Accurate and Efficient Private Release of Datacubes and Contingency Tables

Grigory Yaroslavtsev
Pennsylvania State University
grigory@cse.psu.edu

Graham Cormode Cecilia M. Procopiuc Divesh Srivastava
AT&T Labs – Research
{graham,magda,divesh}@research.att.com

***Abstract*— A central problem in releasing aggregate information about sensitive data is to do so accurately while providing a privacy guarantee on the output. Recent work focuses on the class of *linear queries*, which include basic counting queries, data cubes, and contingency tables. The goal is to maximize the utility of their output, while giving a rigorous privacy guarantee. Most results follow a common template: pick a "strategy" set of linear queries to apply to the data, then use the noisy answers to these queries to reconstruct the queries of interest. This entails either picking a strategy set that is hoped to be good for the queries, or performing a costly search over the space of all possible strategies.**

In this paper, we propose a new approach that balances accuracy and efficiency: we show how to improve the accuracy of a given query set by answering some strategy queries more accurately than others. This leads to an efficient optimal noise allocation for many popular strategies, including wavelets, hierarchies, Fourier coefficients and more. For the important case of marginal queries we show that this strictly improves on previous methods, both analytically and empirically. Our results also extend to ensuring that the returned query answers are consistent with an (unknown) data set at minimal extra cost in terms of time and noise.

I. Introduction

The long-term goal of much work in data privacy is to enable the release of information that accurately captures the behavior of an input data set, while preserving the privacy of individuals described therein. There are two central, interlinked questions to address around this goal: what privacy properties should the transformation process possess, and how can we ensure that the output is useful for subsequent analysis and processing? The model of Differential Privacy has lately gained broad acceptance as a criterion for private data release [7], [9]. There are now multiple different methods which achieve Differential Privacy over different data types [1], [2], [4], [6], [11], [12], [14], [16], [17], [22]. Some provide a strong utility guarantee, while others demonstrate their utility via empirical studies. These algorithms also vary from the highly practical, to taking time exponential in the data size.

The output of the data release should be compatible with existing tools and processes in order to provide usable results. The model of contingency tables is universal, in that any relation can be represented exactly in this form. That is, the contingency table of a dataset over a subset of attributes contains, for each possible attribute combination, the number of tuples that occur in the data with that set of attribute values. In this paper, we call such a contingency table the *marginal* (distribution) of the database over the respective subset of attributes. The set of all possible marginals for a relation is captured by the data cube. Contingency tables and the data cube in turn are examples of a more general class of linear queries, i.e., each query is a linear combination of the entries of the contingency table over all attributes in the input relation.

There has been much interest in providing methods to answer such linear queries with privacy guarantees. In this paper, we argue that these all fit within a general framework: answer some set of queries S over the data (not necessarily the set that was requested), with appropriate noise added to provide the privacy, then use the answers to answer the given queries A limitation of prior work based on a fixed strategy S is that it applies *uniform noise* to the answers: the same magnitude of noise is added to each query in S. However, it turns out that the accuracy can be much improved by using *non-uniform noise*: using different noise for each answer, while providing the same overall guarantee. The main contribution of this paper is to provide a full formal understanding of this problem and the role that non-uniform noise can play.

Example. Figure 1(a) shows a table with 3 binary attributes A, B and C. As in prior work [16], we think of a database $\mathcal{D}$ as an N-dimensional vector $x \in \mathbb{R}^N$, where N is the domain size of $\mathcal{D}$; i.e., if $\mathcal{D}$ has attributes $A_1, \ldots, A_d$, then $N = \Pi_{i=1}^{d} |A_i|$. We linearize the domain of $\mathcal{D}$, so that each index position i, $1 \leq i \leq N$, corresponds to a unique combination α of attribute values, and x_i is the number of tuples in $\mathcal{D}$ that have values α. In Figure 1(a), we linearized the domain in the order $000, 001, \ldots, 111$. Here, position $i = 2$ corresponds to the combination of values $\alpha = 001$. Thus, $x_2 = 2$ since $\mathcal{D}$ contains two tuples (1 and 4) with these values.

Suppose that we want to compute two marginals over $\mathcal{D}$: the marginal over A, and the marginal over A, B. The query marginals can be represented as a matrix Q, as depicted in Figure 1(b), so that the answer is Qx: The first two rows compute the marginal over A; i.e., the first row is the linear query that counts all tuples t with $t.A = 0$; while the second row counts all tuples with $t.A = 1$. Similarly, the third row counts all tuples with $t.A = 0$ and $t.B = 0$, and so on. Differentially private mechanisms answer Q in the form of $y = Qx + \tau$, where τ is a random vector whose distribution provides a certain level of privacy. The error of the answer is generally defined as the variance $\mathrm{Var}(y)$ [16], [22].

$\mathcal{D}$:

tid	A	B	C
1	0	0	1
2	0	1	1
3	0	0	0
4	0	0	1
5	1	1	0

x = (1,2,0,1,0,0,1,0)

(a) Table $\mathcal{D}$ for vector x

$$Q=\begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

(b) Query Q for marginals on A and A, B

$$S=\begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

(c) Strategy matrix S

$$R=\begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

(d) Recovery matrix R

Fig. 1. Example contingency table, query matrix, with strategy and recovery matrices.

For example, one way to provide ε-differential privacy adds uniform noise to each answer. Based on the structure of Q in Figure 1(b), we can add noise with variance $\frac{8}{\varepsilon^2}$ to each answer; see details in Section II). Over the six queries, the sum of variances is $\frac{48}{\varepsilon^2}$. However, we can do better with a non-uniform approach. For example, we can add noise with variance $2(\frac{9}{4\varepsilon})^2$ to the answers for the first two rows of Q, and noise with variance $2(\frac{9}{5\varepsilon})^2$ to the remaining four answers, and still provide ε-differential privacy. The sum of the six variances is then $2\cdot 2(\frac{9}{4\varepsilon})^2 + 4\cdot 2(\frac{9}{5\varepsilon})^2 = 46.17/\varepsilon^2$. We can improve this even further by changing how we answer the queries: we can answer the first query Q_1 by taking half of the first answer, and adding half of the third and fourth answers. The resulting variance of Q_1 is

$$\tfrac{1}{4}\cdot 2\big(\tfrac{9}{4\varepsilon}\big)^2 + \tfrac{1}{4}\cdot 2\big(\tfrac{9}{5\varepsilon}\big)^2 + \tfrac{1}{4}\cdot 2\big(\tfrac{9}{5\varepsilon}\big)^2 = 5.77/\varepsilon^2.$$

Similar tricks yield the same variance for all other answers, so the sum of all six variances is now $34.6/\varepsilon^2$, a 28% reduction over the uniform approach. ■

This example shows that we can significantly improve the accuracy of our answers while preserving the same level of privacy by adopting non-uniform noise and careful combination of intermediate answers to give the final answer. Yet further improvement can result by choosing a different set of queries to obtain noisy answers to. The problem we address in this paper is how to use these techniques to efficiently and accurately provide answers to such queries Q that meet the differential privacy guarantee. This captures the core problems of releasing data cubes, contingency tables and marginals. Our results are more general, as they apply to arbitrary sets of linear queries Q, but our focus is on these important special cases. We also discuss how to additionally ensure that the answers meet certain consistency criteria, i.e. there is some x such that the query answers are Qx. Next, we study how existing techniques can be applied to this problem, and discuss their limitations.

The Strategy/Recovery approach. Mechanisms for minimizing the error of linear counting queries under differential privacy have attracted a lot of attention. Work in the theory community [2], [10], [11], [12], [13], [21] has focused on providing the best bounds on noise for an arbitrary set of such queries, in both the online and offline setting. However, these mechanisms are rarely practical for large databases with moderately high dimensionality: they can scale exponentially with the size of their input.

Work in database research has aimed to deliver methods that scale to realistic data sizes. Much of this work builds on basic primitives in differential privacy such as adding appropriately scaled noise to a numeric quantity from a specific random distribution (see Section II). Repeating this process for multiple different quantities, and reasoning about how the privacy guarantees compose, it is possible to ensure that the full output meets the privacy definition. The goal is then to minimize the error introduced into the query answers (as measured by their variance) while satisfying the privacy conditions.

Given this outline, we observe that the bulk of methods using noise addition fit into a two-step framework that we dub the ‘strategy/recovery’ approach:

- **Step 1.** Find a *strategy matrix* S and compute the vector $z = Sx+\nu$, where ν is a random *noise vector* drawn from an appropriate distribution. Then z is the differentially private answer to the queries represented by S.
- **Step 2.** Compute a *recovery matrix* R, such that $Q = RS$. Return $y = Rz$ as the differentially private answer to the queries Q. The variance $\mathrm{Var}(y)$ is often used as an error measure for the approach.

We show this method schematically in Figure 2. For example, Figures 1(c) and 1(d) show a possible choice of matrices S and R for the query matrix Q in Figure 1(b). In this case, the strategy S computes the marginal on A, B; Step 1 above adds random noise independently to all cells in this marginal. The recovery R computes the marginal on A by aggregating the corresponding noisy cells from the marginal on A, B (the first two rows of R), and also outputs the marginal on A, B (the last four rows of R).

We now show how prior work fits into this approach. In many cases, the first step directly picks a fixed matrix for S, by arguing that this is suitable for a particular class of queries Q. For example, when setting $S = I$ (hence $R = Q$), the approach computes a set of noisy counts $\tilde{x}_i$ by adding Laplace noise independently to each x_i. The answer to any query matrix Q is computed over these noisy counts, i.e., $y = Q\tilde{x}$; this model was analyzed in [1]. By contrast, when $S = Q$ (and $R = I$), as discussed in [7], the approach adds noise to the result of each query in Q, i.e., $y = Qx + \nu$.

Several more sophisticated strategies have been designed, with the goal of minimizing the error $\mathrm{Var}(y)$ for various query workloads. When Q consists of low-dimensional range queries, [22] proposes S to be the wavelet transform, while [14] studies the strategy S corresponding to a hierar-

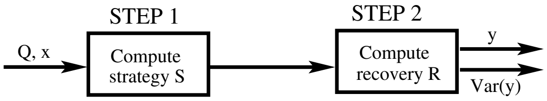


Fig. 2. Framework of prior work.

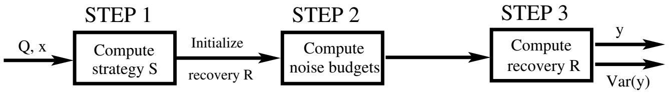


Fig. 3. Our proposed framework.

chical structure over x. However, as shown in [16], neither of these strategies is particularly accurate for other types of queries. For marginals, [1] chooses S to be the Fourier transform matrix, and [6] employs a clustering algorithm over the queries to compute S. Figures 1(c) and 1(d) depict the output computed via [6] on query matrix Q (Figure 1(b)). Other work has suggested the use of random projections as the strategy matrix, connecting to the area of sparse recovery [5], [18]. Many of these choices are relatively fast: that is, S and S^{-1} can be applied to a vector of length N in time $O(N)$ or $O(N \log N)$ in the case of wavelet and Fourier transforms, respectively. This is important, since real data can have large values of N, and so asymptotically higher running time may not be practical. A limitation of [6] is that the clustering step is very expensive, limiting the scalability of the approach.

An important technical distinction for the strategy/recovery approach is whether or not the strategy S is invertible. If it is (e.g., when S is the Fourier or wavelet transform), then the recovery matrix $R = QS^{-1}$ is unique, and the query answer y is guaranteed to be consistent (see Definition 2.3). Then the error measure $\mathrm{Var}(y)$ depends only on S (and Q). However, if S is not invertible, then there can be many choices for R, and $\mathrm{Var}(y)$ depends on both S and R. The optimal recovery R that minimizes $\mathrm{Var}(y)$ (for a fixed S) can be computed via the least squares method [14], [16] and $\mathrm{Var}(y)$ has a closed-form expression as a function of S. Using this fact, Li et al. [16] study the following optimization problem: *Given queries Q and a formula for* $\mathrm{Var}(y)$ *as a function of S, compute the strategy S that minimizes* $\mathrm{Var}(y)$. This is a tough optimization, since the search is over all possible strategy matrices S. Their *matrix mechanism* uses a rank-constrained semidefinite program (SDP) to compute the optimal S. Solving this SDP is very costly as a function of N, making it impractical for data with more than a few tens of entries[1].

In summary, the search for a strategy matrix S is currently done either by picking one that we think is likely to be "good" for queries Q, or by solving an SDP, which is impractical even for moderate size problems.

Our Contributions. Most of the prior approaches discussed above use the uniform "noise budgeting" strategy, i.e., each value ν_i of the noise vector is (independently) drawn from the same random distribution. The scaling parameter of this distribution depends on the desired privacy guarantee ε, as well as the "sensitivity" of the strategy matrix S (see Section II).

In the extended version of [16], the authors prove that any non-uniform noise budgeting strategy can be reduced to a uniform budgeting strategy by scaling the rows of S with different factors. However, computing the optimal scaling factors this way is impractical, as it requires solving an SDP. The only efficient method for computing non-uniform noise budgets we are aware of applies to the special case when Q is a range query workload [4]. There, S corresponds to a multi-dimensional hierarchical decomposition, and recovery R corresponds to the greedy range decomposition. The resulting budgeting is not always optimal.

In this paper we show how to compute the *optimal* noise budgets in time at most linear in the sizes of R and S, for a large class of queries Q (including marginal queries), and for most of the matrices S considered in prior work. This includes the Fourier transform, the wavelet transform, the hierarchical structure over x, and any strategy consisting of a set of marginals (in particular, the clustering strategy of [6]).

The overall framework introduced is depicted in Figure 3: Given strategy matrix S and recovery matrix R, we compute optimal noise budgets ε_i for each query, and draw each noise value ν_i from a random distribution that depends on ε_i (Step 2). We then derive a new recovery matrix R that minimizes $\mathrm{Var}(y)$ (Step 3), for the noise budgets computed in Step 2.

The most general approach would be to provide a mathematical formulation for the following global optimization problem: *Given the query matrix Q, compute the strategy S, the recovery R, and the noise budgets ε_i that minimize* $\mathrm{Var}(y)$. However, this problem essentially reduces to that addressed by the matrix mechanism [16], and requires solving an SDP.

Instead, we study how to efficiently solve optimization problems where two out of the three parameters S, R and $\{\varepsilon_i\}_i$ are fixed. In Section III-A, we solve the optimization problem: *Given a decomposition of query matrix Q into strategy S and recovery R, compute the optimal noise budgets ε_i that minimize* $\mathrm{Var}(y)$. We provide a formula for $\mathrm{Var}(y)$, as a function of S and R. In Section III-B, we apply the generalized least squares method to solve the following problem: *Given the query matrix Q, the strategy S, and the noise budgets ε_i, compute the recovery R that minimizes* $\mathrm{Var}(y)$. Following the steps in this framework provides efficient algorithms with low error—the time overhead is less than 1 second in our experiments (Section V). We also consider ways to generate a consistent output y of Step 3 with small (but non-optimal) error under other metrics; see Sections III-C and IV-C. Our approach strictly improves over the previous result from [6].

Section IV shows that for the subclass of linear queries that has been most extensively studied in the differential privacy literature, which is the set of low-order k-way marginals, our approach achieves state-of-the-art bounds on the running time while giving provably better bounds on the magnitude of noise per query. This follows from the analysis of the optimum non-

[1] Independent work has tried to improve the efficiency of such mechanisms via convex optimization and spectral methods [23], [17]

uniform noise allocation in the Fourier basis. This basis plays an important role in the analysis of k-way marginals because it allows to express them without redundancy. Using Fourier basis we achieve consistency for noise allocation strategies in other bases more efficiently than the LP-based approach of [1].

To summarize, our contributions are as follows:

- We propose a framework for minimizing the error of differentially private answers, via a combination of noise budgeting and computation of an improved recovery matrix. This improves on the accuracy of existing strategies, at minimal computation cost.
- We develop fast algorithms within this framework for marginal queries. Our algorithms compute consistent answers. In particular, when Q is the set of all k-way marginals, we give asymptotic bounds on the error of our mechanism; we are not aware of any such analysis for the matrix mechanism. As a by-product, our analysis also improves the error bound for the uniform noise case.
- We conduct an extensive experimental study on marginal query workloads and show that our framework reduces the error of existing strategies (including the Fourier strategy [1] and the Cluster strategy [6]).

Organization. Section II introduces the necessary definitions for describing our framework. The optimization results required by Steps 2 and 3 are developed in Section III. Section IV decribes the application of our framework to marginal queries, giving bounds on error and consistent results. Our experimental study is presented in Section V, and we conclude in Section VI.

II. Definitions

We begin by recalling the definition of differential privacy and some fundamental mechanisms which satisfy this definition.

Definition 2.1 (Differential privacy [9], [8]): A randomized algorithm $\mathcal{A}$ satisfies (ϵ, δ)-differential privacy if for all databases D_1 and D_2 differing in at most one element, and all measurable subsets $S \subseteq \mathrm{Range}(\mathcal{A})$,

$$\Pr[\mathcal{A}(D_1) \in S] \leq e^{\epsilon} \cdot \Pr[\mathcal{A}(D_2) \in S] + \delta.$$

We say that an algorithm satisfies ϵ-differential privacy if it satisfies $(\epsilon, 0)$-differential privacy.

Definition 2.2 (L^p-sensitivity): For $p \geq 1$ let the L^p-sensitivity $\Delta_p(f)$ of a function $f \colon D \to \mathbb{R}^q$ be defined as:

$$\Delta_p(f) = \max_{D_1, D_2} \|f(D_1) - f(D_2)\|_p,$$

for all D_1 and D_2 differing in at most one element. Here, $\|\cdot\|_p$ denotes the standard L^p norm, i.e., $\|x\|_p = (\sum_{i=1}^n |x_i|^p)^{1/p}$ for $x \in \mathbb{R}^n$.

We rely on the following two basic mechanisms to construct differentially private algorithms:

Theorem 2.1 (Laplace mechanism [9]): If f is a function $f \colon D \to \mathbb{R}^q$, then releasing f with additive q-dimensional Laplace noise with variance $2\left(\frac{\Delta_1(f)}{\epsilon}\right)^2$ in each component satisfies ϵ-differential privacy.

Theorem 2.2 (Gaussian mechanism [8], [19]): If f is a function $f \colon D \to \mathbb{R}^q$, then releasing f with additive q-dimensional Gaussian noise with variance $\left(2\Delta_2^2(f)\frac{\log(2/\delta)}{\epsilon^2}\right)$ in each component satisfies (ϵ, δ)-differential privacy.

Query workloads, consistency, strategy and recovery. As mentioned in Section I, we represent the database as a vector $x \in \mathbb{R}^N$ and the query workload as a matrix $Q \in \mathbb{R}^{q \times N}$: each row $Q_{i\cdot}$, $1 \leq i \leq q$, is a linear query over database x. It is easy to see that the sensitivity of Q is $\Delta_p(Q) = \max_{j=1}^N \|Q_{\cdot j}\|_p$, where $Q_{\cdot j}$ denotes the jth column of Q.[2] One differentially private answer to Q is a vector $y = Qx + \tau$, where $\tau \in \mathbb{R}^q$ is the noise vector drawn from an appropriate (Laplace or Gaussian) distribution. Our formal goal is to minimize the variance of a given linear functional $a^T \cdot \mathrm{Var}(y)$ for some fixed vector $a \in \mathbb{R}_+^q$, while guaranteeing differential privacy. For example, if $a = \overrightarrow{\mathbf{1}}$ we minimize the sum of the variances of noise over all queries. In particular, we study workloads Q that consist of marginals over x, such as the set of all k-way marginals, for some small integer k.

Definition 2.3: A noisy output $y = Qx + \tau$ is *consistent* if there exists at least one vector x^c such that $y = Qx^c$.

We decompose a query workload Q into a *strategy* matrix $S \in \mathbb{R}^{m \times N}$, and a *recovery* matrix $R \in \mathbb{R}^{q \times m}$, such that $Q = RS$. The query answer y is then computed as $y = Rz$, where $z = Sx + \nu$ is the noisy answer to S (hence, $\tau = R\nu$). In general, there are many possible ways to pick R and S given Q, and our goal will be to minimize the resulting $\mathrm{Var}(y)$.

III. Our Framework

In this section we solve the optimization problems required by Steps 2 and 3 of our framework from Figure 3.

A. Optimal Noise Budgeting (Step 2)

A novel part of our scheme is a special purpose *budgeting mechanism*: For each row $S_{i\cdot}$ in the strategy S, we release $z_i = S_{i\cdot}x + \nu_i$, where ν_i is drawn from a Laplace distribution that depends on a value ε_i. We show how to choose the values ε_i optimally so that the overall method satisfies ε-differential privacy and the resulting noise is minimized. We also design an approach based on *grouping* rows of the strategy matrix S, which allows us to compute the optimal ε_i's efficiently.

Proposition 3.1: Let S be an $m \times N$ strategy matrix, and let $\varepsilon_1, \ldots, \varepsilon_m$ be a set of m non-negative values. Define the noisy answer to S to be an m-dimensional vector z such that $z_i = S_{i\cdot}x + \nu_i$, $1 \leq i \leq m$.

(i) If ν_i is drawn from the Laplace distribution with variance $\frac{2}{\varepsilon_i^2}$, then z satisfies α-differential privacy, where $\alpha = 2\max_{j=1}^N(\sum_{i=1}^m |S_{ij}|\varepsilon_i)$.

(ii) If ν_i is drawn from the Gaussian distribution with variance $2\frac{\log(2/\delta)}{\varepsilon_i^2}$, and $\alpha = 2\max_{j=1}^N \sqrt{\sum_{i=1}^m S_{ij}^2 \varepsilon_i^2}$, z satisfies (α, δ)-differential privacy.

[2] We assume that each individual contributes a weight of 1 to some entry of x, in line with prior work. Other cases can be handled by rescaling the sensitivity accordingly.

Proof: We decompose S as $D^{-1}DS$ where D is the diagonal matrix $D = \operatorname{diag}(\varepsilon_1, \dots \varepsilon_m)$. We now consider the L^p sensitivity of the function $f(x) = (DS)x$. From Definition 2.2, we have

$$\Delta_p(f) \leq 2 \max_{j=1}^{N} \|(DS)_{\cdot j}\|_p = 2 \max_{j=1}^{N} (\sum_{i=1}^{m} |S_{ij}\varepsilon_i|^p)^{1/p}$$

Thus, adding noise with variance proportional to $(2\frac{\Delta_p(f)}{\alpha})^2$ provides α-differential privacy (via Theorem 2.1 with $p = 1$) or (α, δ)-differential privacy (via Theorem 2.2 with $p = 2$). Finally, multiplying by D^{-1} has the effect of rescaling the variance in each component: the ith component now has variance proportional to $(\frac{\Delta_p(f)}{\alpha\varepsilon_i})^2$. Setting $\alpha = \Delta_p(f)$ for $p = 1$ or $p = 2$ and applying the correct scaling constants gives the claimed result. ■

Recall that the output is computed as $y = Rz$. Our goal is to choose values ε_i that minimize the variance $a^T \operatorname{Var}(y) = a^T \operatorname{Var}(R\nu)$. We detail this for Laplace mechanism:

$$a^T \cdot \operatorname{Var}(R\nu) = 2\sum_{i=1}^{q} a_i \sum_{j=1}^{m} \frac{R_{ij}^2}{\varepsilon_j^2} = 2\sum_{i=1}^{m} \frac{1}{\varepsilon_i^2} \sum_{j=1}^{q} a_j R_{ji}^2.$$

Let $b_i = 2\sum_{j=1}^{q} a_j R_{ji}^2$. By Proposition 3.1, it follows that the optimal noise budgeting $\{\varepsilon_i\}$ is the solution to the following optimization problem:

$$\text{Minimize: } \sum_{i=1}^{m} \frac{b_i}{\varepsilon_i^2} \quad (1)$$

$$\text{Subject to: } \sum_{i=1}^{m} |S_{ij}|\varepsilon_i \leq \varepsilon, \quad 1 \leq j \leq N. \quad (2)$$

$$\epsilon_i \geq 0, \quad 1 \leq i \leq m \quad (3)$$

Because all b_i's are non-negative, the objective function is convex. The body defined by the linear inequalities is also convex. The resulting problem can thus be solved using a convex optimization package that implements, e.g., interior point methods. Such methods require time polynomial in m, N, and the required accuracy of the solution [3].

Efficient Solution via Grouping. Convex optimization solvers may require a large number of iterations and be too inefficient for databases of moderate dimensionality. However, for most of the frequently used strategy matrices, the optimization problem can be significantly simplified, if we partition the rows of the strategy matrix S into *groups*, and define the corresponding values ε_i to be the same for all rows in a group. We show that the groups can be chosen in such way that all conditions $\sum_{i=1}^{m} |S_{ij}|\varepsilon_i \leq \varepsilon$ become *identical* once we set the ε_i's to be equal in each group, which leads to a closed form solution. This approach was implicitly used in [4]. We show that this concept can be applied to a larger class of strategy matrices. The optimal solution for the simplified problem is a feasible solution for the general problem. If recovery matrix R satisfies a certain property (as is the case for *all* matrices we consider), then the optimal solution for the simplified problem is also guaranteed optimal for the general case. In particular, we find optimal noise budgets for strategy/recovery methods such as Fourier [1] and clustering [6].

Definition 3.1: Let S be an $m \times N$ strategy matrix. We say that S satisfies the *grouping property* if there exists a grouping function over its rows $G : [m] \to [g]$, $g \leq m$, such that the following two conditions are satisfied:

— row-wise disjointness: for any two rows i_1, i_2 of S with $G(i_1) = G(i_2)$ and for any column j, $S_{i_1 j} S_{i_2 j} = 0$;

— bounded column norm: for any group r, and for any two columns j_1, j_2, we have $\max_{i:G(i)=r} |S_{ij_1}| = \max_{i:G(i)=r} |S_{ij_2}| = C_r$.

The minimum g for which S has a grouping function G is called the *grouping number* of S.

Together, the two conditions in Definition 3.1 imply that any column of S contains at most one non-zero value from each group, and that its magnitude is the same (within a group) for all columns. Hence, not every S can meet this definition: while we could put every row in a singleton group, we also then require that the magnitude of all non-zero entries in the row are identical. Nevertheless, as we show below, many commonly used matrices are groupable.

Example. Matrix S in Figure 1(c) has grouping number $g = 1$: each column has exactly one entry equal to 1, so $C_1 = 1$. On the other hand, if $S = Q$ is the matrix in Figure 1(b), the grouping number is 2: we define one group containing the first two rows, and another containing the last four rows. We have $C_1 = C_2 = 1$. Note that, e.g., the first and third rows cannot be grouped together, since $Q_{11}Q_{31} = 1 \neq 0$. We now apply this definition to the other strategy matrices proposed:

Base counts. As noted in the introduction, directly materializing the noisy version of x is equivalent to $S = I$. In this case, all rows form a single group; hence, $g = 1$ and $C_1 = 1$.

Collections of marginals. When S is a set of marginals, all rows that compute the cells in the same marginal can be grouped together, as in the above example. Hence, the number of groups g is the number of marginals computed; and $C_r = 1$ for each group r.

Hierarchical structures. When S represents a hierarchy over x, all rows that compute the counts at the same level in the hierarchy form one group. Hence, the grouping number g is the depth of the hierarchy and all C_r values are 1. Specifically, when S represents a binary tree over x, the grouping number is $g = \lceil \log_2 N \rceil$. The same essentially holds for the one-dimensional Haar wavelet (here, $g = \lceil \log_2 N \rceil + 1$). For higher dimensional wavelets, the grouping number grows exponentially with the dimension of the wavelet transform.

Fourier transform. The Fourier transform (discussed in more detail in Section IV-A) is dense: every entry is non-zero and has absolute value $2^{-d/2}$. In this case, each row forms its own group, the grouping number is N, and $C_r = 2^{-d/2}$ for any group r.

Sparse random projections. Sketches are sparse random projections that partition the data x into buckets, repeated t times [5]. All entries in the sketch matrix S are $\{-1, 0, +1\}$. In this case, all rows that define one particular partition of the data form one group, so $g = t$ and $C_r = 1$.

Arbitrary strategies S. If S is groupable, we can greedily find a grouping as follows: start a group with an arbitrary row, and try to add each remaining row to existing groups; if a row cannot be added to an existing group, a new group is created for it. While this may not result in a minimum g, any grouping suffices for our purposes. We do not discuss the greedy approach further, since all the strategies we study can be grouped directly as discussed above.

Definition 3.2: Let S be an $m \times N$ strategy matrix with grouping function G. Let R be a corresponding $q \times m$ recovery matrix. We say that R is *consistent with* G if for any rows i_1, i_2 of S with $G(i_1) = G(i_2)$, we have $b_{i_1} = b_{i_2}$ (where $b_i = 2\sum_{j=1}^{q} a_j R_{ji}^2$ are as in objective function (1)).

When Q is a set of marginals and $a = \overrightarrow{\mathbf{1}}$, it is easy to verify that R is consistent with the optimal grouping of S, for all the choices of S considered in prior work: $S = I$, $S = Q$, $S =$Fourier transform, and $S =$ strategy marginals computed by clustering [6] (here, R aggregates cells of the centroid marginal to compute each of the marginals assigned to a cluster).

The next result follows directly from the properties of the grouping function.

Lemma 3.2: Let S be a strategy matrix with grouping function G. There is a feasible solution to the optimization problem (1) – (3) such that for each group r and for all pairs of rows i_1, i_2 with $G(i_1) = G(i_2) = r$, we have $\varepsilon_{i_1} = \varepsilon_{i_2}$. Moreover, all privacy conditions (2) are equivalent, and can be satisfied with equality.

If R is consistent with G, then the above solution is optimal for the problem defined by (1) – (3).

Proof: Let $\eta = \eta_1, \ldots, \eta_g$ be the noise budgets corresponding to the g groups of S; i.e., all ε values for the rows in group 1 are equal to η_1, etc. Because of the grouping property, each condition (2) becomes $\sum_{i=1}^{g} C_i \eta_i \leq \varepsilon$, where C_i is the value defined by the bounded column norm for the group i (recall Definition 3.1). Since the objective function is a minimization, we can make this inequality an equality. Clearly, $\{\eta_i\}_i$ are a feasible solution for (1) – (3).

If R is consistent with G, we can change any optimal solution of (1) – (3) into a solution in which all ε values in a group are equal, without increasing the objective function. We omit a formal proof here. ■

Thus, when S has grouping function G, we can write a simpler optimization problem for noise budgeting:

$$\text{Minimize: } \sum_{i=1}^{g} \frac{\sum_{r:G(r)=i} b_r}{\eta_i^2} \tag{4}$$

$$\text{Subject to: } \sum_{i=1}^{g} C_i \eta_i = \varepsilon. \tag{5}$$

$$\eta_i \geq 0, \quad 1 \leq i \leq m \tag{6}$$

Since there is now just a single constraint on the η_is, we can solve this via a simple Lagrange multiplier method. The corresponding Lagrange function is:

$$\Lambda(\lambda, \eta) = \big(\sum_{i=1}^{g} \frac{\sum_{r:G(r)=i} b_r}{\eta_i^2}\big) + \lambda\big(\sum_{i=1}^{g} C_i \eta_i - \varepsilon\big).$$

Setting the partial derivatives $\frac{\partial}{\partial \eta_i}$ to zero, we obtain $\eta_i = \big(\frac{2}{\lambda C_i} \sum_{r:G(r)=i} b_r\big)^{1/3}$. By the privacy constraint (5), $\sum\limits_{i=1}^{g} \left(\frac{2C_i^2}{\lambda} \sum_{r:G(r)=i} b_r\right)^{1/3} = \varepsilon$ and thus:

$$\lambda = \tfrac{2}{\varepsilon^3}\big(\sum_{i=1}^{g} \big(C_i^2 \sum_{r:G(r)=i} b_r\big)^{1/3}\big)^3$$

Corollary 3.3: In the case when all values C_i are equal to the same value C the optimum value of the objective function is equal to $\frac{C^2}{\epsilon^2}\left(\sum_{i=1}^{g} s_i^{1/3}\right)^3$, where $s_i = \sum_{r:\, G(r)=i} b_r$. For (ϵ, δ)-differential privacy the corresponding value of the objective function is equal to $\frac{2C^2 \log(2/\delta)}{\epsilon^2}\left(\sum_{i=1}^{g} \sqrt{s_i}\right)^2$.

Lemma 3.2 implies the following.

Theorem 3.4: Let S be a strategy matrix with grouping function G, and R be a corresponding recovery matrix consistent with G. Then the solution to the optimization problem (4) – (6) is the optimal noise budgeting for S and R.

As discussed above, when Q is a set of marginals, all the strategy/recovery matrices proposed in prior work fit the conditions of Theorem 3.4, and thus their accuracy can be improved via optimal noise budgeting. Observe that the optimization problem (4) – (6) can be solved reasonably fast: given R, S and a grouping of S, we can derive the vector of b_i values in time linear in the size of R, i.e., in $O(qm)$. For particular S (e.g. the Fourier matrix), the cost can be even lower, due to the symmetric structure of S and R. Finally, all ε_i values, as well as $\mathrm{Var}(y)$, can be computed in $O(m)$ time.

B. Optimal Recovery Matrix (Step 3)

Given $S \in \mathbb{R}^{m \times N}$ and the noise parameters ε_i, we wish to compute a matrix $R \in \mathbb{R}^{q \times m}$ such that $Q = RS$ and $\mathrm{Var}(y)$ is minimized. Recall that $y = Rz = R(Sx + \nu)$, where ν_i is drawn from the Laplace distribution of variance $\frac{2}{\varepsilon_i^2}$, as in Section III-A (the case of Gaussian distribution is similar). As we show below, the resulting y will also be consistent.

We derive R via least squares statistical estimators. More precisely, given $z = Sx + \nu$, we first compute an estimate $\hat{x}$ of x which is linear in z and has minimum variance. The vector $\hat{x}$ is called the optimal (generalized) least squares solution. As we show below, $\hat{x} = Gz$ for some matrix G. We then define $R = QG$ and $y = Rz = Q\hat{x}$. Similar approaches have been used in the past for finding an optimal R in the case of uniform noise. We discuss the case of non-uniform noise budgets ε_i.

Let Σ be the covariance matrix of z: $\Sigma = \mathrm{Cov}(z) = \mathrm{diag}(\frac{2}{\varepsilon_i^2})$. Define $U = \Sigma^{-1/2}S$; hence, $\mathrm{rank}(U) = \mathrm{rank}(S)$. For simplicity, we assume that $\mathrm{rank}(S) = N$. The same ideas as in [16, Section 3.3] can be used to handle the case $\mathrm{rank}(S) < N$; see also [20] for further details. Then the LS solution is computed as

$$\hat{x} = (U^T U)^{-1} U^T \Sigma^{-1/2} z.$$

Since Σ is diagonal, $\Sigma = \Sigma^T$. We obtain

$$\begin{aligned}(U^T U)^{-1} U^T &= (S^T \Sigma^{-1/2} \Sigma^{-1/2} S)^{-1} S^T \Sigma^{-1/2} \\ &= (S^T \Sigma^{-1} S)^{-1} S^T \Sigma^{-1/2}.\end{aligned}$$

$$\text{Thus, } \hat{x} = (S^T \Sigma^{-1} S)^{-1} S^T \Sigma^{-1} z.$$

Let $G = (S^T \Sigma^{-1} S)^{-1} S^T \Sigma^{-1}$. We define $R = QG$, i.e.,

$$R = Q(S^T \Sigma^{-1} S)^{-1} S^T \Sigma^{-1}. \tag{7}$$

Note that $y = Rz = Q\hat{x}$ is consistent, as per Definition 2.3 (with $x^c = \hat{x}$). By a well-known result from linear statistic estimation [20], the following holds:

Lemma 3.5: Matrix R computed as in (7) minimizes $a^T \operatorname{Var}(y)$ (where $y = Rz$). Moreover, y is consistent and unbiased, i.e., the expectation $\mathbb{E}[y] = Qx$.

Observation 1: If S is an orthonormal basis (as with wavelets, Fourier and identity strategies), we have $S^T = S^{-1}$. This implies $G = S^{-1} = S^T$, so $R = QS^T$.

The cost of finding R as above is relatively high, due to the need to perform matrix inversion. While the diagonal matrix Σ is trivial to invert, since $\Sigma_{ii}^{-1} = (\Sigma_{ii})^{-1}$, the matrix $S^T \Sigma^{-1} S$ is generally dense, so is more costly to invert.

C. Fast Consistency

The vector $y = Rz = R(Sx + \nu)$ computed via the optimal recovery matrix R in Step 3 (Section III-B) has two important properties: (i) y is consistent (Definition 2.3); and (ii) $\operatorname{Var}(y_i)$ is minimized for each $1 \leq i \leq q$. Since $\mathbb{E}[y_i] = Q_i.x$ we have $\operatorname{Var}(y_i) = \mathbb{E}[(y_i - Q_i.x)^2]$. Thus, y achieves minimum error for each query in expectation. As observed in [1], [6] for practical applications it may be necessary to return a vector y^1 which is consistent and minimizes a different error measure, e.g., we may wish to minimize $\|y^1 - Qx\|_p$. For example, $p = 1$ implies that y^1 minimizes the average error and $p = \infty$ minimizes maximum error.

In this section we show how to efficiently compute another recovery matrix R^1 such that $y^1 = R^1 z$ is consistent and $\|y^1 - Qx\|_p$ is small. This approach is particularly useful when the query matrix $Q \in \mathbb{R}^{q \times N}$ has $q \ll N$. As we show below, we significantly improve the running times of the approaches used in [1], [6] for this case. The approach in [1], [6], translated in our strategy/recovery framework, is described below.

Start by defining a recovery matrix R^0 such that $Q = R^0 S$ and $y^0 = R^0(Sx + \nu)$ has bounded error $\|y^0 - Qx\|_p \leq t$. Usually, R^0 is the recovery matrix from Step 2 of our framework. For example, the matrix R^0 in [6] is implied by the clustering function over marginals, which heuristically minimizes some L^p-error of the noisy answers. Next, compute a consistent answer y^1 that minimizes $\|y^1 - y^0\|_p$. Recall that y^1 is consistent if there exists x^c such that $y^1 = Qx^c$. Hence, for $p = 1$ or $p = \infty$, y^1 can be computed via a linear program (LP) with variables corresponding to the entries of x^c, and consistency conditions expressed as linear constraints. Other requirements can also be imposed on x^c, e.g., integrality or non-negativity. For $p = 2$, y^1 is the solution to a least squares problem (LS).

However, such an LP, resp. LS, uses at least N variables corresponding to the entries in x^c. When N is large (as is usually the case), this leads to large linear programs. This issue was reported as a bottleneck in the experimental evaluation of [6]. We now propose a different LP, resp. LS, formulation for the consistency problem, which requires at most q variables (recall that q is the number of queries in the workload Q). This leads to large improvements in running time when $q \ll N$.

First, note that $\operatorname{rank}(Q) = q$ implies that any answer $y \in \mathbb{R}^q$ is consistent. This is because the linear system $Qx^c = y$ admits the solution $x^c = Q^T(QQ^T)^{-1}y$ ($\operatorname{rank}(Q) = q$ implies that QQ^T is invertible). In particular, $y^1 = y^0$ is consistent and minimizes $\|y^1 - y^0\|_p$ for any p.

Assume that $\operatorname{rank}(Q) = q' < q$. We pick q' linearly independent rows of Q, denoted as $Q' \in \mathbb{R}^{q' \times N}$, and use them to decompose Q as $Q = CQ'$ for some matrix $C \in \mathbb{R}^{q \times q'}$. Because Q' has linearly independent rows, the above argument implies that, for any $y \in \mathbb{R}^{q'}$, the linear system $Q'x^c = y$ has a solution. Hence, any answer y is consistent for the queries Q'. Then $y^1 = Cy$ is consistent for all queries Q: $y^1 = Cy = CQ'x^c = Qx^c$. We find y that minimizes $\|Cy - y^0\|_p$ and return Cy: For $p = 1$ and $p = \infty$, y is the solution to an LP; for $p = 2$, y is the solution to a least squares problem. In all cases, the number of variables is $q' < q \ll N$. As observed in [1], the utility guarantee follows by the triangle inequality. If $\|Qx - y^0\|_p \leq t$, then

$$\|y^1 - y^0\|_p = \min_{y \in \mathbb{R}^{q'}} \|Cy - y^0\|_p \leq \|CQ'x - y^0\|_p \leq t.$$

Thus, the additional L_p-error introduced by consistency is at most the L_p-error of the original noisy answer, i.e., the error at most doubles.

When Q is a set of marginals, we can formulate the LP, resp. LS, without explicitly computing $\operatorname{rank}(Q)$ or finding a collection of linearly independent rows Q'. Rather, we use the Fourier coefficients of the marginals. The discussion is deferred to Section IV-C.

IV. Consistent Marginals via Fourier Strategies

In this section, we focus on the case when all queries Q correspond to marginals. Here, we show that the choice of S as an appropriate Fourier matrix gives strong guarantees on the variance, as well as providing consistent query answers.

A. Marginals and Fourier analysis

In this section we assume that all d attributes in the database table are binary; for simplicity, let the domain of each attribute be $\{0, 1\}$. We emphasize that this assumption is without loss of generality: an attribute which has $|A|$ distinct values can be mapped to $\lceil \log |A| \rceil$ binary attributes (and we do so in our experimental study). However, we present our results with binary attributes to avoid overcomplicating the notation. Consequently, there are $N = 2^d$ entries in the database vector x, where each entry is indexed by some $\alpha \in \{0, 1\}^d$, and x_α is the number of entries in the database with attributes α; recall the example in Figure 1(a).

There are also 2^d possible marginals (a.k.a. subcubes of the data cube) of interest, corresponding to aggregations along a subset of dimensions. For any $\alpha \in \{0, 1\}^d$, let C^α denote the marginal over non-zero attributes in α, and let $\|\alpha\|$ denote the number of non-zero entries in α, i.e., the dimensionality of the marginal. Note that here α is the bit-vector indicator for the attributes in the marginal. We will consistently use it as a superscript in such cases, and as a subscript when it indexes an entry in a vector.

We use the following notations, as in [1]: For any pair of $\alpha, \beta \in \{0,1\}^d$ we write $\alpha \wedge \beta$ for the bit-wise intersection of the pair, i.e. $(\alpha \wedge \beta)_i = \alpha_i \wedge \beta_i$. The inner-product in this space, $\langle \alpha, \beta \rangle$, can also be expressed via the intersection operator: $\langle \alpha, \beta \rangle = \|\alpha \wedge \beta\|$. We say that α is *dominated by* β, denoted $\alpha \preceq \beta$, if $\alpha \wedge \beta = \alpha$.

The computation of a marginal C^α over the input can be thought of as a linear operator $C^\alpha \colon \mathbb{R}^{2^d} \to \mathbb{R}^{2^{\|\alpha\|}}$ mapping the full-dimensional contingency table to the marginal over non-zero attributes in α, by adding relevant entries over the attributes not in α. More precisely, for each $\beta \preceq \alpha$, the cell β in the marginal C^α, denoted $(C^\alpha x)_\beta$, sums the entries in the contingency table x whose attributes in α are set to values specified by β: $(C^\alpha x)_\beta = \sum_{\gamma\colon \gamma \wedge \alpha = \beta} x_\gamma$.

Example. Let x be the vector in Figure 1(a). Assume we want to compute the marginal $C^\alpha = C^{110}$, i.e., the marginal over attributes A and B. Then the value in the cell $(A = 0, B = 0)$ is denoted by $(C^{110}x)_{000}$ (i.e., $\beta = 000$). The value in the cell $(A = 0, B = 1)$ is denoted by $(C^{110}x)_{010}$. Note that $000 \preceq 110$ and $010 \preceq 110$. On the other hand, $001 \not\preceq 110$, so there is no cell $(C^{110}x)_{001}$ in the marginal over A, B. So, while the cell index β is d-dimensional, only the $\|\alpha\|$ bits corresponding to non-zeros in α vary—the rest are held at 0. Hence, there are only $2^{\|\alpha\|}$ cell indexes in the marginal $C^\alpha x$. In this example, there are only 4 cells in $C^{110}x$. By the above formula, $(C^{110}x)_{000} = x_{000} + x_{001} = 3$ and $(C^{110}x)_{010} = x_{010} + x_{011} = 1$. ■

The set of all marginals C^α with $\|\alpha\| = k$ is referred to as the set of all k-way marginals. They are commonly used to visualize the low-rank dependencies between attributes, to build efficient classifiers from the data, and so on.

We use the Hadamard transform, which is the 2^d-dimensional discrete Fourier transform over the Boolean hypercube $\{0,1\}^d$. This allows us to represent any marginal as a summation over relevant Fourier coefficients. The advantage is that the number of coefficients needed for each marginal is just the number of entries in the marginal. The Fourier basis vectors f^α for $\alpha \in \{0,1\}^d$ have components $f^\alpha_\beta = 2^{-d/2}(-1)^{\langle \alpha, \beta \rangle}$. The vectors f^α form an orthonormal basis in $\mathbb{R}^{2^d}$. We will use the following properties of Fourier basis vectors and marginal operators in the Fourier basis (proofs can be found in [1]):

Theorem 4.1: For all $\alpha, \beta \in \{0,1\}^d$ we have:

1) $(C^\alpha f^\beta)_\gamma = \sum\limits_{\eta\colon \eta \wedge \alpha = \gamma} f^\beta_\eta = \sum\limits_{\eta\colon \eta \wedge \alpha = \gamma} (-1)^{\langle \beta, \eta \rangle} / 2^{d/2}$.

2) $C^\alpha x = \sum\limits_{\beta \preceq \alpha} \langle f^\beta, x \rangle C^\alpha f^\beta$

B. Bounds for marginals

The use of a Fourier strategy matrix was studied in [1], under a uniform error budget. Here, we show that using a non-uniform budgeting can provide asymptotically improved results. We study the case when the query set Q corresponds to a collection of ℓ marginals $C^{\alpha_1}, \ldots, C^{\alpha_\ell}$. For a given marginal C^{α_i} the accuracy bounds will be parametrized by its dimensionality $\|\alpha_i\|$, the total number of marginals ℓ and the total number of Fourier coefficients corresponding to the collection of marginals, denoted as $|\mathcal{F}|$. Theorem 4.1(2) implies that $|\mathcal{F}| = |\cup_i \{\beta : \beta \preceq \alpha_i\}|$. If the random variable corresponding to the differentially private value of a marginal $C^\alpha x$ is denoted as $\tilde{C}^\alpha x$, then we state a bound on the expected absolute error, $\mathbb{E}\left[\|C^\alpha x - \tilde{C}^\alpha x\|_1\right]$ to simplify presentation and comparison with prior work. All our bounds can also be stated in terms of the variance $\mathrm{Var}(\tilde{C}^\alpha x)$, or as high-probability bounds.

The asymptotic bounds on error are easier to interpret in the important special case of the set of all k-way marginals. In this case because of the symmetry of the query workload, the expected error in all marginals is the same. Table I summarizes bounds on error in this case together with the unconditional lower bounds for all differentially private algorithms from [15]. While in the case of (ε, δ)-differential privacy our upper bounds are almost tight with the lower bounds from [15], for ε-differential privacy the gap is still quite significant and remains a challenging open problem.

Our next result gives bounds on expected error of the Fourier strategy with *non-uniform* noise. We omit the proof for lack of space.

Lemma 4.2: For a query workload consisting of all k-way marginals over data $x \in \mathbb{R}^{2^d}$ the bounds on the expected error of the Fourier strategy mechanism with non-uniform noise are given as follows:
1. For ε-differential privacy the expected noise per marginal is $O\big(\frac{1}{\varepsilon} \cdot k\sqrt{\binom{d}{k}\binom{d+k}{k}}\big)$.
2. For (ε, δ)-differential privacy the expected noise per marginal is $O\big(\frac{1}{\varepsilon} \cdot \sqrt{k\binom{d+k}{k}\log(1/\delta)}\big)$.

These bounds are summarized in Table I, along with those that follow from other approaches.[3] Using the Fourier transform as the strategy matrix in our framework improves substantially over prior bounds: strategies based on $S = I$ or $S = Q$ incur factors exponential in d (so linear in N, the size of the contingency table). In contrast, our bounds depend only on k, and factors in d that are polynomial for constant k (the region of most interest). That is, the noise depends on $\log N$.

Time cost comparison. To directly compute a single k-way marginal over d-dimensional data takes time $O(2^d)$, and so computing all k-way marginals takes $O(d^k 2^d)$ naively. Computing the Fourier transform of the data takes time $O(d2^d)$, and deriving the k-way marginals from this takes time $O(4^k)$ per marginal, i.e., $O(d2^d + 4^k d^k)$ for all marginals [1]. We compare the cost of different strategies to these costs. Materializing noisy counts ($S = I$) and aggregating them to obtain the k-way marginals also takes time $O(d^k 2^d)$, as does materializing the marginals and then adding noise ($S = Q$).

[3] We derive a tighter bound for the Fourier stategy under uniform noise than in [1], but details are omitted for space reasons.

Strategy	ε-privacy	(ε, δ)-privacy
Base counts	$O(\frac{1}{\varepsilon}2^{(d+k)/2})$ [9]	$O(\frac{1}{\varepsilon}2^{(d+k)/2}\sqrt{\log(1/\delta)})$ [8]
Marginals	$O(\frac{1}{\varepsilon}2^k\binom{d}{k})$ [1]	$O(\frac{1}{\varepsilon}2^k\sqrt{\binom{d}{k}\log(1/\delta)})$ [1]
Fourier coefficients (uniform noise)	$O(\frac{1}{\varepsilon}k\binom{d}{k}\sqrt{2^k})$	$O(\frac{1}{\varepsilon}\sqrt{k2^k\binom{d}{k}\log(1/\delta)})$ [1]
Fourier coefficients (non-uniform noise)	$O(\frac{1}{\varepsilon}k\sqrt{\binom{d}{k}\binom{d+k}{k}})$ Lemma 4.2	$O(\frac{1}{\varepsilon}\sqrt{k\binom{d+k}{k}\log(1/\delta)})$ Lemma 4.2
Lower bound	$\tilde{\Omega}(\frac{1}{\varepsilon}\sqrt{\binom{d}{k}})$ [15]	$\tilde{\Omega}(\frac{1}{\varepsilon}\sqrt{\binom{d}{k}}(1-\delta/\varepsilon))$ [15]

TABLE I

Releasing all k-way marginals for $k < d/2$. Expected noise per marginal: $\mathbb{E}\left[\|C^\beta x - \tilde{C}^\beta\|_1\right]$. The total number of released marginals is $\binom{d}{k}$ and the total number of Fourier coefficients required to compute these marginals is $\sum_{i=0}^{k}\binom{d}{k} \leq k \cdot \binom{d}{k}$.

The clustering method proposed in [6] is more expensive, due to a search over the space of possible marginals to output. The cost is $O(d^k k \min(2^d d^k, 3^d))$: clearly as the dimensionality grows, this rapidly becomes infeasible. However, across all strategies, the step of choosing the non-uniform error budget is dominated by the other costs, and so does not alter that asymptotic cost.

C. Consistency via Fourier coefficients

In Section III-C we discussed a general approach for computing consistent answers for a query workload $Q \in \mathbb{R}^{q \times N}$ with $\operatorname{rank}(Q) < q$. The approach required explicitly finding a set of $\operatorname{rank}(Q)$ linearly independent rows in Q and decomposing Q. We now show that when Q is a set of marginals we can compute consistent answers without such expensive steps. Instead, we ensure consistency by writing an LP that uses the Fourier coefficients corresponding to the marginals in Q. Let Q consist of ℓ marginals $C^{\alpha_1}, \ldots, C^{\alpha_\ell}$. We introduce variables for the Fourier coefficients corresponding to these marginals, denoted as $\mathcal{F} = \{\hat{f}^\beta | \exists i\colon \beta \preceq \alpha_i\}$. To simplify notation, we rename them as $\mathcal{F} = \{\hat{f}_1, \ldots, \hat{f}_m\}$, where $|\mathcal{F}| = m$. Marginals $C^{\alpha_1}, \ldots, C^{\alpha_\ell}$ can be computed from $\mathcal{F}$, using formulas from Theorem 4.1:

$$(C^{\alpha_i})_\gamma = \textstyle\sum_{\alpha \preceq \alpha_i} \hat{f}^\alpha \left(C^{\alpha_i} f^\alpha\right)_\gamma,$$

for all $i \leq \ell$ and $\gamma \preceq \alpha_i$. We will index entries in the marginals by pairs (i, γ), where $\gamma \preceq \alpha_i$. Let the total number of entries in marginals $C^{\alpha_1}, \ldots, C^{\alpha_k}$ be equal to $\sum_{i=1}^{\ell} 2^{\|\alpha_i\|} = K$. Let R be the recovery matrix for the Fourier strategy: $R \in \mathbb{R}^{K \times m}$ with entries $R_{(i,\gamma),\alpha} = (C^{\alpha_i} f^\alpha)_\gamma$. Then $(C^{\alpha_1}, \ldots, C^{\alpha_\ell}) = R \cdot (\hat{f}_1, \ldots, \hat{f}_m)$. Suppose that we are given a set of inconsistent noisy values of these marginals $(\tilde{C}^{\alpha_1}, \ldots, \tilde{C}^{\alpha_\ell})$. We formulate the following optimization problem to find the consistent set of marginals $(\bar{C}^{\alpha_1}, \ldots, \bar{C}^{\alpha_\ell})$ that is closest to the noisy values in L^p-norm:

$$\begin{gathered}\text{Minimize } \|(\bar{C}^{\alpha_1}, \ldots, \bar{C}^{\alpha_k}) - (\tilde{C}^{\alpha_1}, \ldots, \tilde{C}^{\alpha_\ell})\|_p \\ \text{Subject to } (\bar{C}^{\alpha_1}, \ldots, \bar{C}^{\alpha_\ell}) = R \cdot (\hat{f}_1, \ldots, \hat{f}_m)\end{gathered}$$

For $p = 2$ this is gives a least squares problem, with m variables and K constraints, which is expressed as:

$$\text{Minimize } \|R \cdot (\hat{f}_1, \ldots, \hat{f}_m) - (\tilde{C}^{\alpha_1}, \ldots, \tilde{C}^{\alpha_\ell})\|_2.$$

For $p = 1$ and $p = \infty$, this gives an LP similar to [1].

The running time of this consistency step via least squares only depends on the number of queries. For example, for the case of all k-way marginals, we need to work with matrices of size $O(d^k)$, and perform a constant number of multiplications and inversions. In contrast, prior work required solving LPs of size proportional to the size of the data, $N = 2^d$, which takes time polynomial in N.

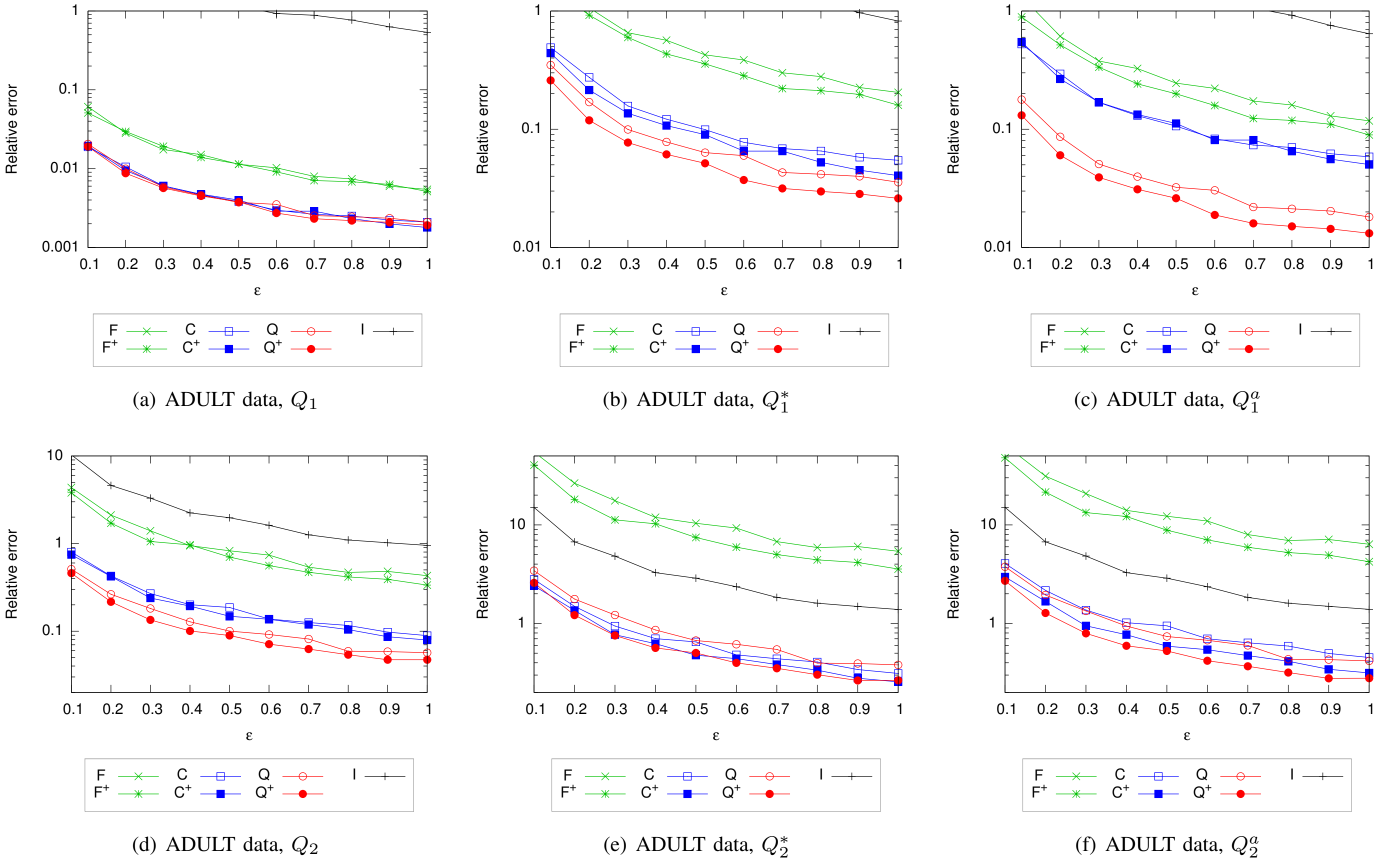
V. Experimental study

We performed an experimental study to compare the efficiency and accuracy of the different methods on real data sets, applied to the problem of releasing marginals.

Datasets. We studied performance on two real datasets:
Adult: The Adult dataset from the UCI Machine Learning repository (`http://archive.ics.uci.edu/ml/`) has census information on 32561 individuals. As in [6], we extract a subset of sensitive categorical attributes, for workclass (cardinality 9), education (16), marital-status (7), occupation (15), relationship (6), race (5), sex (2) and salary (2).
NLTCS: The National Long-Term Case Study from StatLib (`http://lib.stat.cmu.edu/`), contains information about 21576 individuals. Each record consists of 16 binary attributes, which correspond to functional disability measures: 6 activities of daily living and 10 instrumental activities of daily living.

Query workloads. The choice of query workloads in our experimental study is motivated by an application of low-order marginals to statistical model fitting. In this setting, the typical set of queries consists of all k-way marginals (for some small value of k) together with some subset of $(k+1)$-way marginals, chosen depending on the application. We consider three different approaches:
1. Q_k: all the k-way marginal tables.
2. Q_k^*: all the k-way marginal tables, plus half of all $(k+1)$-way marginals.
3. Q_k^a: all the k-way marginal tables, plus all $(k+1)$-way marginals that include a fixed attribute.

Evaluation metrics. We measure the average absolute error per entry in the set of marginal queries. To show the utility of these results, we scale each error by the mean true answer of its respective marginal query, i.e., we plot it as a relative error.



(a) ADULT data, Q_1

(b) ADULT data, Q_1^*

(c) ADULT data, Q_1^a

(d) ADULT data, Q_2

(e) ADULT data, Q_2^*

(f) ADULT data, Q_2^a

Fig. 4. Accuracy of marginal release on ADULT data

Thus, a relative error of less than 1 is desirable, as otherwise the true answers are dwarfed by the noise (on average). Note that while the number of tuples in each dataset is relatively small, our approaches do not depend on the tuple count, but rather the dimensionality of the domain $N = \Pi_{i=1}^{d} |A_i|$. Larger datasets would only improve the quality metrics, while keeping the running time essentially unchanged.

Algorithms Used. We present results for ϵ-differential privacy. Results for (ϵ, δ)-differential privacy are similar, and are omitted. We include seven approaches within the strategy/recovery framework, based on choice of the strategy matrix, S. Here, the notation S^+ indicates that we use the non-uniform noise allocation for strategy S as described in Section III-A, while the corresponding S is with uniform noise.

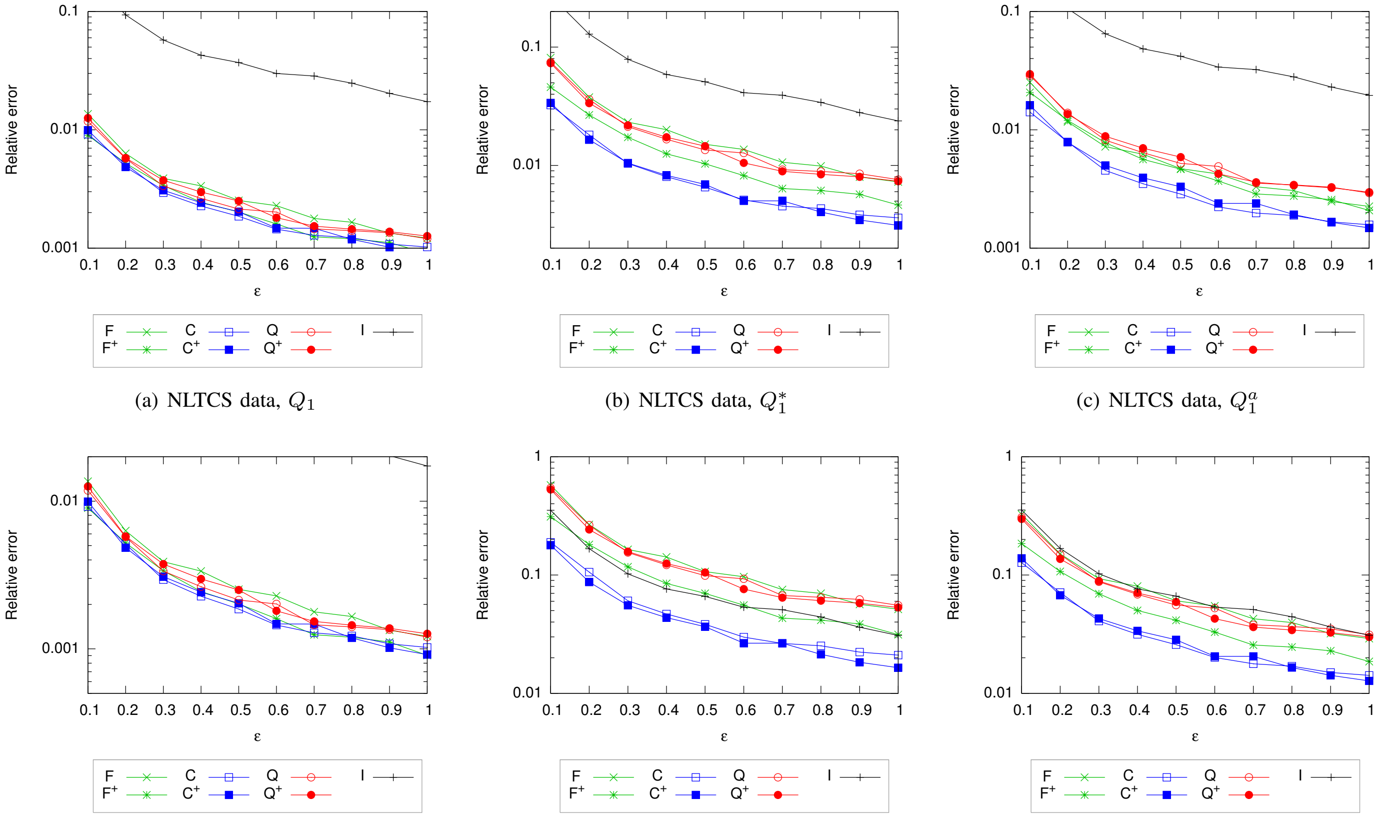
- $S = I$ — Add noise via Laplace mechanism directly to base cells and aggregate up to compute the marginals. Here, the optimal noise allocation is always uniform.
- $S = Q$ — Add uniform (Q) or non-uniform (Q^+) noise to each marginal independently.
- $S = F$ — Add uniform (F) or non-uniform (F^+) noise to each Fourier coefficient, corresponding to the given query workload.
- $S = C$ — Add uniform (C) or non-uniform (C^+) noise to each marginal returned by the greedy clustering strategy proposed in [6].

Our goal is to study the effects of non-uniform noise budgeting over all strategies. The decision on which strategy to use rests with the data owner. However, we show clear tradeoffs between running time and accuracy for all strategies, which can provide helpful hints.

To ensure consistency of the released marginals, we use the Fourier analytic approach, described in Section IV-C.

A. Adult Dataset

Figure 4 shows the results on the Adult data set, for query workloads $Q_1, Q_1^*, Q_1^a, Q_2, Q_2^*$ and Q_2^a. The attributes in this data set have varying cardinalities, but are encoded as binary attributes as described in Section IV-A. We plot the results on a logarithmic scale as we vary the privacy parameter ϵ, to more clearly show the relative performance of the different measures. Immediately, we can make several observations about the relative performance of the different methods. On this data, the naive method of materializing counts (I) is never effective: the noise added is comparable to the magnitude of the data in all cases. Across the different query workloads, choosing the strategy $S = Q$ works generally well. In this case non-uniform noise allocation can significantly improve the accuracy. For example, over workload Q_1^* (Figure 4(b)), we see an improvement of 20-25% in accuracy.



(a) NLTCS data, Q_1

(b) NLTCS data, Q_1^*

(c) NLTCS data, Q_1^a

(d) NLTCS data, Q_2

(e) NLTCS data, Q_2^*

(f) NLTCS data, Q_2^a

Fig. 5. Accuracy of marginal release on NLTCS data

For more complex queries which result in more marginals of higher degree (Q_2^* and Q_2^a, in Figures 4(e) and 4(f) respectively), the accuracy is lower overall, and the noise is greater than the magnitude of the data for more restrictive settings of the privacy parameter ϵ. To some extent this can be mitigated as the number of individuals represented in the data increases: the noise stays constant as the value of the counts in the table grows.

Across this data, we observe that while the non-uniform approach improves the accuracy of the Fourier strategy, it is inferior to other strategies. Although asymptotically this strategy has good properties (as described in Table I), in this case k is not very large, so the gap between the k and 2^k terms for constant k is absorbed within the big-Oh notation. The running times of our methods were all fast: the Fourier (F) and Query (Q) methods took at most tens of seconds to complete, while the clustering (C) took longer, due to the more expensive clustering step.

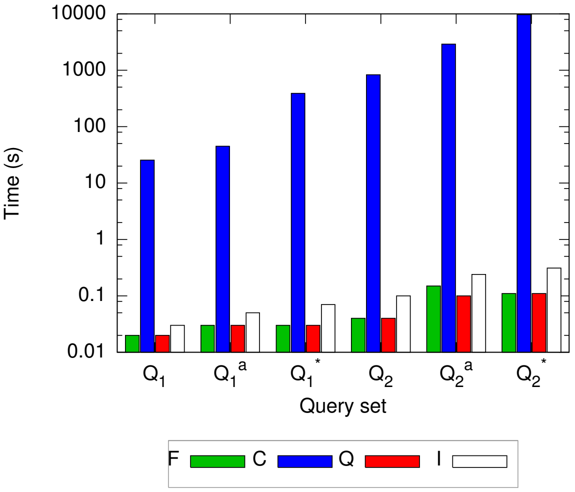


Fig. 6. Running time over NLTCS data

B. NLTCS data

Figure 5 shows the corresponding results on the binary NLTCS data. Over all experiments, there is an appreciable benefit to applying the optimal non-uniform budgeting. The optimal budgeting case is reliably better than the uniform version, for the same strategy matrix. There are occasional inversions, due to the random nature of the mechanisms used, but the trend is clear. The advantage can be notable: for example, on Q_1^* (Figure 5(b)) and Q_2^* (Figure 5(e)), the error of the Fourier strategy is reduced 30-35% by using non-uniform budgeting. For the clustering approach, the improvement is smaller, but still measurable, around 5% on average. However, recall that strategy C becomes infeasible on higher dimensional data, due to its exponential cost.

Figure 6 shows the end-to-end running time of the different methods. This demonstrates clearly the dramatically slow running time of the clustering method: reaching several hours to operate on a single, moderate-sized dataset. As the dimensionality increases, this becomes exponentially worse. By contrast, the time needed by the other strategies is negligible: always less than a second, and typically less than a tenth of a second. The optimization and consistency steps take essentially no time at all, compared to the data handling and processing. On the other hand, all methods have virtually constant time as a function of the number of tuples in the data.

Returning to the accuracy, over the $k = 1$-way marginals and variations, the approach of materializing the base counts (I) is not competitive, while the clustering strategy (C) achieves the least error. The more lightweight Fourier-based approach achieves slightly more error, but is much more scalable. As the degree of the marginals increases (Q_2^*, Figure 5(e), and Q_2^a, Figure 5(f)), the trivial solution of materializing the base cells becomes more accurate. For workloads that are made up of high-degree marginals, this method dominates the other approaches, although such workloads are considered less realistic.

VI. Concluding Remarks

We considered the problem of releasing data based on linear queries, which captures the common case of data cubes and marginals. We showed how existing matrix-based strategies can be improved by using non-uniform noise based on the query workload. Our results show that such non-uniform noise results in significantly lower error across all cases considered. Further, the cost of this is low, and the results can be made consistent with minimal extra effort.

Other notions of consistency are possible within this framework. For example, it is sometimes required that the query answers correspond to a data set in which all counts are integral and non-negative. This can be achieved when the method actually materializes a noisy set of base counts $\hat{x}$ (as in the case of strategy I) by adding the constraints that $\hat{x}_j \geq 0$ and rounding the results to the nearest integer. It remains to show how to enforce such consistency constraints efficiently when base counts are not explicitly materialized.

On the theoretical side, we have shown bounds on accurate k-way release under differential privacy of $O(\frac{k}{\varepsilon}\sqrt{\binom{d}{k}\binom{d+k}{k}})$. An open problem is to close the gap between this and the lower bound of $\tilde{\Omega}(\frac{1}{\varepsilon}\sqrt{\binom{d}{k}})$.

Acknowledgments. We thank Adam D. Smith and Moritz Hardt for multiple useful comments and suggestions.

References

[1] 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*, PODS '07, pages 273–282, New York, NY, USA, 2007. ACM.

[2] 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*, STOC '08, pages 609–618, New York, NY, USA, 2008. ACM.

[3] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[4] G. Cormode, C. Procopiuc, E. Shen, D. Srivastava, and T. Yu. Differentially private spatial decompositions. In *International Conference on Data Engineering (ICDE)*, 2012.

[5] G. Cormode, C. Procopiuc, D. Srivastava, and T. Tran. Differentially private publication of sparse data. In *International Conference on Database Theory (ICDT)*, 2012.

[6] B. Ding, M. Winslett, J. Han, and Z. Li. Differentially private data cubes: optimizing noise sources and consistency. In *Proceedings of the 2011 international conference on Management of data*, SIGMOD '11, pages 217–228, New York, NY, USA, 2011. ACM.

[7] C. Dwork. Differential privacy. In *ICALP*, pages 1–12, 2006.

[8] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor. Our data, ourselves: Privacy via distributed noise generation. In S. Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 486–503. Springer Berlin / Heidelberg, 2006.

[9] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In S. Halevi and T. Rabin, editors, *Theory of Cryptography*, volume 3876 of *Lecture Notes in Computer Science*, pages 265–284. Springer Berlin / Heidelberg, 2006.

[10] A. Gupta, A. Roth, and J. Ullman. Iterative constructions and private data release. *CoRR*, abs/1107.3731, 2011.

[11] M. Hardt, K. Ligett, and F. McSherry. A simple and practical algorithm for differentially private data release. *CoRR*, abs/1012.4763, 2010.

[12] M. Hardt and G. N. Rothblum. A multiplicative weights mechanism for privacy-preserving data analysis. In *Proceedings of the 2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, FOCS '10, pages 61–70, Washington, DC, USA, 2010. IEEE Computer Society.

[13] M. Hardt and K. Talwar. On the geometry of differential privacy. In *Proceedings of the 42nd ACM symposium on Theory of computing*, STOC '10, pages 705–714, New York, NY, USA, 2010. ACM.

[14] M. Hay, V. Rastogi, G. Miklau, and D. Suciu. Boosting the accuracy of differentially-private histograms through consistency. In *VLDB*, 2010.

[15] S. P. Kasiviswanathan, M. Rudelson, A. Smith, and J. Ullman. The price of privately releasing contingency tables and the spectra of random matrices with correlated rows. In *Proceedings of the 42nd ACM symposium on Theory of computing*, STOC '10, pages 775–784, New York, NY, USA, 2010. ACM.

[16] 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*, PODS '10, pages 123–134, New York, NY, USA, 2010. ACM.

[17] C. Li and G. Miklau. An adaptive mechanism for accurate query answering under differential privacy. In *PVLDB*, 2012.

[18] Y. Li, Z. Zhang, M. Winslett, and Y. Yang. Compressive mechanism: Utilizing sparse representation in differential privacy. In *Workshop on Privacy in the Electronic Society*, 2011.

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

[20] C. R. Rao. *Linear statistical inference and its applications*. Wiley, 1965.

[21] A. Roth and T. Roughgarden. Interactive privacy via the median mechanism. In *Proceedings of the 42nd ACM symposium on Theory of computing*, STOC '10, pages 765–774, New York, NY, USA, 2010. ACM.

[22] X. Xiao, G. Wang, and J. Gehrke. Differential privacy via wavelet transforms. In *ICDE*, 2010.

[23] G. Yuan, Z. Zhang, M. Winslett, X. Xiao, Y. Yang, and Z. Hao. Low-rank mechanism: optimizing batch queries under differential privacy. *Proceedings of the VLDB Endowment*, 5(11):1352–1363, 2012.