* of a query $Q : \mathbb{R}^n \rightarrow \mathbb{R}^d$ is defined to be

$$\Delta_Q = \max_{\boldsymbol{D}_1, \boldsymbol{D}_2} ||g(\boldsymbol{D}_1) - g(\boldsymbol{D}_2)||_1,$$

for all neighboring $\boldsymbol{D}_1, \boldsymbol{D}_2 \in \mathbb{R}^n$. [21] shows the following result:

Lemma 2 ([21]) *For $Q : \mathbb{R}^n \rightarrow \mathbb{R}^d$, the mechanism $\mathcal{K}_Q$ that adds independently generated noise with distribution $Lap(\Delta_Q/\epsilon)$ to each of the d output values provides ϵ-differential privacy.*

Exponential mechanism: This mechanism is for the case where the query answer y is not numerical. We rely on a pre-defined utility function $u(\boldsymbol{D}, y)$ (with a numeric output) to measure the quality of y, compared to the exact answer. The exponential mechanism outputs y with probability proportional to $e^{-\epsilon u(\boldsymbol{D}, y)/(2\Delta u)}$, where Δu is the sensitivity of the utility function $u(\boldsymbol{D}, y)$. The exponential mechanism provides ϵ-differential privacy [35]. The distance of an answer from the best answer, which has the smallest u, exhibits an exponential tail and with probability almost 1, the exponential mechanism outputs an object with an approximately optimal value.

Continual mechanism and pan-privacy: For the case where the database is updated over time, the theory community has investigated what they call differential privacy under continual observation [23]. In their setting, the input is no longer a static vector, but instead a stream of 0's and 1's, denoted by $\boldsymbol{\sigma} \in \{0, 1\}^T$, where T is an upper limit of time. The *continual mechanism* [23]

receives an input $\boldsymbol{\sigma}[t] \in \{0, 1\}$ at each time $t \in [1, T]$, and outputs an approximation to the number of 1's seen in the length t prefix of the stream. Two streams' *prefixes* $\boldsymbol{\sigma}_t, \boldsymbol{\sigma}'_t \in \{0, 1\}^t$, $t \in [1, T]$, are *neighboring* iff $||\boldsymbol{\sigma}_t - \boldsymbol{\sigma}'_t||_1 \leq 1$. The definition of ϵ-*differential pan-privacy* under continual observation [24, 23] is stronger than ϵ-differential privacy and is stated as follows.

Definition 2 ([24, 23]) *Let $I_\mathcal{K}$ denote the set of internal states of the randomized mechanism $\mathcal{K}$. $\mathcal{K}$ provides ϵ-differential pan-privacy (against a single intrusion) if for all neighboring stream prefixes $\boldsymbol{\sigma}_t, \boldsymbol{\sigma}'_t \in \{0, 1\}^t$, $t \in [1, T]$ and for all sets $I' \subseteq I_\mathcal{K}$ and $Sub \subseteq Range(\mathcal{K})$,*

$$Prob(\mathcal{K}(\boldsymbol{\sigma}_t) \in (I', Sub)) \leq e^\epsilon \times Prob(\mathcal{K}(\boldsymbol{\sigma}'_t) \in (I', Sub)),$$

where the probability space in each case is over the coin flips of $\mathcal{K}$.

The details of the continual mechanism are included in the appendix. The continual mechanism can be easily generalized to the case in which input $\boldsymbol{D} \in \mathbb{R}^T$. If we denote the approximate sum of the first t entries by Σ_t^* and the true sum of the first t input items as Σ_t, the result of [23] can be formalized (in a slightly differently way) as follows.

Lemma 3 ([23]) *The continual mechanism provides ϵ-differential pan-privacy. At each time $t \in [1, T]$, with probability at least $1 - \beta$, $|\Sigma_t^* - \Sigma_t| = O(\log(1/\beta) \log^{1.5}(T)/\epsilon)$.*

4 Problem Definition

This section formally defines the problems investigated in this paper. Specifically, we focus on ϵ-*differentially private randomized mechanisms* with *numeric* outputs, which can be formalized as $\mathcal{K} : \mathbb{R}^n \to \mathbb{R}^d$. Such randomized mechanisms can be used to answer (numeric) statistical queries (in the form of $Q : \mathbb{R}^n \to \mathbb{R}^d$) about a database. In general, a randomized mechanism for publishing query results about a database must resolve a trade-off between *utility* and *privacy*. Utility means that the outputs should not be too far away from the true answers of the query, to ensure that the perturbed answers can still be helpful to users. Privacy requires that the outputs not be too near to the true answers, since some amount of random perturbation is essential for the mechanism to be ϵ-differentially private. While many previous studies mainly deal with *linear counting* queries, in practice the queries are likely to be much more diverse and complicated, such as a multiphase analysis task. A *general* randomized mechanism with respect to a certain kind of database should be able to answer *all possible* queries with guarantees about utility and privacy. In other words, given a database $\boldsymbol{D}$ of a certain kind X, any query Q over $\boldsymbol{D}$ has to be answered with reasonable utility and privacy guarantees. For this purpose, this section defines the notion of a *universal mechanism*. We begin with the notion of an *identity query*, which just returns the entire database unchanged. Intuitively, the purpose of a universal mechanism is to accurately answer the identity query, subject to differential privacy.

Definition 3 *A universal mechanism with respect to a class X of databases is a randomized mechanism $\mathcal{U}_X : X \to \mathbb{R}^n$ for answering the identity query, satisfying the following conditions:*

- ϵ-*differential privacy;*

- *with high probability, $||\boldsymbol{D} - \boldsymbol{D}^*||_2 = O(\log(n)/\epsilon)$, for any input $\boldsymbol{D} \in X$ and its corresponding output $\boldsymbol{D}^* \in \mathbb{R}^n$.*

Table 1: Error bounds (in terms of $||\boldsymbol{D} - \boldsymbol{D}^*||_2$) for the identity query.

Mechanism	Bound
Compressive Mechanism	$O(\log n/\epsilon)$
Laplacian Mechanism	$O(\sqrt{n}/\epsilon)$
[46, 28]	$\tilde{O}(\sqrt{n}/\epsilon)$

A universal mechanism is a good base for answering all kinds of statistical queries. To answer any query Q, we first apply $\mathcal{U}_X$ to a database $\boldsymbol{D} \in X$, obtaining $\boldsymbol{D}^*$. Then we deterministically compute an answer to Q over $\boldsymbol{D}^*$. Since $\boldsymbol{D}^*$ must be very close to $\boldsymbol{D}$, utility can be guaranteed for any query.

Another important property of a universal mechanism is that it allows us to answer an unbounded number of queries without *any concern* for privacy budget issues. Previous ϵ-differential privacy mechanisms can only answer a finite number of statistical queries, due to the budget limit. For example, if a mechanism answers two queries each in an ϵ-differentially private way, then the mechanism may have provided only 2ϵ-differential privacy overall. The universal mechanism does not have this problem, as any system based on a universal mechanism *always* satisfies ϵ-differential privacy, no matter how many queries are asked.

Another advantage of universal mechanisms is that $\boldsymbol{D}^*$ can be published in its entirety, thus supporting both interactive and non-interactive querying. For example, biologists who perform GWAS can simply publish allele frequencies, rather than answering queries about the frequencies. So a universal mechanism itself can be very useful, even without consideration of subsequent queries.

We use $O(\log(n)/\epsilon)$ as the upper bound of error in the definition of universal mechanism. One reason for this choice is that we conjecture that $\Omega(\log(n)/\epsilon)$ is the lower bound of error to satisfy ϵ-differential privacy. Previous lower bounds [27, 16, 14] cannot be applied to our case, and therefore we leave this conjecture as an open problem, elaborated further at the end of this article.

To summarize, a universal mechanism is resilient to *any form* and *any number* of statistical attacks from any number and kind of *malicious attackers*, which makes it *universally robust*. Meanwhile, the statistical query results remain relatively accurate, as $\boldsymbol{D}^*$ must be very close to $\boldsymbol{D}$.

The Laplacian mechanism is not a universal mechanism, as it introduces too much error. Suppose we use it to answer the identity query for $\boldsymbol{D}$, producing $\boldsymbol{D}^*$. With high probability $||\boldsymbol{D} - \boldsymbol{D}^*||_2 = \Theta(\sqrt{n}/\epsilon)$ (by using a Chernoff-like argument). Nor is any other known ϵ-differentially private randomized mechanism universal. The challenge, then is to devise a universal mechanism. In later sections we introduce the *compressive mechanism* and show that it is a universal mechanism with respect to databases with a sparse representation. Table 1 compares the error bounds of the compressive mechanism and other contenders. In the remainder of this section, we present two example use cases for universal mechanisms.

Example 1 *Biomedical researchers use human genome sequence data to determine the correlations between particular diseases and combinations of SNPs. They perform statistical tests on the data after computing preliminary estimates of the importance of the SNPs [45]. In the US, frequency information for the SNPs in NIH-funded studies is no longer publicly available, due to privacy concerns. A universal mechanism could allow publication of this information, which would be used by many other researchers.*

Example 2 *Internet service providers (ISPs) share statistical data from their network traces, to help detect anomaly events of large scale [36]. The identification of anomalies involves matrix com-*
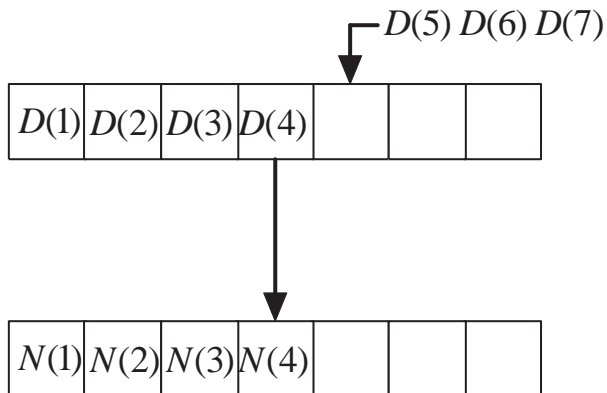
Figure 2: A universal mechanism for databases with updates (continual observation).

putations on the statistics, which are not generally supported by existing mechanisms for differential privacy.

A universal mechanism can also be helpful for databases that get updated. The continual mechanism of [23] can only answer linear counting queries (more precisely, only range counting queries starting from time 1). We aim to establish a pan-private mechanism such that at each time t, we can answer any query over the database states $(\boldsymbol{D}[1], \ldots, \boldsymbol{D}[t])$ we have seen so far. We will realize this ambition by designing a mechanism for answering the identity query at each time t. The definition of the identity query in a dynamic setting is a straightforward generalization of identity queries in a static setting, and we omit it here.

Figure 2 illustrates how a universal mechanism works under continual observation. When a new tuple $D(4)$ comes into the database, an updating procedure is triggered to generate a corresponding noisy tuple $N(4)$ under a universal mechanism. The noisy values calculated so far are published continuously to the public, guaranteeing that any snapshot of the publication follows the requirements of a universal mechanism.

Example 3 *The continual observation setting is important for public health report publication. For example, Singapore's Ministry of Health periodically publishes the number of patients with certain chronic diseases. Treating each reporting period as a separate and unrelated database, and providing ϵ-differential privacy for each separate database, does not provide ϵ-differential privacy for patients across multiple reporting periods. To preserve the privacy of patients, it is important to design mechanisms that can support periodic release of statistics over a long period of time.*

5 Compressive Mechanism

5.1 Mechanism Design

The overall aim of the compressive mechanism is to answer the identity query for databases that have a sparse representation, in an ϵ-differentially private manner. Algorithm 1 summarizes the compressive mechanism. The input to the compressive mechanism includes the privacy budget ϵ and a database $\boldsymbol{D}$ whose sparse representation is a compressible vector $\boldsymbol{x}$ under an orthonormal basis $\boldsymbol{\Psi} \in \mathbb{R}^{n \times n}$. Vector $\boldsymbol{x}$ can be well approximated by its $\boldsymbol{x}_S$, for some constant S. X is the set of all databases with a sparse representation.

Applying the sampling operator $\boldsymbol{\Phi}$ produces a sample vector $\boldsymbol{y} = \boldsymbol{\Phi}\boldsymbol{D} \in \mathbb{R}^k$, where $k = \Theta(S \log(n/S))$. We add random noise to each entry of $\boldsymbol{y}$. That is to say, for every $\boldsymbol{y}[i]$, we have

$\boldsymbol{y}^*[i] = \boldsymbol{y}[i] + \boldsymbol{e}[i]$, where $\boldsymbol{e}[i] \sim Lap(\sqrt{k}/\epsilon)$. Then $\boldsymbol{y}^* = \boldsymbol{y} + \boldsymbol{e}$. We can recover a noisy $\boldsymbol{x}^*$ from $\boldsymbol{y}^*$ through the reconstruction process of compressed sensing. A noisy $\boldsymbol{D}^*$ is obtained by $\boldsymbol{D}^* = \boldsymbol{\Psi} \boldsymbol{x}^*$. Finally, $\boldsymbol{D}^*$ is output by the compressive mechanism.

Algorithm 1 Compressive Mechanism

Input: privacy budget $\epsilon \in \mathbb{R}^+$, $\boldsymbol{D} \in X$ (possibly together with a sparse basis $\boldsymbol{\Psi} \in \mathbb{R}^{n \times n}$).
Output: $\boldsymbol{D}^* \in \mathbb{R}^n$.

1: Generate a (normalized) random matrix $\boldsymbol{\Phi} \in \mathbb{R}^{k \times n}$ with i.i.d. symmetric Bernoulli distribution.

2: Acquire the sample $\boldsymbol{y} = \boldsymbol{\Phi} \boldsymbol{D} \in \mathbb{R}^k$.
3: Get a noisy sample $\boldsymbol{y}^* = \boldsymbol{y} + \boldsymbol{e} \in \mathbb{R}^k$ for $\boldsymbol{e} \in \mathbb{R}^k$ with i.i.d. $Lap(\sqrt{k}/\epsilon)$.
4: Reconstruct $\boldsymbol{x}^* \in \mathbb{R}^n$.
5: Output $\boldsymbol{D}^* = \boldsymbol{\Psi} \boldsymbol{x}^*$.

5.2 Analysis

Next we analyze the compressive mechanism. First we prove that it satisfies ϵ-differential privacy.

Lemma 4 *The compressive mechanism satisfies ϵ-differential privacy.*

Proof: See the appendix. □

Next we show that $\boldsymbol{D}^*$ is very close to $\boldsymbol{D}$, thus ensuring utility.

Lemma 5 *With high probability, $||\boldsymbol{D} - \boldsymbol{D}^*||_2 = O(\log(n)/\epsilon)$.*

Proof: See the appendix. □

The two lemmas above lead immediately to the following theorem.

Theorem 1 *The compressive mechanism is a universal mechanism with respect to databases with a sparse representation.*

5.3 Discussion

The compressive mechanism is ϵ-differentially private for *all* $\boldsymbol{D} \in \mathbb{R}^n$, and works especially well in terms of error bounds for $\boldsymbol{D} \in X$. In other words, for any input $\boldsymbol{D} \in \mathbb{R}^n$ we guarantee $||\boldsymbol{D}^* - \boldsymbol{D}||_2 = O(n^{2/3}/\epsilon)$ (from Corollary 2), while for $\boldsymbol{D} \in X$, $||\boldsymbol{D}^* - \boldsymbol{D}||_2 \leq O(\log(n)/\epsilon)$ with high probability. In short, the compressive mechanism is ϵ-differentially private for all $\boldsymbol{D} \in \mathbb{R}^n$ and is a universal mechanism for databases with a sparse representation.

Consider the issue of choosing the right S, the sparsity parameter. S may not be known in advance, and we may have to choose the best (i.e., with least error) S ourselves. S depends on the input data $\boldsymbol{D}$, and since $k = \Theta(S \log(n/S))$ and $||\boldsymbol{e}||_2 = O(k/\epsilon)$, S also affects the encoding (adding noise) results. Therefore, we have to choose S in a differentially private way, and a natural method to achieve this aim is the exponential mechanism. S could be any element in $[1, n]$ (if we require the compressive mechanism to work for all possible $\boldsymbol{D} \in \mathbb{R}^n$), and for each possible S, we define its utility function as

$$u(\boldsymbol{D}, S) = \frac{C_2 ||\boldsymbol{x} - \boldsymbol{x}_S||_1}{\sqrt{S}} + \frac{C_4 S \log(n/S)}{\epsilon}, \tag{1}$$

where C_2 and C_4 are (known) constants. The right hand side of (1) is the upper bound on the error according to Lemma 1 (with θ instantiated). We calculate that the sensitivity of $u(\boldsymbol{D}, S), \forall S \in [1, n]$, is $\Delta_{u(\boldsymbol{D}, S)} = C_5/\sqrt{S}$, where C_5 is a constant. Accordingly, the exponential mechanism outputs S with probability proportional to $e^{-\epsilon u(S)/(2\Delta_{u(S)})}$, where $u(S)$ stands for $u(\boldsymbol{D}, S)$; satisfies ϵ-differential privacy; and ensures the near-optimality of S with truly negligible failure probability.

If the total privacy budget is ϵ, we may choose to distribute a small part (say, 0.1ϵ or 0.01ϵ) to the exponential mechanism. In this case, the analysis of the compressive mechanism remains unchanged (recall that we analyze the error in an asymptotic way).

The compressive mechanism may involve the art of identifying a suitable orthonormal basis $\boldsymbol{\Psi} \in \mathbb{R}^{n \times n}$ under which $\boldsymbol{D} \in \mathbb{R}^n$ can be sparse or compressible. This is a mature and profound area in mathematics that has been explored for many years, and we refer the reader to an outstanding book [15]. In this paper, we simply assume that $\boldsymbol{D} \in X$ has a sparse representation and its sparse basis $\boldsymbol{\Psi} \in \mathbb{R}^{n \times n}$ could be treated as part of the input. In fact, many natural data sets have a sparse representation [8], and that is one reason why compressed sensing has been so widely used since its invention.

The time and space complexity of the compressive mechanism are both $\tilde{O}(n)$.

6 Continual Observation

This section focuses on an extension of the compressive mechanism, the *compressive mechanism under continual observation*, or CMCO for short.

6.1 Mechanism Design

As in the static case, we suppose that $X \subseteq \mathbb{R}^T$ is the set of all databases with a sparse representation. The overall aim of CMCO is to answer the identity query at each time t in a differentially pan-private manner. The main steps of CMCO are summarized in Algorithm 2.

CMCO takes as input a constant ϵ, the parameter for differential pan-privacy; and (a prefix of) $\boldsymbol{D} \in X$, which has a sparse representation $\boldsymbol{x} \in \mathbb{R}^T$ under sparse basis $\boldsymbol{\Psi} \in \mathbb{R}^{T \times T}$. The input differs from the static case in that we do not receive all of $\boldsymbol{D}$ at once, but one value at a time; more precisely, at each time t, we receive $\boldsymbol{D}[t] \in \mathbb{R}$.

At each time t, we generate a new random vector $\boldsymbol{\Phi}_t \in \mathbb{R}^k$ (recall that $k = \Theta(S \log(T/S))$, where S is the sparsity parameter of $\boldsymbol{x}$). Each entry of $\boldsymbol{\Phi}_t$ is distributed according to a symmetric Bernoulli distribution; more concretely,

$$Prob(\boldsymbol{\Phi}_t[i] = \pm 1/\sqrt{k}) = 1/2.$$

Upon receiving $\boldsymbol{D}[t]$, we apply $\boldsymbol{\Phi}_t$ to it and get a new vector $\boldsymbol{u}_t = \boldsymbol{\Phi}_t \boldsymbol{D}[t] \in \mathbb{R}^k$.

As a result, by time t, there are t vectors $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_t \in \mathbb{R}^k$, forming a matrix $\boldsymbol{M} \in \mathbb{R}^{k \times t}$. The i-th row of M, $i \in [1, k]$, is called $\boldsymbol{m}_i \in \mathbb{R}^t$. As t grows, the size of $\boldsymbol{m}_i \in \mathbb{R}^t$ grows correspondingly. We apply k independent continual mechanisms to each $\boldsymbol{m}_i$ to estimate the sum of the first t entries in $\boldsymbol{m}_i \in \mathbb{R}^t$ in an ϵ-differentially pan-private way. The k independent continual mechanisms return a vector $\boldsymbol{v}_t^* \in \mathbb{R}^k$ with each entry $\boldsymbol{v}_t^*[i]$ representing the estimate of the sum of the first t entries in $\boldsymbol{m}_i$. We use $\boldsymbol{v}_t^* \in \mathbb{R}^k$ as a resource for reconstruction and obtain a noisy $\boldsymbol{x}_t^* \in \mathbb{R}^t$. We obtain $\boldsymbol{D}_t^* \in \mathbb{R}^t$ by $\boldsymbol{D}_t^* = \boldsymbol{\Psi}_t \boldsymbol{x}_t^*$, where $\boldsymbol{\Psi}_t \in \mathbb{R}^{t \times t}$ is the orthonormal basis (of the corresponding $\boldsymbol{\Psi} \in \mathbb{R}^{T \times T}$) in a space of smaller dimension.

We do not store $\boldsymbol{M} \in \mathbb{R}^{k \times t}$ at all; and we just discuss it for *analysis*. What we store at each time t is just what is stored for the k ϵ-differentially pan-private continual mechanisms, namely k noisy sums and some independent noise (see the appendix for further details).

11

Table 2: Error bounds comparison (in terms of $||\boldsymbol{D}_t - \boldsymbol{D}_t^*||_2$) for answering the identity query at time t.

Mechanism	Bound
CMCO	$\tilde{O}(\log^{1.5}(T)/\epsilon)$
Continual Mechanism	$\tilde{O}(\sqrt{t}\log^{1.5}(T)/\epsilon)$

At time t, CMCO outputs $\boldsymbol{D}_t^* \in \mathbb{R}^t$, an estimate of $\boldsymbol{D}_t \in \mathbb{R}^t$ (the first t terms of $\boldsymbol{D} \in \mathbb{R}^T$).

Algorithm 2 CMCO (Compressive Mechanism under Continual Observation)

Initial Input: privacy budget $\epsilon \in \mathbb{R}^+$.
Input at time $t \in [1, T]$: $\boldsymbol{D}[t] \in \mathbb{R}$.
Output at time $t \in [1, T]$: $\boldsymbol{D}_t^* \in \mathbb{R}^t$.

1: Generate a (normalized) random vector $\boldsymbol{\Phi}_t \in \mathbb{R}^k$ with i.i.d. symmetric Bernoulli distributions.

2: Acquire the sample $\boldsymbol{u}_t = \boldsymbol{\Phi}_t \boldsymbol{D}[t] \in \mathbb{R}^k$.
3: Get an estimate vector $\boldsymbol{v}_t^* \in \mathbb{R}^k$, where each $\boldsymbol{v}_t^*[i], i \in [1, \dots, k]$, is the output of a continual mechanism estimating $\sum_{j=1}^t \boldsymbol{u}_j[i]$ in an ϵ-differentially pan-private way.
4: Reconstruct $\boldsymbol{x}_t^* \in \mathbb{R}^t$.
5: Output $\boldsymbol{D}_t^* = \boldsymbol{\Psi}_t \boldsymbol{x}_t^*$.

6.2 Analysis

In this section, we analyze the performance of CMCO, which needs to be both private and useful.

Theorem 2 *CMCO is ϵ-differentially pan-private and for each time $t \in [1, T]$, $||\boldsymbol{D}_t^* - \boldsymbol{D}_t||_2 = \tilde{O}(\log^{1.5}(T)/\epsilon)$ with high probability.*

Proof: See the appendix. □

6.3 Discussion

First, what if we directly apply the continual mechanism to a problem? Suppose that at time t, we obtain $\Sigma_1^*, \dots, \Sigma_t^* \in \mathbb{R}$ in an ϵ-differentially pan-private way. Then we perform subtraction operations to get the desired output:

$$\boldsymbol{D}_t^*[i] = \Sigma_i^* - \Sigma_{i-1}^*, \forall i \in [2, t].$$

Obviously $\boldsymbol{D}_t^*[1] = \Sigma_1^*$. Calculations show that with high probability,

$$||\boldsymbol{D}_t - \boldsymbol{D}_t^*||_2 = \tilde{O}(\sqrt{t}\log^{1.5}(T)/\epsilon),$$

indicating the performance is not as good as that of CMCO. The bounds comparison of CMCO and the continual mechanism is listed in Table 2.

Second, the time and space complexity of CMCO are both $\tilde{O}(t + \log(n))$ at each time $t \in [1, T]$.

Last but not least, due to pan-privacy concerns, we are not allowed to store the original input values.

7 Experiments

This section illustrates the effectiveness of the compressive mechanism and CMCO by experimental results on real data sets. We employ the data sets *social_network.txt*, *nettrace.txt*, and *search_logs.txt*, contributed by Hay, Rastogi, Miklau and Suciu [28]. The data set *social_network.txt* is a graph derived from friendship relations in an online social network site; *nettrace.txt* is said to be collected at a university; and *search_logs.txt* comes from search query logs collected between 2004 and 2010. *nettrace.txt* consists of $2^{16} = 65536$ entries, while there are $2^{15} = 32,768$ elements in *search_logs.txt*. *social_network.txt* has 11342 entries. We also use a real data set *tcptrace.txt*, which has 7865 entries. *tcptrace.txt* was collected by Berkeley, and contain a 30-day trace of the TCP connections between their local network and the Internet[1].

7.1 Haar Basis

In this part of the section, we discuss results on the data sets when employing Haar basis in compressive sensing. Haar basis forms the commonly used and simplest wavelet transformation.

7.1.1 Choosing a Good S

During the discussion of the compressive mechanism, we mentioned that the exponential mechanism can be used to choose a good sparsity parameter S. In this section, we show the feasibility of the exponential mechanism by running it on the three real data sets *social_network.txt*, *nettrace.txt* and *search_logs.txt*. We set the total privacy budget to $\epsilon = 1$ and allot $0.1\epsilon = 0.1$ for the exponential mechanism. The remaining $0.9\epsilon = 0.9$ will be used to run the compressive mechanism. In addition, we set $\boldsymbol{\Psi} \in \mathbb{R}^{n \times n}$ to be a Haar basis [2].

Figure 3(a) through Figure 3(c) show how error changes as S varies. Then we run the exponential mechanism 1000 times. Figure 3(d) through Figure 3(f) display the result of the exponential mechanism, explicitly supporting our claim that the exponential mechanism chooses a near-optimal S with truly negligible failure probability.

These experiments show that the exponential mechanism works very well in practice and returns a near-optimal S.

7.1.2 Compressive Mechanism

In this section, we experimentally evaluate the performance of the compressive mechanism by comparing it with the Laplacian mechanism and HRMS mechanism [28]. The sparse basis is still the Haar basis.

Figure 4 shows the performance comparison. Both the horizontal and vertical axes use a logarithmic scale. The horizontal axis denotes different choices of the parameter ϵ for differential privacy; while the vertical coordinates are the errors, namely $||\boldsymbol{D} - \boldsymbol{D}^*||_2$ of the input $\boldsymbol{D}$ and the output $\boldsymbol{D}^*$. Overall, the Laplacian and HRMS mechanisms cannot compete with the compressive mechanism, and as ϵ becomes smaller, the compressive mechanism's advantage becomes larger.

The exception is that for $\epsilon = 1$, the Laplacian mechanism is better than the compressive mechanism. This is because our data set is small, and thus $\sqrt{n}$ is not much larger than $S \log(n/S)$. Combined with the reconstruction errors, the compressive mechanism is slightly worse than the Laplacian mechanism. As n becomes larger, from 11342 in Figure 4(a), $2^{15} = 32768$ in Figure 4(b), to $2^{16} = 65536$ in Figure 4(c), the advantage of the Laplacian mechanism for the case $\epsilon = 1$
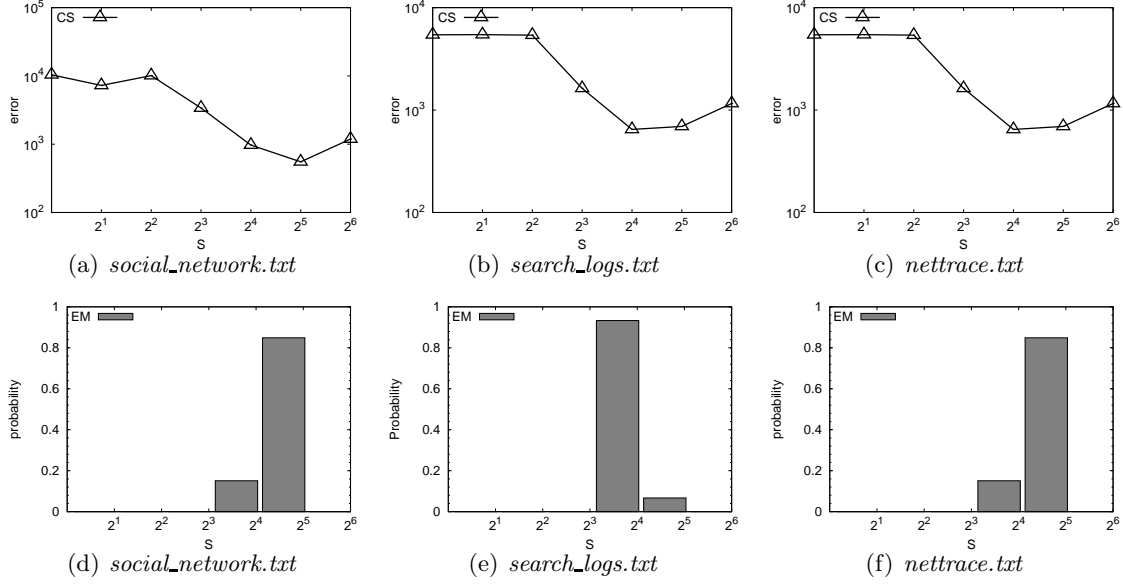
---

[1]http://ita.ee.lbl.gov/html/contrib/LBL-CONN-7.html

Figure 3: Choosing the right S: the first three is using CS(Compressive Sensing); the last three is using EM(Exponential Mechanism).
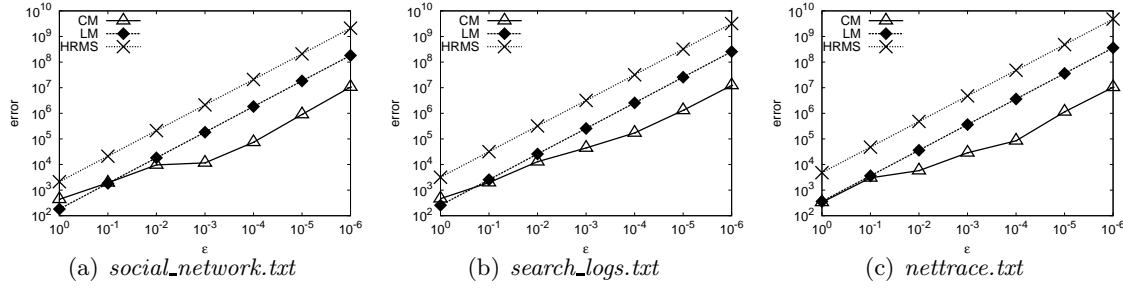

Figure 4: A comparison of CM (Compressive Mechanism), LM (Laplacian Mechanism) and HRMS (HRMS Mechanism).

becomes smaller and smaller. For other choices of ϵ, the advantage of the compressive mechanism becomes larger and larger as n increases.

7.1.3 CMCO

To demonstrate the strength of CMCO over the continual mechanism, we set ϵ and run the two algorithms on *social_network.txt*, *search_logs.txt* and *nettrace.txt*. Figure 5 shows that the continual mechanism has significantly lower utility than the compressive mechanism under continual observation. The x-axis varies the time t and the y-axis is the corresponding change in error, namely $||\boldsymbol{D}_t - \boldsymbol{D}_t^*||_2$. The sparse basis is still the Haar basis.

7.2 Cosine Basis

The previous experiments used the Haar basis as the orthonormal basis $\boldsymbol{\Psi} \in \mathbb{R}^{n \times n}$. This is an appropriate choice because the three data sets of Hay, Rastogi, Miklau and Suciu [28] have the property that every two adjacent elements in a file are very close to each other. To illustrate the power of the compressive mechanism, here we employ a different orthonormal basis, the cosine
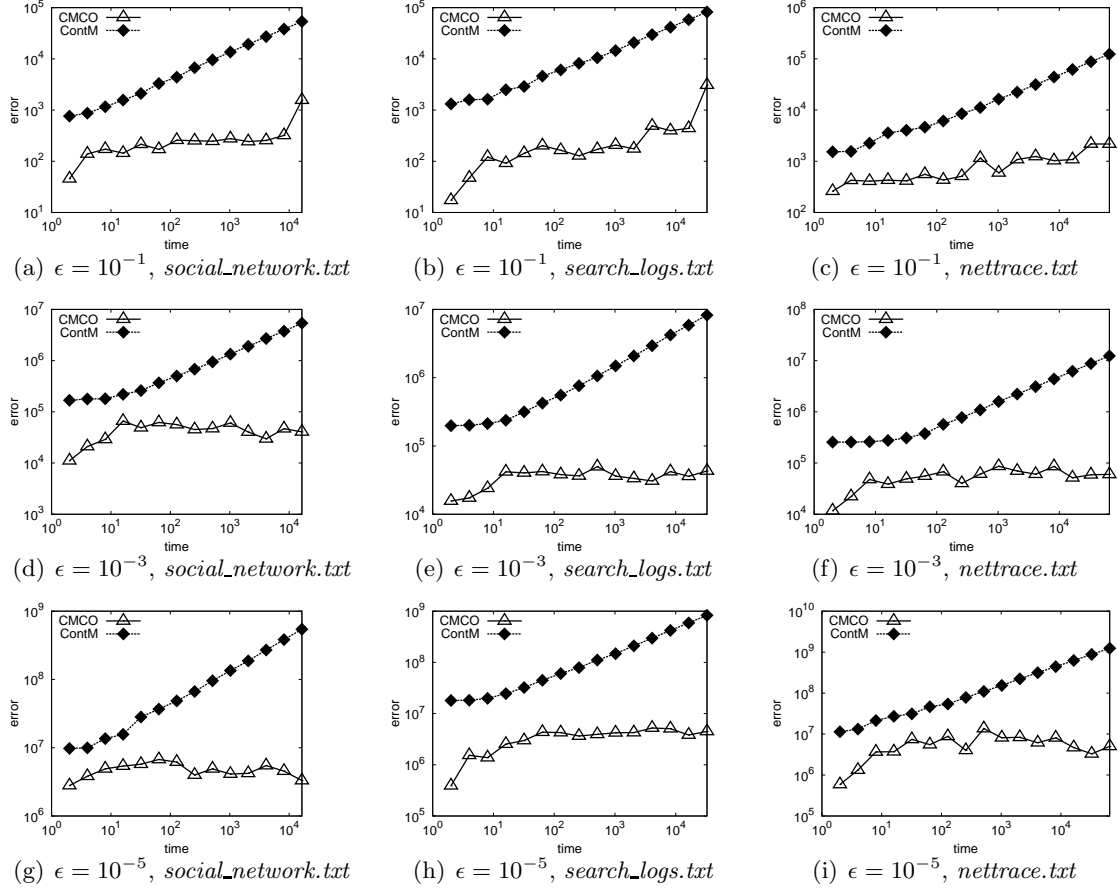
Figure 5: A comparison of CMCO (Compressive Mechanism under Continual Observation) and ContM (Continual Mechanism).

basis [3], on a time-related data set *tcptrace.txt*. Figure 6 shows that the compressive mechanism far outperforms the Laplacian mechanism and HRMS mechanism.

8   Concluding Remarks

We have introduced the compressive mechanism as a means of realizing the idea of a universal mechanism. We have provided theoretical bounds and experimental results for the compressive mechanism, and showed how to apply the compressive mechanism to a case of continual observation.

As mentioned earlier, one open problem concerns the lower bound of error in the definition of the universal mechanism. Formally, assume that there is an ϵ-differentially private mechanism to answer the identity query. Then what is the lower bound of $||\boldsymbol{D} - \boldsymbol{D}^*||_2$, where $\boldsymbol{D}^* \in \mathbb{R}^n$ is the output of the mechanism? Here X can be a subset of $\mathbb{R}^n$, such as databases with a sparse representation.

Regarding future work: are there applications of the compressive mechanism to other scenarios beyond continual observation? What if we use the compressive mechanism to satisfy other privacy definitions such (ϵ, δ)-differential privacy or zero-knowledge privacy? How can one incorporate considerations such as integrity and consistency constraints into our framework?
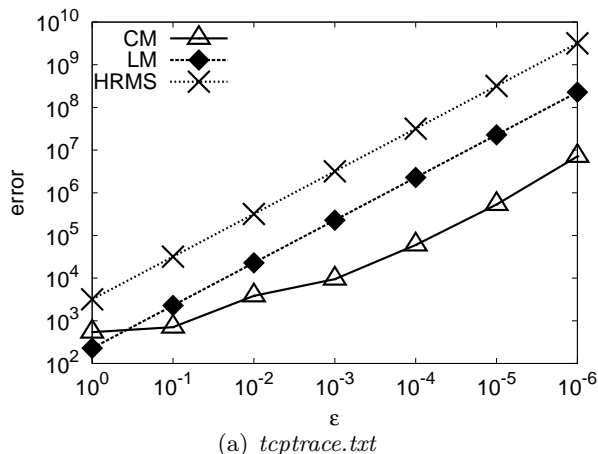
(a) *tcptrace.txt*

Figure 6: A comparison of CM (Compressive Mechanism), LM (Laplacian Mechanism) and HRMS (HRMS Mechanism) for the Cosine Basis.

References

[1] http://www.netflixprize.com/.

[2] http://en.wikipedia.org/wiki/Haar_wavelet.

[3] http://en.wikipedia.org/wiki/Discrete_cosine_transform.

[4] 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 *PODS*, pages 273–282, 2007.

[5] R. Baraniuk, M. Davenport, R. DeVore, and M. Wakin. A simple proof of the restricted isometry property for random matrices. *Constructive Approximation*, 28(3):253–263, 2008.

[6] R. Baraniuk and P. Steeghs. Compressive radar imaging. In *IEEE Radar Conference*, 2007.

[7] E. Candes, J. Romberg, and T. Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on Pure and Applied Mathematics*, 59(8):1207–1223, 2006.

[8] E. Candes and M. Wakin. An introduction to compressive sampling. *IEEE Signal Processing Magazine*, 25(2):21–30, 2008.

[9] E. J. Candès, J. K. Romberg, and T. Tao. Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52(2):489–509, 2006.

[10] E. J. Candès and T. Tao. Decoding by linear programming. *IEEE Transactions on Information Theory*, 51(12):4203–4215, 2005.

[11] E. J. Candès and T. Tao. Near-optimal signal recovery from random projections: Universal encoding strategies? *IEEE Transactions on Information Theory*, 52(12):5406–5425, 2006.

[12] T.-H. H. Chan, E. Shi, and D. Song. Private and continual release of statistics. In *ICALP (2)*, pages 405–417, 2010.

[13] G. Cormode, C. M. Procopiuc, D. Srivastava, and T. T. L. Tran. Differentially private publication of sparse data. *CoRR*, abs/1103.0825, 2011.

[14] A. De. Lower bounds in differential privacy. *arXiv: 1107.2183*, 2011.

[15] R. A. DeVore and G. G. Lorentz. *Constructive Approximation*. Springer-Verlag, 1993.

[16] I. Dinur and K. Nissim. Revealing information while preserving privacy. In *PODS*, pages 202–210, 2003.

[17] D. L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006.

[18] C. Dwork. Differential privacy. In *ICALP (2)*, pages 1–12, 2006.

[19] C. Dwork. A firm foundation for private data analysis. *Commun. ACM*, 54(1):86–95, 2011.

[20] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor. Our data, ourselves: Privacy via distributed noise generation. In *EUROCRYPT*, pages 486–503, 2006.

[21] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, pages 265–284, 2006.

[22] C. Dwork, F. McSherry, and K. Talwar. The price of privacy and the limits of lp decoding. In *STOC*, pages 85–94, 2007.

[23] C. Dwork, M. Naor, T. Pitassi, and G. N. Rothblum. Differential privacy under continual observation. In *STOC*, pages 715–724, 2010.

[24] C. Dwork, M. Naor, T. Pitassi, G. N. Rothblum, and S. Yekhanin. Pan-private streaming algorithms. In *ICS*, pages 66–80, 2010.

[25] J. Gehrke, E. Lui, and R. Pass. Towards privacy for social networks: A zero-knowledge based definition of privacy. In *TCC*, pages 432–449, 2011.

[26] A. Ghosh, T. Roughgarden, and M. Sundararajan. Universally utility-maximizing privacy mechanisms. In *STOC*, pages 351–360, 2009.

[27] M. Hardt and K. Talwar. On the geometry of differential privacy. In *STOC*, pages 705–714, 2010.

[28] M. Hay, V. Rastogi, G. Miklau, and D. Suciu. Boosting the accuracy of differentially private histograms through consistency. *PVLDB*, 3(1):1021–1032, 2010.

[29] 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), 2008.

[30] C. Li, M. Hay, V. Rastogi, G. Miklau, and A. McGregor. Optimizing linear counting queries under differential privacy. In *PODS*, pages 123–134, 2010.

[31] T. T. Y. Lin and F. J. Herrmann. Compressed wavefield extrapolation. *Geophysics*, 72(5):SM77–SM93, 2007.

[32] K. Liu, H. Kargupta, and J. Ryan. Random projection-based multiplicative data perturbation for privacy preserving distributed data mining. *IEEE Trans. Knowl. Data Eng.*, 18(1):92–106, 2006.

[33] M. Lustig, D. Donoho, and J. M. Pauly. Sparse mri: The application of compressed sensing for rapid mr imaging. *Magnetic Resonance in Medicine*, 58(6):1182–1195, 2007.

[34] F. McSherry and I. Mironov. Differentially private recommender systems: Building privacy into the netflix prize contenders. In *KDD*, pages 627–636, 2009.

[35] F. McSherry and K. Talwar. Mechanism design via differential privacy. In *FOCS*, pages 94–103, 2007.

[36] S. Nagaraja, V. Jalaparti, M. Caesar, and N. Borisov. Private anomaly detection across isp networks. In *PETS*, 2010.

[37] A. Narayanan and V. Shmatikov. Robust de-anonymization of large sparse datasets. In *IEEE Symposium on Security and Privacy*, pages 111–125, 2008.

[38] D. Needell and J. A. Tropp. Cosamp: iterative signal recovery from incomplete and inaccurate samples. *Commun. ACM*, 53(12):93–100, 2010.

[39] V. Rastogi and S. Nath. Differentially private aggregation of distributed time-series with transformation and encryption. In *SIGMOD Conference*, pages 735–746, 2010.

[40] A. Roth and T. Roughgarden. Interactive privacy via the median mechanism. In *STOC*, pages 765–774, 2010.

[41] A. C. Sankaranarayanan, P. K. Turaga, R. G. Baraniuk, and R. Chellappa. Compressive acquisition of dynamic scenes. In *Proceedings of the 11th European conference on Computer vision: Part I*, ECCV'10, pages 129–142, Berlin, Heidelberg, 2010. Springer-Verlag.

[42] M. Sheikh, S. Sarvotham, O. Milenkovic, and R. Baraniuk. Dna array decoding from nonlinear measurements by beleif propagation. In *IEEE Workshop on Statistical Signal Processing (SSP)*, 2007.

[43] G. Taubock and F. Hlawatsch. A compressed sensing technique for ofdm channel estimation in mobile environments: Exploiting channel sparsity for reducing pilots. In *IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, 2008.

[44] J. Tropp, M. Wakin, M. Duarte, D. Baron, and R. Baraniuk. Random filters for compressive sampling and reconstruction. In *IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, 2006.

[45] R. Wang, Y. Li, X. Wang, H. Tang, and X. Zhou. Learning your identity and disease from research papers: Information leaks in genome wide association study. In *ACM CCS*, 2009.

[46] X. Xiao, G. Wang, and J. Gehrke. Differential privacy via wavelet transforms. In *ICDE*, pages 225–236, 2010.

[47] S. Zhou, J. D. Lafferty, and L. A. Wasserman. Compressed and privacy-sensitive sparse regression. *IEEE Transactions on Information Theory*, 55(2):846–866, 2009.

[48] S. Zhou, K. Ligett, and L. Wasserman. Differential privacy with compression. In *Proceedings of the 2009 IEEE international conference on Symposium on Information Theory*, pages 2718–2722, Piscataway, NJ, USA, 2009.

A Proof of Lemma 4

First let us focus on the sampling process, which can be characterized as a random projection $g : X \to \mathbb{R}^k$. For any $\boldsymbol{D} \in X$, $g(\boldsymbol{D}) = \boldsymbol{y} = \boldsymbol{\Phi D}$. We have mentioned that the random matrix $\boldsymbol{\Phi}$ is formed by sampling i.i.d. entries from a symmetric Bernoulli distribution; more accurately,

$$Prob(\boldsymbol{\Phi}(i,j) = \pm 1/\sqrt{k}) = 1/2.$$

The sensitivity of g is $\Delta_g = \sqrt{k}$. In the compressive mechanism, we make use of a Laplacian mechanism $\mathcal{K}_g : X \to \mathbb{R}^k$, which is ϵ-differentially private according to Lemma 2.

The subsequent reconstruction process of compressed sensing is a deterministic process and does not involve probability. Thus the whole compressive mechanism is ϵ-differentially private.

B Proof of Lemma 5

Each $\boldsymbol{e}[i]$ is distributed according to

$$Lap(\sqrt{k}/\epsilon) = Lap(\lambda).$$

We define a random variable $Y = \sum_{i=1}^{k} \boldsymbol{e}[i]^2$. We can compute that $E(Y) = 2k\lambda^2$ and that $Var(Y) = 20k\lambda^4$. By the Chebyshev inequality[2],

$$Prob(|Y| \le 2\lambda^2(k + \sqrt{2k/\delta})) \ge 1 - \delta.$$

Along with the fact that $||\boldsymbol{e}||_2 = \sqrt{Y}$, we see that with probability at least $1 - \delta$,

$$||\boldsymbol{e}||_2 = O(k/(\epsilon\delta^{1/4})).$$

Recall that with probability at least $1 - \exp(-k)$, $\boldsymbol{A} = \boldsymbol{\Phi\Psi} \in \mathbb{R}^{k \times n}$ satisfies RIP. Using a union bound and Corollary 1, with high probability,

$$||\boldsymbol{D} - \boldsymbol{D}^*||_2 = O(\log(n)/\epsilon).$$

C Proof of Theorem 2

The ϵ-differential pan-privacy of CMCO is straightforward: at time t, we estimate $\sum_{j=1}^{t} \boldsymbol{u}_j[i]$ in an ϵ-differentially pan-private way and any operation after this step preserves pan-privacy.

Suppose that at each time t, each entry $\boldsymbol{v}_t[i]$ of the vector $\boldsymbol{v}_t \in \mathbb{R}^k$ represents the sum of the first t entries in $\boldsymbol{m}_i \in \mathbb{R}^t$. We apply a continual mechanism to each $\boldsymbol{m}_i \in \mathbb{R}^t$. For two neighboring inputs $\boldsymbol{D}_t, \boldsymbol{D}'_t \in \mathbb{R}^t$ with $||\boldsymbol{D}_t - \boldsymbol{D}'_t||_1 \le 1$, their corresponding $\boldsymbol{m}_i, \boldsymbol{m}'_i \in \mathbb{R}^t$ satisfies $||\boldsymbol{m}_i - \boldsymbol{m}'_i||_1 \le 1/\sqrt{k}$. Combining this fact with Lemma 3, we know that with probability at least $1 - \beta$,

$$|\boldsymbol{v}_t[i] - \boldsymbol{v}_t^*[i]| = O(\log(1/\beta)\log^{1.5}(T)/(\epsilon\sqrt{k})),$$

for each $i \in [1, \ldots, k]$. By a union bound, with probability at least $1 - \beta$,

$$|\boldsymbol{v}_t[i] - \boldsymbol{v}_t^*[i]| = O(\log(k/\beta)\log^{1.5}(T)/(\epsilon\sqrt{k}))$$

---

[2]The Chernoff bound cannot be applied, as the integral defining the moment-generating function of Y does not converge unconditionally in a neighborhood of 0.

holds for all i simultaneously. Along with the Chebyshev inequality, we have that with probability at least $1 - \beta - \delta$,

$$||\boldsymbol{v}_t - \boldsymbol{v}_t^*||_2 = O(\log(k/\beta) \log^{1.5}(T)/(\epsilon \delta^{1/4})).$$

Then by Corollary 1 and a union bound, we obtain the result that with high probability,

$$||\boldsymbol{D}_t - \boldsymbol{D}_t^*||_2 = \tilde{O}(\log^{1.5}(T)/\epsilon).$$

D   More on Reconstruction Process

In the paper, we just used the reconstruction process in a black-box manner. Here we discuss the two reconstruction algorithms mentioned [7, 38] more concretely.

Remember the equation we have mentioned:

$$\boldsymbol{y}^* = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{e}, \tag{2}$$

where $\boldsymbol{A} \in \mathbb{R}^{k \times n}$, $\boldsymbol{e}, \boldsymbol{y}^* \in \mathbb{R}^k$ and $\boldsymbol{x} \in \mathbb{R}^n$. $\boldsymbol{A}$ and $\boldsymbol{y}^*$ are known, $\boldsymbol{e}$ is some unknown noise and we want to know what $\boldsymbol{x}$ is (reconstruct $\boldsymbol{x}$).

What [7] does is to solve the convex program

$$\min_{\boldsymbol{x}^* \in \mathbb{R}^n} ||\boldsymbol{x}^*||_1 \qquad s.t. ||\boldsymbol{A}\boldsymbol{x}^* - \boldsymbol{y}||_2 \leq \theta,$$

where θ bounds the amount of noise $\boldsymbol{e}$.

[38] is more complicated by using a greedy algorithm to solve (2). The essential idea is to iteratively approximate the target vector $\boldsymbol{x}$. At each step, the current approximation induces a residual, the part of the target vector that has not been approximated. Samples are updated so that they reflect the current residual and are used to construct a proxy for the residual, allowing us to identify the large components in the residual. This step yields a tentative support for the next approximation and samples are used to estimate the approximation on this support set using least squares. The process is repeated until recoverable energy is met in the vector.

E   More on Continual Mechanism

We also used continual mechanism [23] in a black-box way in the body of the paper. Here we discuss the content of the mechanism.

Continual mechanism receives an input $\boldsymbol{\sigma}(t) \in \{0, 1\}$ at each time $t \in [T]$, and outputs an approximation to the number of 1's seen in the length t.

There is a preprocessing step: segmentation. For $i \in [\log T]$ (assume T to be a power of 2 without loss of generality), associate with each string $s \in \{0, 1\}^i$ the time segment S of $2^{\log T - i}$ time periods $\{s \circ 0^{\log T - i}, \ldots, s \circ 1^{\log T - i}\}$. The segment begins in time $s \circ 0^{\log T - i}$ and ends in time $s \circ 1^{\log T - i}$.

Then comes the processing steps. At each time $t \in [T]$, continual mechanism maintains $\log T$ segments that contain $t - 1$. Each of these $\log T$ segments has a noise associated with it sampling from $Lap((1 + \log T)/\epsilon)$. The output at each time $t \in [T]$ is the summation of the noise from these $\log T$ segments and the count (true count plus a random noise sampling from $Lap((1 + \log T)/\epsilon)$).