Optimizing Linear Counting Queries Under Differential Privacy

Chao Li[†], Michael Hay[†], Vibhor Rastogi[‡], Gerome Miklau[†], Andrew McGregor[†]

[†]University of Massachusetts Amherst,
Amherst, Massachusetts, USA
{chaoli,mhay,miklau,mcgregor}@cs.umass.edu

[‡]University of Washington,
Seattle, Washington, USA
vibhor@cs.washington.edu

ABSTRACT

Differential privacy is a robust privacy standard that has been successfully applied to a range of data analysis tasks. But despite much recent work, optimal strategies for answering a collection of related queries are not known.

We propose the matrix mechanism, a new algorithm for answering a workload of predicate counting queries. Given a workload, the mechanism requests answers to a different set of queries, called a query strategy, which are answered using the standard Laplace mechanism. Noisy answers to the workload queries are then derived from the noisy answers to the strategy queries. This two stage process can result in a more complex correlated noise distribution that preserves differential privacy but increases accuracy.

We provide a formal analysis of the error of query answers produced by the mechanism and investigate the problem of computing the optimal query strategy in support of a given workload. We show this problem can be formulated as a rank-constrained semidefinite program. Finally, we analyze two seemingly distinct techniques, whose similar behavior is explained by viewing them as instances of the matrix mechanism.

Categories and Subject Descriptors: H.2.8 [**Database Management**]: Database Applications—*Statistical databases*; G.1 [**Numerical Analysis**]: Optimization

General Terms: Algorithms, Security, Theory

Keywords: private data analysis, output perturbation, differential privacy, semidefinite program.

1. INTRODUCTION

Differential privacy [8] offers participants in a dataset the compelling assurance that information released about the dataset is virtually indistinguishable whether or not their personal data is included. It protects against powerful adversaries and offers precise accuracy guarantees. As outlined in recent surveys [5, 6, 7], it has been applied successfully to a range of data analysis tasks and to the release of summary statistics such as contingency tables [1], histograms [11, 17], and order statistics [13].

Differential privacy is achieved by introducing randomness into query answers. The original algorithm for achieving differential privacy, commonly called the Laplace mechanism [8], returns the sum of the true answer and random noise drawn from a Laplace distribution. The scale of the distribution is determined by a property of the query called its sensitivity: roughly the maximum possible change to the query answer induced by the addition or removal of one tuple. Higher sensitivity queries are more revealing about individual tuples and must receive greater noise.

If an analyst requires only the answer to a single query about the database, then the Laplace mechanism has recently been shown optimal in a strong sense [9]. But when multiple query answers are desired, an optimal mechanism is not known.

At the heart of our investigation is the suboptimal behavior of the Laplace mechanism when answers to a set of correlated queries are requested. We say two queries are correlated if the change of a tuple in the underlying database can affect both answers. Asking correlated queries can lead to suboptimal results because correlation increases sensitivity and therefore the magnitude of the noise. The most extreme example is when two duplicate queries are submitted. The sensitivity of the pair of queries is twice that of an individual query. This means the magnitude of the noise added to each query is doubled, but combining the two noisy answers (in the natural way, by averaging) gives a less accurate result than if only one query had been asked.

Correlated workloads arise naturally in practice. If multiple users are interacting with a database, the server may require that they share a common privacy budget to avoid the threat of a privacy breach from collusion. Yet, in acting independently, they can easily issue redundant or correlated queries. Further, in some settings it is appealing to simultaneously answer a large structured set of queries, (e.g. all range queries), which are inherently correlated.

In this work we propose the *matrix mechanism*, an improved mechanism for answering a workload of predicate counting queries. Each query is a linear combination of base counts reporting the number of tuples with the given combination of attribute values. A set of such queries is represented as a matrix in which each row contains the coefficients of a linear query. Histograms, sets of marginals, and data cubes can be viewed as workloads of linear counting queries.

The matrix mechanism is built on top of the Laplace mech-

PODS'10, June 6–11, 2010, Indianapolis, Indiana, USA.

anism. Given a workload of queries, the matrix mechanism asks a different set of queries, called a *query strategy*, and obtains noisy answers by invoking the Laplace mechanism. Noisy answers to the workload queries are then derived from the noisy answers to the strategy queries. There may be more than one way to estimate a workload query from the answers to the strategy queries. In this case the derived answer of the matrix mechanism combines the available evidence into a single consistent estimate that minimizes the variance of the noisy answer.

While the Laplace mechanism always adds independent noise to each query in the workload, the noise of the matrix mechanism may consist of a complex linear combination of independent noise samples. Such correlated noise preserves differential privacy but can allow more accurate results, particularly for workloads with correlated queries.

The accuracy of the matrix mechanism depends on the query strategy chosen to instantiate it. This paper explores the problem of designing the optimal strategy for a given workload. To understand the optimization problem we first analyze the error of any query supported by a strategy. The error is determined by two essential features of the strategy: its *error profile*, a matrix which governs the distribution of error across queries, and its *sensitivity*, a scalar term that uniformly scales the error on all queries. Accurately answering a workload of queries requires choosing a strategy with a good error profile (relatively low error for the queries in the workload) and low sensitivity. We show that natural strategies succeed at one, but not both, of these objectives.

We then formalize the optimization problem of finding the strategy that minimizes the total error on a workload of queries as a semi-definite program with rank constraints. Such problems can be solved with iterative algorithms, but we are not aware of results that bound the number of iterations until convergence. In addition, we propose two efficient approximations for deciding on a strategy, as well as a heuristic that can be used to improve an existing strategy.

Lastly, our framework encompasses several techniques proposed in the literature. We use it to analyze two techniques [11, 17], each of which can be seen as an instance of the matrix mechanism designed to support the workload consisting of all range queries. Our analysis provides insight into the common behavior of these seemingly distinct techniques, and we prove novel bounds on their error.

After a background discussion we describe the matrix mechanism in Section 3. We analyze its error formally in Section 4. In Section 5, we characterize the optimization problem of choosing a query strategy and propose approximations. We use our results to compare existing strategies in Section 6. We discuss related work, including other recent techniques that improve on the Laplace mechanism, in Section 7.

2. BACKGROUND

This section describes the domain and queries considered, and reviews the basic principles of differential privacy. We use standard terminology of linear algebra throughout the paper. Matrices and vectors are indicated with bold letters (e.g $\mathbf{A}$ or $\mathbf{x}$) and their elements are indicated as a_{ij} or x_i. For a matrix $\mathbf{A}$, $\mathbf{A}^t$ is its transpose, $\mathbf{A}^{-1}$ is its inverse, and trace($\mathbf{A}$) is its trace (the sum of values on the main diagonal). We use $diag(c_1, \dots c_n)$ to indicate an $n \times n$ diagonal matrix with scalars c_i on the diagonal. We use $\mathbf{0}^{m \times n}$ to indicate a matrix of zeroes with m rows and n columns.

$$\underset{\mathbf{I}_4}{\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}} \quad \underset{\mathbf{H}_4}{\begin{bmatrix}1&1&1&1\\1&1&0&0\\0&0&1&1\\1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}} \quad \underset{\mathbf{Y}_4}{\begin{bmatrix}1&1&1&1\\1&1&-1&-1\\1&-1&0&0\\0&0&1&-1\end{bmatrix}}$$

Figure 1: Query matrices with $dom = \{1,2,3,4\}$. Each is full rank. I_4 returns each unit count. H_4 computes seven sums, hierarchically partitioning the domain. W_4 is based on the Haar wavelet.

2.1 Linear queries

The database is an instance I of relational schema $R(\mathbb{A})$, where $\mathbb{A}$ is a set of attributes. We denote by $dom(\mathbb{A})$ the cross-product of the domains of attributes in $\mathbb{A}$. The analyst chooses a set of attributes $\mathbb{B} \subseteq \mathbb{A}$ relevant to their task. For example if the analyst is interested in a subset of two dimensional range queries over attributes A_1 and A_2, they would set $\mathbb{B} = \{A_1, A_2\}$. We then form a frequency vector $\mathbf{x}$ with one entry for each element of $dom(\mathbb{B})$. For simplicity we assume $dom(\mathbb{B}) = \{1, 2, \dots, n\}$ and for each $i \in dom(\mathbb{B})$, x_i is the count of tuples equal to i in the projection $\Pi_{\mathbb{B}}(I)$. We represent $\mathbf{x}$ as a column vector of counts: $\mathbf{x} = [x_1 \dots x_n]^t$.

A *linear query* computes a linear combination of the counts in $\mathbf{x}$.

Definition 2.1 (Linear query). *A* linear query *is a length-n row vector* $\mathbf{q} = [q_1 \dots q_n]$ *with each* $q_i \in \mathbb{R}$. *The answer to a linear query* $\mathbf{q}$ *on* $\mathbf{x}$ *is the vector product* $\mathbf{qx} = q_1x_1 + \cdots + q_nx_n$.

We will consider sets of linear queries organized into the rows of a *query matrix*.

Definition 2.2 (Query matrix). *A* query matrix *is a collection of* m *linear queries, arranged by rows to form an* $m \times n$ *matrix.*

If $\mathbf{Q}$ is an $m \times n$ query matrix, the query answer for $\mathbf{Q}$ is a length m column vector of query results, which can be computed as the matrix product $\mathbf{Qx}$.

Example 1. *Figure 1 shows three query matrices, which we use as running examples throughout the paper.* $\mathbf{I}_4$ *is the identity matrix of size four. This matrix consists of four queries, each asking for an individual element of* $\mathbf{x}$. $\mathbf{H}_4$ *contains seven queries, which represent a binary hierarchy of sums: the first row is the sum over the entire domain (returning the total number of tuples in* $\mathbf{I}$*), the second and third rows each sum one half of the domain, and the last four rows return individual elements of* $\mathbf{x}$. $\mathbf{Y}_4$ *is the matrix of the Haar wavelet. It can also be seen as a hierarchical set of queries: the first row is the total sum, the second row computes the difference between sums in two halves of the domain, and the last two rows return differences between smaller partitions of the domain. In Section 6 we study general forms of these matrices for domains of size n [11, 17].*

2.2 The Laplace mechanism

Because the true counts in $\mathbf{x}$ must be protected, only noisy answers to queries, satisfying differential privacy, are released. We refer to the noisy answer to a query as an *estimate* for the true query answer. The majority of our results concern classical ϵ-differential privacy, reviewed below. (We consider a relaxation of differential privacy briefly in Sec. 5.2.)

Informally, a randomized algorithm is differentially private if it produces statistically close outputs whether or not any one individual's record is present in the database. For any input database I, let $nbrs(I)$ denote the set of neighboring databases, each differing from I by at most one record; i.e., if $I' \in nbrs(I)$, then $|(I - I') \cup (I' - I)| = 1$.

Definition 2.3 (ϵ-differential privacy). *A randomized algorithm $\mathcal{K}$ is ϵ-differentially private if for any instance I, any $I' \in nbrs(I)$, and any subset of outputs $S \subseteq Range(\mathcal{K})$, the following holds:*

$$Pr[\mathcal{K}(I) \in S] \leq \exp(\epsilon) \times Pr[\mathcal{K}(I') \in S],$$

where the probability is taken over the randomness of the $\mathcal{K}$.

Differential privacy can be achieved by adding random noise to query answers. The noise added is a function of the privacy parameter, ϵ, and a property of the queries called *sensitivity.* The sensitivity of a query bounds the possible change in the query answer over any two neighboring databases. For a single linear query, the sensitivity bounds the absolute difference of the query answers. For a query matrix, which returns a vector of answers, the sensitivity bounds the L_1 distance between the answer vectors resulting from any two neighboring databases. The following proposition extends the standard notion of query sensitivity to query matrices. Note that because two neighboring databases I and I' differ in exactly one tuple, it follows that their corresponding vectors $\mathbf{x}$ and $\mathbf{x}'$ differ in exactly one component, by exactly one.

Proposition 1 (Query matrix sensitivity). *The sensitivity of matrix $\mathbf{Q}$, denoted $\Delta_{\mathbf{Q}}$, is:*

$$\Delta_{\mathbf{Q}} \quad =^{def} \quad \max_{\|\mathbf{x}-\mathbf{x}'\|_1=1} \left\|\mathbf{Q}\mathbf{x} - \mathbf{Q}\mathbf{x}'\right\|_1 = \max_j \sum_{i=1}^{n} |q_{ij}|.$$

Thus the sensitivity of a query matrix is the maximum L_1 norm of a column.

Example 2. *The sensitivities of the query matrices in Figure 1 are: $\Delta_{\mathbf{I}_4} = 1$ and $\Delta_{\mathbf{H}_4} = \Delta_{\mathbf{Y}_4} = 3$. A change by one in any component $\mathbf{x}_i$ will change the query answer $\mathbf{I}_4\mathbf{x}$ by exactly one, but will change $\mathbf{H}_4\mathbf{x}$ and $\mathbf{Y}_4\mathbf{x}$ by three since each x_i contributes to three linear queries in both $\mathbf{H}_4$ and $\mathbf{Y}_4$.*

The following proposition describes an ϵ-differentially private algorithm, adapted from Dwork et al. [5], for releasing noisy answers to the workload of queries in matrix $\mathbf{W}$. The algorithm adds independent random samples from a scaled Laplace distribution.

Proposition 2 (Laplace mechanism). *Let $\mathbf{W}$ be a query matrix consisting of m queries, and let $\tilde{\mathbf{b}}$ be a length-m column vector consisting of independent samples from a Laplace distribution with scale 1. Then the randomized algorithm $\mathcal{L}$ that outputs the following vector is ϵ-differentially private:*

$$\mathcal{L}(\mathbf{W}, \mathbf{x}) = \mathbf{W}\mathbf{x} + (\frac{\Delta_{\mathbf{W}}}{\epsilon})\tilde{\mathbf{b}}.$$

Recall that $\mathbf{W}\mathbf{x}$ is a length-m column vector representing the true answer to each linear query in $\mathbf{W}$. The algorithm adds independent random noise, scaled by ϵ and the sensitivity of $\mathbf{W}$. Thus $\mathcal{L}(\mathbf{W}, \mathbf{x})$, which we call the *output vector*, is a length-m column vector containing a noisy answer for each linear query in $\mathbf{W}$.

3. THE MATRIX MECHANISM

Central to our approach is the distinction between a query *strategy* and a query *workload.* Both are sets of linear queries represented as matrices. The workload queries are those queries for which the analyst requires answers. Submitting the workload queries to the Laplace mechanism described above is the standard approach, but may lead to greater error than necessary in query estimates. Instead we submit a different set of queries to the differentially private server, called the query strategy. We then use the estimates to the strategy queries to derive estimates to the workload queries. Because there may be more than one derived estimate for a workload query, we wish to find a single consistent estimate with least error.

In this section we present the formal basis for this derivation process. We define the set of queries whose estimates can be derived and we provide optimal mechanisms for deriving estimates. Using this derivation, we define the *matrix mechanism*, an extension of the Laplace mechanism that uses a query strategy $\mathbf{A}$ to answer a workload $\mathbf{W}$ of queries. The remainder of the paper will then investigate, given $\mathbf{W}$, how to choose the strategy $\mathbf{A}$ to instantiate the mechanism.

3.1 Deriving new query answers

Suppose we use the Laplace mechanism to get noisy answers to a query strategy $\mathbf{A}$. Then there is sufficient evidence, in the noisy answers to $\mathbf{A}$, to construct an estimate for a workload query $\mathbf{w}$ if $\mathbf{w}$ can be expressed as a linear combination of the strategy queries:

Definition 3.1 (Queries supported by a strategy). *A strategy $\mathbf{A}$ supports a query $\mathbf{w}$ if $\mathbf{w}$ can be expressed as a linear combination of the rows of $\mathbf{A}$.*

In other words, $\mathbf{A}$ supports any query $\mathbf{w}$ that is in the subspace defined by the rows of $\mathbf{A}$. If a strategy matrix consists of at least n linearly independent row vectors (i.e., its row space has dimension n), then it follows immediately that it supports all linear queries. Such matrices are said to have *full rank.* We restrict our attention to full rank strategies and defer discussion of this choice to the end of the section.

To derive new query answers from the answers to $\mathbf{A}$ we first compute an estimate, denoted $\hat{\mathbf{x}}_{\mathbf{A}}$, of the true counts $\mathbf{x}$. Then the derived estimate for an arbitrary linear query $\mathbf{w}$ is simply the vector product $\mathbf{w}\hat{\mathbf{x}}_{\mathbf{A}}$. The estimate of the true counts is computed as follows:

Definition 3.2 (Estimate of $\mathbf{x}$ using $\mathbf{A}$). *Let $\mathbf{A}$ be a full rank query strategy $\mathbf{A}$ consisting of m queries, and let $\mathbf{y} = \mathcal{L}(\mathbf{A}, \mathbf{x})$ be the noisy answers to $\mathbf{A}$. Then $\hat{\mathbf{x}}_{\mathbf{A}}$ is the estimate for $\mathbf{x}$ defined as:*

$$\hat{\mathbf{x}}_{\mathbf{A}} = \mathbf{A}^{+}\mathbf{y},$$

where $\mathbf{A}^{+} = (\mathbf{A}^t\mathbf{A})^{-1}\mathbf{A}^t$ is the pseudo-inverse *of $\mathbf{A}$.*

Because $\mathbf{A}$ has full rank, the number of queries in $\mathbf{A}$, m, must be at least n. When $m = n$, then $\mathbf{A}$ is invertible and

$\mathbf{A}^+ = \mathbf{A}^{-1}$. Otherwise, when $m > n$, $\mathbf{A}$ is not invertible, but $\mathbf{A}^+$ acts as a left-inverse for $\mathbf{A}$ because $\mathbf{A}^+\mathbf{A} = \mathbf{I}$. We explain next the justification for the estimate $\hat{\mathbf{x}}_\mathbf{A}$ above, and provide examples, considering separately the case where $m = n$ and the case where $m > n$.

A is square. In this case $\mathbf{A}$ is an $n \times n$ matrix of rank n, and it is therefore invertible. Then given the output vector $\mathbf{y}$, it is always possible to compute a unique estimate for the true counts by inverting $\mathbf{A}$. The expression in Definition 3.2 then simplifies to :

$$\hat{\mathbf{x}}_\mathbf{A} = \mathbf{A}^{-1}\mathbf{y}.$$

In this case, query strategy $\mathbf{A}$ can be viewed as a linear transformation of the true counts, to which noise is added by the privacy mechanism. The transformation is then reversed, by the inverse of $\mathbf{A}$, to produce a consistent estimate of the true counts.

Example 3. *In Figure 1,* $\mathbf{I}_4$ *and* $\mathbf{Y}_4$ *are both square, full rank matrices which we will use as example query strategies. The inverse of* $\mathbf{I}_4$ *is just* $\mathbf{I}_4$ *itself, reflecting the fact that since* $\mathbf{I}_4$ *asks for individual counts of* $\mathbf{x}$*, the estimate* $\hat{\mathbf{x}}$ *is just the output vector* $\mathbf{y}$*. The inverse of* $\mathbf{Y}_4$ *is shown in Figure 2(c). Row* i *contains the coefficients used to construct an estimate of count* x_i*. For example, the first component of* $\hat{\mathbf{x}}_\mathbf{A}$ *will be computed as the following weighted sum:* $.25y_1 + .25y_2 + .5y_3$.

Specific transformations of this kind have been studied before. A Fourier transformation is used in [1], however, rather than recover the entire set of counts, the emphasis is on a set of marginals. A transformation using the Haar wavelet is considered [17]. Our insight is that any full rank matrix is a viable strategy, and our goal is to understand the properties of matrices that make them good strategies. In Section 6 we analyze the wavelet technique [17] in detail.

A is rectangular. When $m > n$, we cannot invert $\mathbf{A}$ and we must employ a different technique for deriving estimates for the counts in $\mathbf{x}$. In this case, the matrix $\mathbf{A}$ contains n linearly independent rows, but has additional row queries as well. These are additional noisy observations that should be integrated into our estimate $\hat{\mathbf{x}}_\mathbf{A}$. Viewed another way, we have a system of equations given by $\mathbf{y} = \mathbf{A}\mathbf{x}$, with more equations ($m$) than the number of unknowns in $\mathbf{x}$ (n). The system of equations is likely to be inconsistent due to the addition of random noise.

We adapt techniques of linear regression, computing an estimate $\hat{\mathbf{x}}_\mathbf{A}$ that minimizes the sum of the squared deviations from the output vector. Because we assume $\mathbf{A}$ has full rank, this estimate, called the *least squares* solution, is unique. The expression in Definition 3.2 computes the well-known least squares solution as $\hat{\mathbf{x}} = (\mathbf{A}^t\mathbf{A})^{-1}\mathbf{A}^t\mathbf{y}$.

This least squares approach was originally proposed in [11] as a method for avoiding inconsistent answers in differentially private outputs, and it was shown to improve the accuracy of a set of histogram queries. In that work, a specific query strategy is considered (related to our example $\mathbf{H}_4$) consisting of a hierarchical set of queries. An efficient algorithm is proposed for computing the least squares solution in this special case. We analyze this strategy further in Sec. 6.

Example 4. $\mathbf{H}_4$*, shown in Figure 1, is a rectangular full rank matrix with* $m = 7$*. The output vector* $\mathbf{y} = \mathcal{L}(\mathbf{H}_4, \mathbf{x})$ *does not necessarily imply a unique estimate. For example, each of the following are possible estimates of* x_1*:* y_4*,* $y_2 - y_5$*,* $y_1 - y_3 - y_5$*, each likely to result in different answers. The reconstruction matrix for* $\mathbf{H}_4$*,* $\mathbf{H}_4^+$ *shown in Fig 2(b), describes the unique least squares solution. The estimate for* x_1 *is a weighted combination of values in the output vector:* $\frac{3}{21}y_1 + \frac{5}{21}y_2 - \frac{2}{21}y_3 + \frac{13}{21}y_4 - \frac{8}{21}y_5 - \frac{1}{21}y_6 - \frac{1}{21}y_7$*. Notice that greatest weight is given to* y_4*, which is the noisy answer to the query that asks directly for* x_1*; but the other output values contribute to the final estimate.*

In summary, whether $m = n$ or $m > n$, Definition 3.2 shows how to derive a unique, consistent estimate $\hat{\mathbf{x}}_\mathbf{A}$ for the true counts $\mathbf{x}$. Once $\hat{\mathbf{x}}_\mathbf{A}$ is computed, the estimate for any $\mathbf{w}$ is computed as $\mathbf{w}\hat{\mathbf{x}}_\mathbf{A}$. The following theorem shows that $\hat{\mathbf{x}}_\mathbf{A}$ is an unbiased estimate of $\mathbf{x}$ and that in a certain sense it is the best possible estimate given the answers to strategy query $\mathbf{A}$.

Theorem 1 (Minimal Variance of estimate of $\mathbf{x}$). *Given noisy output* $\mathbf{y} = \mathcal{L}(\mathbf{A}, \mathbf{x})$*, the estimate* $\hat{\mathbf{x}} = \mathbf{A}^+\mathbf{y}$ *is unbiased (i.e.,* $\mathbb{E}[\hat{\mathbf{x}}_\mathbf{A}] = \mathbf{x}$*), and has the minimum variance among all unbiased estimates that are linear in* $\mathbf{y}$*.*

The theorem follows from an application of the Gauss-Markov theorem [16] and extends a similar result from [11].

3.2 The Matrix Mechanism

In the presentation above, we used the Laplace mechanism to get noisy answers $\mathbf{y}$ to the queries in $\mathbf{A}$, and then derived $\hat{\mathbf{x}}_\mathbf{A}$, from which any workload query could then be estimated. It is convenient to view this technique as a new differentially private mechanism which produces noisy answers to workload $\mathbf{W}$ directly. This mechanism is denoted $\mathcal{M}_\mathbf{A}$ when instantiated with strategy matrix $\mathbf{A}$.

Proposition 3 (Matrix Mechanism). *Let* $\mathbf{A}$ *be a full rank* $m \times n$ *strategy matrix, let* $\mathbf{W}$ *be any* $p \times n$ *workload matrix, and let* $\tilde{\mathbf{b}}$ *be a length-*m *column vector consisting of independent samples from a Laplace distribution with scale* 1*. Then the randomized algorithm* $\mathcal{M}_\mathbf{A}$ *that outputs the following vector is* ϵ*-differentially private:*

$$\mathcal{M}_\mathbf{A}(\mathbf{W}, \mathbf{x}) \quad = \quad \mathbf{W}\mathbf{x} + (\frac{\Delta_\mathbf{A}}{\epsilon})\mathbf{W}\mathbf{A}^+\tilde{\mathbf{b}}.$$

Proof. The expression above can be rewritten as follows:

$$\begin{aligned} \mathcal{M}_\mathbf{A}(\mathbf{W}, \mathbf{x}) &= \mathbf{W}(\mathbf{x} + (\frac{\Delta_\mathbf{A}}{\epsilon})\mathbf{A}^+\tilde{\mathbf{b}}) \\ &= \mathbf{W}\mathbf{A}^+(\mathbf{A}\mathbf{x} + (\frac{\Delta_\mathbf{A}}{\epsilon})\tilde{\mathbf{b}}) \\ &= \mathbf{W}\mathbf{A}^+\mathcal{L}(\mathbf{A}, \mathbf{x}). \end{aligned}$$

Thus, $\mathcal{M}_\mathbf{A}(\mathbf{W}, \mathbf{x})$ is simply a post-processing of the output of the ϵ-differentially private $\mathcal{L}$ and therefore $\mathcal{M}$ is also ϵ-differentially private. □

Like the Laplace mechanism, the matrix mechanism computes the true answer, $\mathbf{W}\mathbf{x}$, and adds to it a noise vector. But in the matrix mechanism the independent Laplace noise $\tilde{\mathbf{b}}$ is transformed by the matrix $\mathbf{W}\mathbf{A}^+$, and then scaled by $\Delta_\mathbf{A}/\epsilon$. The potential power of the mechanism arises from precisely these two features. First, the scaling is proportional to the sensitivity of $\mathbf{A}$ instead of the sensitivity of $\mathbf{W}$, and the former may be lower for carefully chosen $\mathbf{A}$. Second,

$$\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}$$

(a) $\mathbf{I}_4{}^{-1}$

$$\frac{1}{21}\times\begin{bmatrix}3&5&-2&13&-8&-1&-1\\3&5&-2&-8&13&-1&-1\\3&-2&5&-1&-1&13&-8\\3&-2&5&-1&-1&-8&13\end{bmatrix}$$

(b) $\mathbf{H}_4^{+}$

$$\begin{bmatrix}0.25&0.25&0.5&0.0\\0.25&0.25&-0.5&0.0\\0.25&-0.25&0.0&0.5\\0.25&-0.25&0.0&-0.5\end{bmatrix}$$

(c) $\mathbf{Y}_4{}^{-1}$

Figure 2: For strategy $\mathbf{A}$ equal to $\mathbf{I}_4$, $\mathbf{H}_4$ and $\mathbf{Y}_4$, respectively, the matrices above are used to derive the estimate $\hat{\mathbf{x}}_\mathbf{A}$ from the noisy output $\mathbf{y} = \mathcal{L}(\mathbf{A}, \mathbf{x})$. Row i in each matrix contains the coefficients of the linear combination of $\mathbf{y}$ used to construct an estimate for count $\mathbf{x}_i$. The inverse of the identity $\mathbf{I}_4$ is the identity; the reconstruction matrix for $\mathbf{H}_4$ is $\mathbf{H}_4^{+} = (\mathbf{H}_4^t\mathbf{H}_4)^{-1}\mathbf{H}_4^t$; $\mathbf{Y}_4{}^{-1}$ describes the wavelet reconstruction coefficients.

because the noise vector $\tilde{\mathbf{b}}$ consists of independent samples, the Laplace mechanism adds independent noise to each query answer. However, in the matrix mechanism, the noise vector $\tilde{\mathbf{b}}$ is transformed by $\mathbf{W}\mathbf{A}^{+}$. The resulting noise vector is a linear combination of the independent samples from $\tilde{\mathbf{b}}$, and thus it is possible to add *correlated* noise to query answers, which can result in more accurate answers for query workloads whose queries are correlated.

3.3 Rank deficient strategies and workloads

If a workload is not full rank, then it follows from Definition 3.1 that a full rank strategy is not needed. Instead, any strategy whose rowspace spans the workload queries will suffice. However, if we wish to consider a rank deficient strategy $\mathbf{B}$, we can always transform $\mathbf{B}$ into a full rank strategy $\mathbf{A}$ by adding a scaled identity matrix $\delta\mathbf{I}$, where δ approaches zero. The result is a full rank matrix $\mathbf{A}$ which supports all queries, but for which the error for all queries not supported by $\mathbf{B}$ will be extremely high. Another alternative is to apply dimension reduction techniques, such as principle components analysis, to the workload queries to derive a transformation of the domain in which the workload has full rank. Then the choice of full rank strategy matrix can be carried out in the reduced domain.

4. THE ANALYSIS OF ERROR

In this section we analyze the error of the matrix mechanism formally, providing closed-form expressions for the mean squared error of query estimates. We then use matrix decomposition to reveal the properties of the strategy that determine error. This analysis is the foundation for the optimization problems we address in the following section.

4.1 The error of query estimates

While a full rank query strategy $\mathbf{A}$ can be used to compute an estimate for *any* linear query $\mathbf{w}$, the accuracy of the estimate varies based on the relationship between $\mathbf{w}$ and $\mathbf{A}$. We measure the error of strategy $\mathbf{A}$ on query $\mathbf{w}$ using mean squared error.

Definition 4.1 (Query and Workload Error). *Let $\hat{\mathbf{x}}_\mathbf{A}$ be the estimate for $\mathbf{x}$ derived from query strategy $\mathbf{A}$. The mean squared error of the estimate for $\mathbf{w}$ using strategy $\mathbf{A}$ is:*

$$\textsc{Error}_\mathbf{A}(\mathbf{w}) = \mathbb{E}[(\mathbf{w}\mathbf{x} - \mathbf{w}\hat{\mathbf{x}}_\mathbf{A})^2].$$

Given a workload $\mathbf{W}$, the total mean squared error of answering $\mathbf{W}$ using strategy $\mathbf{A}$ is:

$$\textsc{TotalError}_\mathbf{A}(\mathbf{W}) = \sum_{\mathbf{w}_i \in \mathbf{W}} \textsc{Error}_\mathbf{A}(\mathbf{w}_i).$$

For any query strategy $\mathbf{A}$ the following proposition describes how to compute the error for any linear query $\mathbf{w}$ and the total error for any workload $\mathbf{W}$:

Proposition 4 (Error Under Strategy $\mathbf{A}$). *For a full rank query matrix $\mathbf{A}$ and linear query $\mathbf{w}$, the estimate of $\mathbf{w}$ is unbiased (i.e. $\mathbb{E}[\mathbf{w}\hat{\mathbf{x}}_\mathbf{A}] = \mathbf{w}\mathbf{x}$), and the error of the estimate of $\mathbf{w}$ using $\mathbf{A}$ is equal to:*

$$\textsc{Error}_\mathbf{A}(\mathbf{w}) = (\frac{2}{\epsilon^2})\ \Delta_\mathbf{A}^2\ \mathbf{w}(\mathbf{A}^t\mathbf{A})^{-1}\mathbf{w}^t. \qquad (1)$$

The total error of the estimates of workload $\mathbf{W}$ using $\mathbf{A}$ is:

$$\textsc{TotalError}_\mathbf{A}(\mathbf{W}) = (\frac{2}{\epsilon^2})\ \Delta_\mathbf{A}^2\ \mathit{trace}((\mathbf{A}^t\mathbf{A})^{-1}\mathbf{W}^t\mathbf{W}). \quad (2)$$

Proof. It is unbiased because $\hat{\mathbf{x}}_\mathbf{A}$ is unbiased. Thus, for formula (1), the mean squared error is equal to the variance:

$$\begin{aligned}\textsc{Error}_\mathbf{A}(\mathbf{w}) &= Var(\mathbf{w}\hat{\mathbf{x}}_\mathbf{A}) = Var(\mathbf{w}\mathbf{x} + (\frac{\Delta_\mathbf{A}}{\epsilon})\mathbf{w}\mathbf{A}^{+}\tilde{\mathbf{b}})\\ &= (\frac{\Delta_\mathbf{A}}{\epsilon})^2 Var(\mathbf{w}\mathbf{A}^{+}\tilde{\mathbf{b}}).\end{aligned}$$

With algebraic manipulation and that $Var(\tilde{\mathbf{b}}) = 2\mathbf{I}_m$, we get:

$$\begin{aligned}Var(\mathbf{w}\mathbf{A}^{+}\tilde{\mathbf{b}}) &= \mathbf{w}\mathbf{A}^{+}Var(\tilde{\mathbf{b}})(\mathbf{w}\mathbf{A}^{+})^t\\ &= \mathbf{w}\mathbf{A}^{+}2\mathbf{I}_m(\mathbf{w}\mathbf{A}^{+})^t\\ &= 2\mathbf{w}(\mathbf{A}^t\mathbf{A})^{-1}\mathbf{A}^t\mathbf{A}((\mathbf{A}^t\mathbf{A})^{-1})^t\mathbf{w}^t\\ &= 2\mathbf{w}(\mathbf{A}^t\mathbf{A})^{-1}\mathbf{w}^t,\end{aligned}$$

where $((\mathbf{A}^t\mathbf{A})^{-1})^t = (\mathbf{A}^t\mathbf{A})^{-1}$ because the matrix is symmetric. Therefore, $\textsc{Error}_\mathbf{A}(\mathbf{w}) = (\frac{\Delta_\mathbf{A}}{\epsilon})^2 2\mathbf{w}(\mathbf{A}^t\mathbf{A})^{-1}\mathbf{w}^t$.

For formula (2) if $\mathbf{w}_i$ is row i of workload $\mathbf{W}$, then $\textsc{Error}_\mathbf{A}(\mathbf{w}_i)$ is the i-th entry on the diagonal of matrix $(\frac{2}{\epsilon^2})\Delta_\mathbf{A}^2\mathbf{W}(\mathbf{A}^t\mathbf{A})^{-1}\mathbf{W}^t$. Therefore, since the trace of a matrix is the sum of the values on its diagonal, the $\textsc{TotalError}_\mathbf{A}(\mathbf{W})$ is equal to

$$(\frac{2}{\epsilon^2})\Delta_\mathbf{A}^2\mathrm{trace}(\mathbf{W}(\mathbf{A}^t\mathbf{A})^{-1}\mathbf{W}^t).$$

Formula 2 follows from a standard property of the trace: $\mathrm{trace}(\mathbf{W}(\mathbf{A}^t\mathbf{A})^{-1}\mathbf{W}^t) = \mathrm{trace}((\mathbf{A}^t\mathbf{A})^{-1}\mathbf{W}^t\mathbf{W})$. □

These formulas underlie much of the remaining discussion in the paper. Formula (1) shows that, for a fixed ϵ, error is determined by two properties of the strategy: (i) its squared sensitivity, $\Delta_\mathbf{A}^2$; and (ii) the term $\mathbf{w}(\mathbf{A}^t\mathbf{A})^{-1}\mathbf{w}^t$. In the sequel, we refer to the former as simply the *sensitivity term.* We refer to the latter term as the *profile term* and we call matrix $(\mathbf{A}^t\mathbf{A})^{-1}$ the *error profile* of query strategy $\mathbf{A}$.

Definition 4.2 (Error profile). *For any full rank $m \times n$ query matrix $\mathbf{A}$, the error profile of $\mathbf{A}$, denoted $\mathbf{M}$, is defined to be the $n \times n$ matrix $(\mathbf{A}^t\mathbf{A})^{-1}$.*

The coefficients of the error profile $\mathbf{M} = (\mathbf{A}^t\mathbf{A})^{-1}$ measure the covariance of terms in the estimate $\hat{\mathbf{x}}_{\mathbf{A}}$. Element m_{ii} measures the variance of the estimate of x_i in $\hat{\mathbf{x}}_{\mathbf{A}}$ (and is always positive), while m_{ij} measures the covariance of the estimates of x_i and x_j (and may be negative). We can equivalently express the profile term as:

$$\mathbf{w}(\mathbf{A}^t\mathbf{A})^{-1}\mathbf{w}^t = \sum_i w_i^2 m_{ii} + \sum_{i<j} 2w_iw_jm_{ij},$$

which shows that error is a weighted sum of the (positive) diagonal variance terms of $\mathbf{M}$, plus a (possibly negative) linear combination of off-diagonal covariance terms. This illustrates that it is possible to have a strategy that has relatively high error on individual counts yet is quite accurate for other queries that are linear combinations of the individual counts. We analyze instances of such strategies in Sec. 6.

Example 5. *Figure 3 shows the error profiles for each sample strategy. $\mathbf{I}_4$ has the lowest error for queries that ask for a single count of $\mathbf{x}$, such as $\mathbf{w} = [1, 0, 0, 0]$. For such queries the error is determined by the diagonal of the error profile (subject to scaling by the sensitivity term). Queries that involve more than one count will sum terms off the main diagonal and these terms can be negative for the profiles of $\mathbf{H}_4$ and $\mathbf{Y}_4$. Despite the higher sensitivity of these two strategies, the overall error for queries that involve many counts, such as $\mathbf{w} = [1, 1, 1, 1]$, approaches that of $\mathbf{I}_4$. The small dimension of these examples hides the extremely poor performance of $\mathbf{I}_n$ on queries that involve many counts.*

The next example uses Prop. 4 to gain insight into the behavior of some natural strategies for answering a workload of queries.

Example 6. *If $\mathbf{A}$ is the identity matrix, then Prop. 4 implies that the total error will depend only on the workload, since the sensitivity of $\mathbf{I}_n$ is 1:*

$$\textsc{TotalError}_{\mathbf{I}_n}(\mathbf{W}) = (\frac{2}{\epsilon^2})\ \mathit{trace}(\mathbf{W}^t\mathbf{W}).$$

Here the trace of $\mathbf{W}^t\mathbf{W}$ is the sum of squared coefficients of each query. This will tend to be a good strategy for workloads that sum relatively few counts. Assuming the workload is full rank, we can use the workload itself as a strategy, i.e. $\mathbf{A} = \mathbf{W}$. Then Prop. 4 implies that the total error is:

$$\textsc{TotalError}_{\mathbf{W}}(\mathbf{W}) = (\frac{2}{\epsilon^2})\ \Delta_{\mathbf{W}}^2\ n.$$

since $\mathit{trace}((\mathbf{W}^t\mathbf{W})^{-1}\mathbf{W}^t\mathbf{W}) = \mathit{trace}(\mathbf{I}_n) = n$. In this case the trace term is low, but the strategy will perform badly if $\Delta_{\mathbf{W}}$ is high. Note that if $\mathbf{W}$ is $m \times n$, the total error of the Laplace mechanism for $\mathbf{W}$ will be $((\frac{2}{\epsilon^2})\Delta_{\mathbf{W}}^2 m)$, which is worse than the matrix mechanism whenever $m > n$.

In some sense, good strategies fall between the two extremes above: they should have sensitivity less than the workload but a trace term better than the identity strategy.

4.2 Error profile decomposition

Because an error profile matrix $\mathbf{M}$ is equal to $(\mathbf{A}^t\mathbf{A})^{-1}$ for some $\mathbf{A}$, it has a number of special properties. $\mathbf{M}$ is always a square $(n \times n)$ matrix, it is symmetric, and even further, it is always a *positive definite* matrix. Positive definite matrices $\mathbf{M}$ are such that $\mathbf{w}\mathbf{M}\mathbf{w}^t > 0$ for all non-zero vectors $\mathbf{w}$. In our setting this means that the profile term is always positive, as expected. Furthermore, the function $f = \mathbf{w}\mathbf{M}\mathbf{w}^t$ is a quadratic function if $\mathbf{w}$ is viewed as a vector of variables. Then the function f is an elliptic paraboloid over n-dimensional space. If we consider the equation $\mathbf{w}\mathbf{M}\mathbf{w}^t = 1$, this defines an ellipsoid centered at the origin (the solutions to this equation are the queries whose profile term is one). We can think of the error function of strategy $\mathbf{A}$ as a scaled version of this paraboloid, where the scale factor is $(\frac{2}{\epsilon^2})\ \Delta_{\mathbf{A}}^2$.

To gain a better understanding of the error profile, we consider its decomposition. Recall that a matrix is orthogonal if its transpose is its inverse.

Definition 4.3 (Decomposition of Profile). *Let $\mathbf{M}$ be any $n \times n$ positive definite matrix. The spectral decomposition of $\mathbf{M}$ is a factorization of the form $\mathbf{M} = \mathbf{P}_{\mathbf{M}}\mathbf{D}_{\mathbf{M}}\mathbf{P}_{\mathbf{M}}^t$, where $\mathbf{D}_{\mathbf{M}}$ is an $n \times n$ diagonal matrix containing the eigenvalues of $\mathbf{M}$, and $\mathbf{P}_{\mathbf{M}}$ is an orthogonal $n \times n$ matrix containing the eigenvectors of $\mathbf{M}$.*

Thus the matrices $\mathbf{D}_{\mathbf{M}}$ and $\mathbf{P}_{\mathbf{M}}$ fully describe the error profile. They also have an informative geometric interpretation. The entries of the diagonal matrix $\mathbf{D}_{\mathbf{M}}$ describe the relative stretch of the axes of the ellipsoid. The matrix $\mathbf{P}_{\mathbf{M}}$ is orthogonal, representing a rotation of the ellipsoid. In high dimensions, a set of common eigenvalues mean that the ellipsoid is spherical with respect to the corresponding eigen-space. For example, the profile of $\mathbf{I}_4$ is fully spherical (all eigenvalues are one), but by choosing unequal eigenvalues and a favorable rotation, the error is reduced for certain queries.

4.3 Strategy matrix decomposition

Despite the above insights into tuning the error profile, the matrix mechanism requires the choice of a strategy matrix, not simply a profile. Next we focus on the relationship between strategies and their profile matrices.

We will soon see that more than one strategy can result in a given error profile. Accordingly, we define the following equivalence on query strategies:

Definition 4.4 (Profile Equivalence). *Two query matrices $\mathbf{A}$ and $\mathbf{B}$ are* profile equivalent *if their error profiles match, i.e. $(\mathbf{A}^t\mathbf{A})^{-1} = (\mathbf{B}^t\mathbf{B})^{-1}$.*

A key point is that *two profile equivalent strategies may have different sensitivity.* If $\mathbf{A}$ and $\mathbf{B}$ are profile equivalent, but $\mathbf{A}$ has lower sensitivity, then strategy $\mathbf{A}$ dominates strategy $\mathbf{B}$: the estimate for *any* query will have lower error using strategy $\mathbf{A}$.

Example 7. *Recall that strategies $\mathbf{Y}_4$ and $\mathbf{H}_4$ both have sensitivity 3. This is not the minimal sensitivity for strategies achieving either of these error profiles. A square matrix $\mathbf{H}'$, profile equivalent to $\mathbf{H}_4$, is shown in Figure 4(a). This matrix has sensitivity $\Delta_{\mathbf{H}'} = 2.896$. A matrix $\mathbf{Y}'$, profile equivalent to $\mathbf{Y}_4$ is shown in Figure 4(b). This matrix has sensitivity $\Delta_{\mathbf{Y}'} = 2.210$.*

To analyze strategy matrices we again use matrix decomposition, however because a strategy matrix may not be symmetric, we use the singular value decomposition.

$$\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}$$

(a) $(\mathbf{I_4}^t\mathbf{I_4})^{-1}$

$$\tfrac{1}{21}\times\begin{bmatrix}13&-8&-1&-1\\-8&13&-1&-1\\-1&-1&13&-8\\-1&-1&-8&13\end{bmatrix}$$

(b) $(\mathbf{H_4}^t\mathbf{H_4})^{-1}$

$$\tfrac{1}{8}\times\begin{bmatrix}3&-1&0&0\\-1&3&0&0\\0&0&3&-1\\0&0&-1&3\end{bmatrix}$$

(c) $(\mathbf{Y_4}^t\mathbf{Y_4})^{-1}$

Figure 3: The profile term of the error function for query w on strategy A is $\mathbf{w}(\mathbf{A}^t\mathbf{A})^{-1}\mathbf{w}^t$. Shown are the error profiles for query strategies $\mathbf{I}_4$, $\mathbf{H}_4$, and $\mathbf{Y}_4$. Every error profile matrix is symmetric and positive definite.

$$\mathbf{H}'=\begin{bmatrix}-1.32&-1.32&-1.32&-1.32\\0.87&0.87&-0.87&-0.87\\-0.71&0.71&0.00&0.00\\0.00&0.00&-0.71&0.71\end{bmatrix}$$

(a) $\mathbf{H}'$ is profile equivalent to $\mathbf{H}_4$, but $\Delta_{\mathbf{H}'}=2.896$ while $\Delta_{\mathbf{H}_4}=3$

$$\mathbf{Y}'=\begin{bmatrix}1.73&0.58&0.00&0.00\\0.00&1.63&0.00&0.00\\0.00&0.00&1.73&0.58\\0.00&0.00&0.00&1.63\end{bmatrix}$$

(b) $\mathbf{Y}'$ is profile equivalent to $\mathbf{Y}_4$, but $\Delta_{\mathbf{Y}'}=2.210$ while $\Delta_{\mathbf{Y}_4}=3$

Figure 4: When two strategies A and B are profile equivalent, the one with lower sensitivity dominates.

Definition 4.5 (Decomposition of Strategy). *Let* $\mathbf{A}$ *be any* $m\times n$ *query strategy. The singular value decomposition (SVD) of* $\mathbf{A}$ *is a factorization of the form* $\mathbf{A}=\mathbf{Q_A}\mathbf{D_A}\mathbf{P}^t_{\mathbf{A}}$ *such that* $\mathbf{Q_A}$ *is a* $m\times m$ *orthogonal matrix,* $\mathbf{D_A}$ *is a* $m\times n$ *diagonal matrix and* $\mathbf{P_A}$ *is a* $n\times n$ *orthogonal matrix. When* $m>n$*, the diagonal matrix* $\mathbf{D_A}$ *consists of an* $n\times n$ *diagonal submatrix combined with* $\mathbf{0}^{(m-n)\times n}$.

The following theorem shows how the decompositions of the error profile and strategy are related. It explains exactly how a strategy matrix determines the parameters for the error profile, and it fully defines the set of all profile equivalent strategies.

Theorem 2. *Let* $m\geq n$ *and let* $\mathbf{M}$ *be any* $n\times n$ *positive definite matrix decomposed as* $\mathbf{M}=\mathbf{P_M}\mathbf{D_M}\mathbf{P}^t_{\mathbf{M}}$ *where* $\mathbf{D_M}=diag(\lambda_1\ldots\lambda_n)$*. Then for any* $m\times n$ *matrix* $\mathbf{A}$*, the following are equivalent:*

(i) $\mathbf{A}$ *achieves the profile* $\mathbf{M}$*, that is* $(\mathbf{A}^t\mathbf{A})^{-1}=\mathbf{M}$*;*

(ii) There is a decompostion of $\mathbf{A}$ *into* $\mathbf{A}=\mathbf{Q_A}\mathbf{D_A}\mathbf{P}^t_{\mathbf{A}}$ *where* $\mathbf{Q_A}$ *is an* $m\times m$ *orthogonal matrix,* $\mathbf{D_A}$ *is an* $m\times n$ *matrix equal to* $diag(1/\sqrt{\lambda_1}\ldots1/\sqrt{\lambda_n})$ *plus* $\mathbf{0}^{(m-n)\times n}$*, and* $\mathbf{P_A}=\mathbf{P_M}$*.*

Proof. Given (i), let $\mathbf{D}'=diag(\sqrt{\lambda_1}\ldots\sqrt{\lambda_n})$, and since $\mathbf{P_M}$ is an orthogonal matrix $\mathbf{P}^t_{\mathbf{M}}=\mathbf{P_M}^{-1}$. Then

$$\begin{aligned}\mathbf{D}'^t\mathbf{P}^t_{\mathbf{M}}\mathbf{A}^t\mathbf{A}\mathbf{P_M}\mathbf{D}'&=\mathbf{D}'^t\mathbf{P}^t_{\mathbf{M}}\mathbf{M}^{-1}\mathbf{P_M}\mathbf{D}'\\&=\mathbf{D}'^t\mathbf{P}^t_{\mathbf{M}}\mathbf{P_M}\mathbf{D_M}^{-1}\mathbf{P}^t_{\mathbf{M}}\mathbf{P_M}\mathbf{D}'=\mathbf{I}_n.\end{aligned}$$

Thus $\mathbf{A}=\mathbf{Q}'_{\mathbf{A}}\mathbf{D}'^{-1}\mathbf{P}^t_{\mathbf{M}}$ where the $\mathbf{Q}'_{\mathbf{A}}$ is an $m\times n$ matrix whose column vectors are unit length and orthogonal to each other. Let $\mathbf{Q_A}$ be an $m\times m$ orthogonal matrix whose first n columns are $\mathbf{Q}'_{\mathbf{A}}$. Let $\mathbf{D_A}$ be an $m\times n$ matrix equal to $\mathbf{D}'^{-1}$ plus $\mathbf{0}^{(m-n)\times n}$, which is equivalent to $diag(1/\sqrt{\lambda_1}\ldots1/\sqrt{\lambda_n})$ plus $\mathbf{0}^{(m-n)\times n}$. Then

$$\mathbf{Q_A}\mathbf{D_A}\mathbf{P}^t_{\mathbf{M}}=\mathbf{Q}'_{A}\mathbf{D}'^{-1}\mathbf{P}^t_{\mathbf{M}}=\mathbf{A}.$$

Given (ii) we have $\mathbf{A}=\mathbf{Q_A}\mathbf{D_A}\mathbf{P}^t_{\mathbf{M}}$ and we first compute $\mathbf{A}^t\mathbf{A}=(\mathbf{Q_A}\mathbf{D_A}\mathbf{P}^t_{\mathbf{M}})^t(\mathbf{Q_A}\mathbf{D_A}\mathbf{P}^t_{\mathbf{M}})=(\mathbf{P_M}\mathbf{D}^t_{\mathbf{A}}\mathbf{Q}^t_{\mathbf{A}})\,(\mathbf{Q_A}\mathbf{D_A}\mathbf{P}^t_{\mathbf{M}})$ $=(\mathbf{P_M}\mathbf{D}^t_{\mathbf{A}}\mathbf{D_A}\mathbf{P}^t_{\mathbf{M}})$. Note that while $\mathbf{D_A}$ may be $m\times n$, $\mathbf{D}^t_{\mathbf{A}}\mathbf{D_A}$ is an $n\times n$ diagonal matrix equal to $diag(1/\lambda_1\ldots1/\lambda_n)$. Then $(\mathbf{A}^t\mathbf{A})^{-1}=(\mathbf{P_M}\mathbf{D}^t_{\mathbf{A}}\mathbf{D_A}\mathbf{P}^t_{\mathbf{M}})^{-1}=(\mathbf{P_M}(\mathbf{D}^t_{\mathbf{A}}\mathbf{D_A})^{-1}\mathbf{P}^t_{\mathbf{M}})$. Then since $(\mathbf{D}^t_{\mathbf{A}}\mathbf{D_A})^{-1}=diag(\lambda_1\ldots\lambda_n)=\mathbf{D_M}$ we conclude that $(\mathbf{A}^t\mathbf{A})^{-1}=\mathbf{P_M}\mathbf{D_M}\mathbf{P}^t_{\mathbf{M}}$. □

This theorem has a number of implications that inform our optimization problem. First, it shows that given any error profile $\mathbf{M}$, we can construct a strategy that achieves the profile. We do so by decomposing $\mathbf{M}$ and constructing a strategy $\mathbf{A}$ from its eigenvectors (which are contained in $\mathbf{P_M}$ and inherited by $\mathbf{P_A}$) and the diagonal matrix consisting of the inverse square root of its eigenvalues (this is $\mathbf{D_A}$, with no zeroes added). We can simply choose $\mathbf{Q}$ as the $n\times n$ identity matrix, and then matrix $\mathbf{D_A}\mathbf{P}^t_{\mathbf{A}}$ is an $n\times n$ strategy achieving $\mathbf{M}$.

Second, the theorem shows that there are many such strategies achieving $\mathbf{M}$, and that *all* of them can be constructed in a similar way. There is a wrinkle here only because some of these strategies may have more than n rows. That case is covered by the definition of $\mathbf{D_A}$, which allows one or more rows of zeroes to be added to the diagonal matrix derived from the eigenvalues of $\mathbf{M}$. Adding zeroes, $\mathbf{D_A}$ becomes $m\times n$, we choose any $m\times m$ orthogonal matrix $\mathbf{Q_A}$, and we have an $m\times n$ strategy achieving $\mathbf{M}$.

Third, the theorem reveals that the key parameters of the error profile corresponding to a strategy $\mathbf{A}$ are determined by the $\mathbf{D_A}$ and $\mathbf{P_A}$ matrices of the strategy's decomposition. For a fixed profile, the $\mathbf{Q_A}$ of the strategy has no impact on the profile, but does alter the sensitivity of the strategy. Ultimately this means that choosing an optimal strategy matrix requires determining a profile ($\mathbf{D_A}$ and $\mathbf{P_A}$), and choosing a rotation ($\mathbf{Q_A}$) that controls sensitivity. The rotation should be the one that minimizes sensitivity, otherwise the strategy will be dominated.

We cannot find an optimal strategy by optimizing either of these factors independently. Optimizing only for the sensitivity of the strategy severely limits the error profiles possible (in fact, the identity matrix is a full rank strategy with least sensitivity). If we optimize only the profile, we may choose a profile with a favorable "shape" but this could result in a prohibitively high least sensitivity. Therefore we must optimize jointly for the both the profile and the sensitivity and we address this challenge next.

5. OPTIMIZATION

In this section, we provide techniques for determining or approximating optimal query strategies for the matrix mechanism, and we also give some heuristic strategies that may improve existing strategies. We first state our main problem.

PROBLEM 1 (MINERROR). *Given a workload matrix* $\mathbf{W}$*, find the strategy* $\mathbf{A}$ *that minimizes* $\textsc{TotalError}_{\mathbf{A}}(\mathbf{W})$.

The MINERROR problem is difficult for two reasons. First, the sensitivity $\Delta_{\mathbf{A}}$ is the maximum function applied to the L_1 norms of column vectors, which is not differentiable. Second, we do not believe MINERROR is expressible as a convex optimization problem since the set of all query strategies that support a given workload $\mathbf{W}$ is not convex: if $\mathbf{A}$ supports $\mathbf{W}$, then $-\mathbf{A}$ also supports $\mathbf{W}$ but $\frac{1}{2}(\mathbf{A}+(-\mathbf{A}))=\mathbf{0}$ does not support $\mathbf{W}$.

In Section 5.1 we show that MINERROR can be expressed as a semidefinite program with rank constraints. While rank constraints make the semidefinite program non-convex, there are algorithms that can solve such problems by iteratively solving a pair of related semidefinite programs.

Though the set of all query strategies $\mathbf{A}$ that support a given workload $\mathbf{W}$ is not convex, the set of all possible matrices $\mathbf{A}^t\mathbf{A}$ is convex. In Sec. 5.2 we provide two approaches for finding approximate solutions based on bounding $\Delta_{\mathbf{A}}$ by a function of $\mathbf{A}^t\mathbf{A}$ rather than a function of $\mathbf{A}$. While each technique results in a strategy, they essentially select a profile and a default rotation $\mathbf{Q}$. Error bounds are derived by reasoning about the default rotation. It follows that both of these approximations can be improved considering rotations $\mathbf{Q}$ that reduce sensitivity. Therefore we also consider a secondary optimization problem.

PROBLEM 2 (MINSENSITIVITY). *Given a query matrix* $\mathbf{A}$*, find the query matrix* $\mathbf{B}$ *that is profile equivalent to* $\mathbf{A}$ *and has minimum sensitivity.*

Unfortunately this subproblem is still not a convex problem, since the set of all query matrices that are profile equivalent is also not convex. Notice $\mathbf{A}$ is profile equivalent to $-\mathbf{A}$ but is not profile equivalent to $\frac{1}{2}(\mathbf{A}+(-\mathbf{A}))=\mathbf{0}$. Again the problem can be expressed as an SDP with rank constraints. We defer the details to the full version [12].

5.1 Solution to the MINERROR Problem

In this section we formulate the MINERROR problem for an $n\times n$ workload $\mathbf{W}$. It is sufficient to focus on $n\times n$ workloads because Proposition 4 shows that the strategy that minimizes the total error for some workload $\mathbf{W}$ also minimizes the total error for any workload $\mathbf{V}$ such that $\mathbf{V}^t\mathbf{V}=\mathbf{W}^t\mathbf{W}$. Therefore, if given an $m\times n$ workload for $m>n$, we can use spectral decomposition to transform it into an equivalent $n\times n$ workload.

THEOREM 3. *Given an* $n\times n$ *workload* $\mathbf{W}$*, Program 5.1 is a semidefinite program with rank constraint whose solution is the tuple* $(\mathbf{A},\mathbf{C},\mathbf{u},\mathbf{Z})$ *and the* $m\times n$ *strategy* $\mathbf{A}$ *minimizes* $\textsc{TotalError}_{\mathbf{A}}(\mathbf{W})$ *among all* $m\times n$ *strategies.*

Program 5.1 Minimizing the Total Error

Given: $\mathbf{W}\in\mathbb{R}^{n\times n}$

Minimize: $u_1+u_2+\ldots+u_n$

Subject to: For $i\in[n]$: $\mathbf{e}_i$ is the n dimensional column vector whose i^{th} entry is 1 and remaining entries are 0.

$$\begin{bmatrix} 2\mathbf{I}_m & -\mathbf{A}\mathbf{W}^{-1} & \mathbf{0} \\ -(\mathbf{A}\mathbf{W}^{-1})^t & (\mathbf{W}^t)^{-1}\mathbf{Z}\mathbf{W}^{-1} & \mathbf{e}_i \\ \mathbf{0} & \mathbf{e}_i^t & u_i \end{bmatrix} \succeq 0 \qquad (3)$$

For $i\in[n], j\in[m]$:

$$c_{ji}\geq a_{ji}, \quad c_{ji}\geq -a_{ji}, \quad \sum_{k=1}^{m} c_{ki}\leq 1 \qquad (4)$$

$$\text{rank}\left(\begin{bmatrix} \mathbf{I}_m & \mathbf{A} \\ \mathbf{A}^t & \mathbf{Z} \end{bmatrix}\right) = m \qquad (5)$$

PROOF. In Program 5.1, $u_1+\ldots+u_n$ is an upper bound on the total error (modulo constant factors). The rank constraint in Eq. (5) makes sure that $\mathbf{Z}=\mathbf{A}^t\mathbf{A}$.

The semidefinite constraint, Eq. (3), ensures that u_i is an upper bound on twice the error of the i^{th} query in the workload, ignoring for the moment the sensitivity term.

$$u_i \geq 2(\mathbf{W}(\mathbf{A}^t\mathbf{A})^{-1}\mathbf{W}^t)_{ii}$$

To show this, let $\mathbf{X}$ be the $(m+n)\times(m+n)$ upper left submatrix of the matrix in Eq. (3), substituting $\mathbf{A}^t\mathbf{A}$ for $\mathbf{Z}$:

$$\mathbf{X}=\begin{bmatrix} 2\mathbf{I}_m & -\mathbf{A}\mathbf{W}^{-1} \\ -(\mathbf{A}\mathbf{W}^{-1})^t & (\mathbf{W}^t)^{-1}\mathbf{A}^t\mathbf{A}\mathbf{W}^{-1} \end{bmatrix},$$

and then

$$\mathbf{X}^{-1}=\begin{bmatrix} \mathbf{W}(2\mathbf{I}_m-\mathbf{A}\mathbf{A}^{+})^{-1}\mathbf{W}^t & (\mathbf{W}\mathbf{A}^{+})^t \\ \mathbf{W}\mathbf{A}^{+} & 2\mathbf{W}(\mathbf{A}^t\mathbf{A})^{-1}\mathbf{W}^t \end{bmatrix}.$$

The semidefinite constraints in Eq. (3) are equivalent to:

$$\forall i,\ u_i\geq(\mathbf{X}^{-1})_{m+i,m+i}=2(\mathbf{W}(\mathbf{A}^t\mathbf{A})^{-1}\mathbf{W}^t)_{ii}.$$

Thus, minimizing $u_1+\ldots+u_n$ is equivalent to minimizing the trace of $\mathbf{W}(\mathbf{A}^t\mathbf{A})^{-1}\mathbf{W}^t$. To make $u_1+\ldots+u_n$ a bound on the total error, we must show that $\Delta_{\mathbf{A}}=1$. The constraints in Eq. (4) ensure that $\Delta_{\mathbf{A}}\leq 1$. To see that $\Delta_{\mathbf{A}}\geq 1$, observe that $(k\mathbf{X})^{-1}=\frac{1}{k}\mathbf{X}^{-1}$. So $u_1+\ldots u_n$ is minimized when $\Delta_{\mathbf{A}}=1$ because otherwise we can multiply $\mathbf{X}$ (which contains $\mathbf{A}$) by a constant to make $u_1+\ldots+u_n$ smaller. Above all, we have

$$\begin{aligned} u_1+u_2+\ldots+u_n &= 2\sum_{i=1}^{n}(\mathbf{W}(\mathbf{A}^t\mathbf{A})^{-1}\mathbf{W}^t)_{ii} \\ &= 2\text{trace}(\mathbf{W}(\mathbf{A}^t\mathbf{A})^{-1}\mathbf{W}^t) \\ &= 2\Delta_{\mathbf{A}}^2\ \text{trace}(\mathbf{W}(\mathbf{A}^t\mathbf{A})^{-1}\mathbf{W}^t) \\ &= \epsilon^2\textsc{TotalError}_{\mathbf{A}}(\mathbf{W}). \end{aligned}$$

with ϵ fixed. □

Thus Theorem 3 provides the best strategy to the MINERROR problem with at most m queries. Observe that if the optimal strategy has $m'<m$ queries, then Program 5.1 will return an $m\times n$ matrix with $m-m'$ rows of 0s. In addition, if the workload contains queries with coefficients in $\{-1,0,1\}$, we

can show that n^2 is upper bound on the number of queries in the optimal strategy [12].

Dattorro [4] shows that solving a semidefinite program with rank constraints can be converted into solving two semidefinite programs iteratively. The convergence follows the widely used trace heuristic for rank minimization. We are not aware of results that quantify the number of iterations that are required for convergence. However, notice it takes $O(n^3)$ time to solve a semidefinite program with an $n \times n$ semidefinite constraint matrix and in Program 5.1, there are n semidefinite constraint matrices with size $m+n$, which can be represented as a semidefinite constraint matrix with size $n(m+n)$. Thus, the complexity of solving our semidefinite program with rank constraints is at least $O(m^3n^3)$.

5.2 Approximations to the MinError problem

As mentioned above, the MinError problem can be simplified by bounding the sensitivity of $\mathbf{A}$ with some properties of $\mathbf{A}^t\mathbf{A}$. Here we introduce two approximation methods that use this idea and can both be computed efficiently: the L_2 approximation (5.2.1), and the singular value bound approximation (5.2.2). Error bounds on both methods can be measured by providing upper bounds to $\Delta_{\mathbf{A}}$.

5.2.1 L_2 approximation

Note that the diagonal entries of $\mathbf{A}^t\mathbf{A}$ are the squared L_2 norms of column vectors of $\mathbf{A}$. For sensitivity, recall that we are interested in the maximum L_1 norm of the column vectors of $\mathbf{A}$. But this observation leads to the following approaches: we can either use the L_2 norm as an upper bound to the L_1 norm, or we can relax the definition of differential privacy by measuring the sensitivity in terms of L_2 rather than L_1.

Using L_2 norm as an upper bound to L_1 norm. Instead of MinError, we can solve the following L_2 approximation problem. We use $||\mathbf{A}||_2$ to denote the maximum L_2 norm of column vectors of $\mathbf{A}$.

Problem 3 (L_2 approximation). *Given a workload matrix $\mathbf{W}$, find the strategy $\mathbf{A}$ that minimizes*

$$||\mathbf{A}||_2^2 \mathit{trace}(\mathbf{W}(\mathbf{A}^t\mathbf{A})^{-1}\mathbf{W}^t).$$

According to the basic property of L norms, for any vector $\mathbf{v}$ of dimension n, $||\mathbf{v}||_2 \leq \|\mathbf{v}\|_1 \leq \sqrt{n}||\mathbf{v}||_2$. Therefore we can bound the approximation rate of the L_2 approximation.

Theorem 4. *Given a workload $\mathbf{W}$, let $\mathbf{A}$ be the optimal solution to the minError problem and $\mathbf{A}'$ be the optimal solution to the L_2 approximation. Then*

$$\text{TotalError}_{\mathbf{A}'}(\mathbf{W}) \leq n\text{TotalError}_{\mathbf{A}}(\mathbf{W}).$$

Notice the L_2 bound is equal to the L_1 bound if all queries in strategy $\mathbf{A}$ are uncorrelated, so that the L_2 approximation gives the optimal strategy if the optimal strategy only contains uncorrelated queries such as $\mathbf{I}_n$.

Relaxing the definition of differential privacy. L_2 norms can also be applied by relaxing the definition of ϵ-differential privacy into (ϵ, δ)-differential privacy, which is defined as following:

Program 5.2 L_2 approximation

Given: $\mathbf{W} \in \mathbb{R}^{n\times n}$.

Minimize: $u_1 + u_2 + \ldots + u_n$.

Subject to: For $i \in [n] : \mathbf{e}_i$ is the n dimensional column vector whose i^{th} entry s 1 and other entries are 0.

$$\begin{bmatrix} \mathbf{X} & \mathbf{e}_i \\ \mathbf{e}_i^t & u_i \end{bmatrix} \succeq 0;$$

$$(\mathbf{W}^t\mathbf{X}\mathbf{W})_{ii} \leq 1, \quad i \in [n].$$

Definition 5.1 ((ϵ, δ)-differential privacy). *A randomized algorithm $\mathcal{K}$ is (ϵ, δ)-differentially private if for any instance I, any $I' \in nbrs(I)$, and any subset of outputs $S \subseteq Range(\mathcal{K})$, the following holds:*

$$Pr[\mathcal{K}(I) \in S] \leq \exp(\epsilon) \times Pr[\mathcal{K}(I') \in S] + \delta$$

where the probability is taken over the randomness of the $\mathcal{K}$.

The (ϵ, δ)-differential privacy can be achieved by answering each query in strategy $\mathbf{A}$ with i.i.d Gaussian noise

$$N\left(0, \left(\frac{||\mathbf{A}||_2 3\sqrt{n\ln(1/\delta)}}{\epsilon}\right)^2\right). \tag{6}$$

Recall the proof of Proposition 4 and substitute the variance of noises with the variance in Eq. (6). To minimize the total error with (ϵ, δ)-differential privacy guaranteed is equivalent to minimizing

$$\left(\frac{||\mathbf{A}||_2 3\sqrt{n\ln(1/\delta)}}{\epsilon}\right)^2 \text{trace}(\mathbf{W}(\mathbf{A}^t\mathbf{A})^{-1}\mathbf{W}^t),$$

which is equivalent to solving Problem 3.

A semidefinite program (Program 5.2) can be used to solve Problem 3. For a given solution $\mathbf{X}$ of Program 5.2, any $n \times n$ matrix $\mathbf{A}$ such that $\mathbf{X} = \mathbf{A}^t\mathbf{A}$ is a valid solution to Problem 3. Moreover, when δ is given, the Gaussian noise added in the (ϵ, δ)-differential privacy is $\Theta(\frac{n}{\epsilon^2}||\mathbf{A}||_2^2)$. According to the relationship between L_1 and L_2 norm, the Laplace noise added in the ϵ-differential privacy is $O(\frac{n}{\epsilon^2}||\mathbf{A}||_2^2)$, which indicates relaxing the definition of differential privacy does not reduce the amount of noise to be added.

5.2.2 Singular value bound approximation

Another way to bound the L_1 sensitivity is based on its geometry properties. Remember the matrix $\mathbf{A}$ can be repere-sented by its singular value decomposition $\mathbf{A} = \mathbf{Q}_{\mathbf{A}}\mathbf{D}_{\mathbf{A}}\mathbf{P}_{\mathbf{A}}^t$. Let us consider the geometry explanation of the sensitivity. The sensitivity of $\mathbf{A}$ can be considered as the radius of minimum L_1 ball that can cover all column vectors of $\mathbf{A}$, and column vectors of $\mathbf{A}$ lay on the ellipsoid

$$\phi_{\mathbf{A}} : \mathbf{x}^t\mathbf{Q}_{\mathbf{A}}^t(\mathbf{D}_{\mathbf{A}}^t\mathbf{D}_{\mathbf{A}})^{-1}\mathbf{Q}_{\mathbf{A}}\mathbf{x} = 1.$$

Let $\Delta_{\phi_{\mathbf{A}}}$ denotes radius of the minimum L_1 ball that covers the ellipsoid $\phi_{\mathbf{A}}$. Notice all the column vectors of $\mathbf{A}$ are contained in $\phi_{\mathbf{A}}$, which indicates $\Delta_{\mathbf{A}} \leq \Delta_{\phi_{\mathbf{A}}}$. The minimum sensitivity that can be achieved by the strategies that are profile equivalent to $\mathbf{A}$ can be bounded as following:

$$\min_{\mathbf{B}:\mathbf{B}^t\mathbf{B}=\mathbf{A}^t\mathbf{A}} \Delta_{\mathbf{B}} \leq \min_{\mathbf{B}:\mathbf{B}^t\mathbf{B}=\mathbf{A}^t\mathbf{A}} \Delta_{\phi_{\mathbf{B}}}.$$

The matrix $\mathbf{B}$ that is profile equivalent to $\mathbf{A}$ and has the minimum $\Delta_{\phi_{\mathbf{B}}}$ is given by the theorem below.

THEOREM 5. *Let $\mathbf{A}$ be a matrix with singular value decomposition $\mathbf{A} = \mathbf{Q_A D_A P_A^t}$ and $\delta_1, \delta_2, \ldots, \delta_n$ be its singular values. Then*

$$\underset{\mathbf{B}\,:\,\mathbf{B}^t\mathbf{B}=\mathbf{A}^t\mathbf{A}}{\operatorname{argmin}} \Delta_{\phi_{\mathbf{B}}} = \mathbf{D_A P_A^t},$$

$$\min_{\mathbf{B}\,:\,\mathbf{B}^t\mathbf{B}=\mathbf{A}^t\mathbf{A}} \Delta_{\phi_{\mathbf{B}}} = \sqrt{\delta_1^2 + \delta_2^2 + \ldots + \delta_n^2} \leq \sqrt{n}\Delta_{\mathbf{A}}. \quad (7)$$

Using the singular value bound in Theorem 5 to substitute for the L_1 sensitivity, the *minError* problem can be converted to the following approximation problem.

PROBLEM 4 (SINGULAR VALUE BOUND APPROXIMATION). *Given a workload matrix $\mathbf{W}$, find the strategy $\mathbf{A}$ that minimizes*

$$(\delta_1^2 + \delta_2^2 + \ldots + \delta_n^2) \mathit{trace}(\mathbf{W}(\mathbf{A}^t\mathbf{A})^{-1}\mathbf{W}^t),$$

where $\delta_1, \delta_2, \ldots, \delta_n$ are singular values of $\mathbf{A}$.

The singular value bound approximation has a closed-form solution.

THEOREM 6. *Let $\mathbf{W}$ be the workload matrix with singular value decomposition $\mathbf{W} = \mathbf{Q_W D_W P_W^t}$ and $\delta'_1, \delta'_2, \ldots, \delta'_n$ be its singular values. The optimal solution $\mathbf{D_A}$, $\mathbf{P_A}$ to the singular value bound approximation is to let $\mathbf{P_A} = \mathbf{P_W}$ and $\mathbf{D_A} = \mathit{diag}(\sqrt{\delta'_1}, \sqrt{\delta'_2}, \ldots, \sqrt{\delta'_n})$.*

The solution in Theorem 6 is very similar to the strategy metioned at the end of Sec. 4 that matches $\mathbf{P_A}$ to $\mathbf{P_W}$ and $\mathbf{D_A}$ be $\mathit{diag}(\delta'_1, \delta'_2, \ldots, \delta'_n)$. We use a slightly different $\mathbf{D_A}$ so as to provide an guaranteed error bound based on Theorem 5.

THEOREM 7. *Given a workload $\mathbf{W}$, let $\mathbf{A}$ be the optimal solution to the minError problem and $\mathbf{A}'$ be the optimal solution to the singular value bound approximation. Then*

$$\text{TOTALERROR}_{\mathbf{A}'}(\mathbf{W}) \leq n\text{TOTALERROR}_{\mathbf{A}}(\mathbf{W}).$$

5.3 Augmentation Heuristic

We formalize below the following intuition: as far as the error profile is concerned, additional noisy query answers can never detract from query accuracy as they must have some information content useful to one or more queries. Therefore the error profile can never be worse after augmenting the query strategy by adding rows.

THEOREM 8. *[Augmenting a strategy] Let $\mathbf{A}$ be a query strategy with full rank and consider a new strategy $\mathbf{A}'$ obtained from $\mathbf{A}$ by adding the additional rows of strategy $\mathbf{B}$, so that $\mathbf{A}' = [\begin{smallmatrix}\mathbf{A}\\ \mathbf{B}\end{smallmatrix}]$. For any query $\mathbf{w}$, we have:*

$$\mathbf{w}^t(\mathbf{A}'^t\mathbf{A}')^{-1}\mathbf{w} \leq \mathbf{w}^t(\mathbf{A}^t\mathbf{A})^{-1}\mathbf{w}$$

Further, $\mathbf{w}^t(\mathbf{A}'^t\mathbf{A}')^{-1}\mathbf{w} = \mathbf{w}^t(\mathbf{A}^t\mathbf{A})^{-1}\mathbf{w}$ only for the queries in the set $\{\mathbf{A}^t\mathbf{A}\mathbf{w} \mid \mathbf{B}\mathbf{w} = 0\}$, which is non-empty if and only if $\mathbf{B}$ does not have full column rank.

The proof is included in [12].

This improvement in the error profile may have a cost—namely, augmenting $\mathbf{A}$ with strategy $\mathbf{B}$ may lead to a strategy $\mathbf{A}'$ with greater sensitivity than $\mathbf{A}$. A heuristic that follows from Theorem 8 is to augment strategy $\mathbf{A}$ only by completing deficient columns, that is, by adding rows with non-zero entries only in columns whose absolute column sums are less the sensitivity of $\mathbf{A}$. In this case the augmentation does not increase sensitivity and is guaranteed to strictly improve accuracy for any query with a non-zero coefficient in an augmented column.

Our techniques could also be used to reason formally about augmentations that do incur a sensitivity cost. We leave this as future work, as it is relevant primarily to an interactive differentially private mechanism which is not our focus here.

6. APPLICATIONS

In this section we use our techniques to analyze and improve existing approaches. We begin by analyzing two techniques proposed recently [17, 11]. Both strategies can be seen as instances of the matrix mechanism, each using different query strategies designed to support a workload consisting of all range queries. Although both techniques can support multidimensional range queries, we focus our analysis on one dimensional range queries, i.e. interval queries with respect to a total order over $dom(\mathbb{B})$.

We will show that the seemingly distinct approaches have remarkably similar behavior: they have low (but not minimal) sensitivity, and they are highly accurate for range queries but much worse for queries that are not ranges. We describe these techniques briefly and how they can each be represented in matrix form.

In the *hierarchical* scheme proposed in [11], the query strategy can be envisioned as a recursive partitioning of the domain. We consider the simple case of a binary partitioning. First we ask for the total sum over the whole domain, and then ask for the count of each half of the domain, and so on, terminating with counts of individual elements of the domain. For a domain of size n (assumed for simplicity to be a power of 2), this results in a query strategy consisting of $2n - 1$ rows. We represent this strategy as matrix $\mathbf{H}_n$, and $\mathbf{H}_4$ in Fig. 1 is a small instance of it.

In the *wavelet* scheme, proposed in [17], query strategies are based on the Haar wavelet. For one dimensional range queries, the technique can also be envisioned as a hierarchical scheme, asking the total query, then asking for the difference between the left half and right half of the domain, continuing to recurse, asking for the difference in counts between each binary partition of the domain at each step.[1] This results in n queries—fewer than the hierarchical scheme of [11]. The matrix corresponding to this strategy is the matrix of the Haar wavelet transform, denoted $\mathbf{Y}_n$, and $\mathbf{Y}_4$ in Fig. 1 is a small instance of it.

Thus $\mathbf{H}_n$ is a rectangular $(2n-1)\times n$ strategy, with answers derived using the linear regression technique, and $\mathbf{Y}_n$ is an $n \times n$ strategy with answers derived by inverting the strategy matrix. As suggested by the examples in earlier sections, these seemingly different techniques have similar behavior. We analyze them in detail below, proving new bounds on the error for each technique, and proving new results about their relationship to one another. We also include $\mathbf{I}_n$ in the analysis, which is the strategy represented by the dimension n identity matrix, which asks for each individual count.

[1] We note that the technique in [17] is presented somewhat differently, but that the differences are superficial. The authors use queries that compute averages rather than sums, and their differentially private mechanism adds scaled noise at each level in the hierarchy. We prove the equivalence of that construction with our formulation $\mathbf{Y}_n$ in [12].

6.1 Geometry of $\mathbf{I}_n$, $\mathbf{H}_n$ and $\mathbf{Y}_n$

Recall from Section 4 that the decomposition of the error profile of a strategy explains its error. The decomposition of $\mathbf{I}_n$ results in a $\mathbf{D}$ that is itself the identity matrix. This means the error profile is spherical. To understand the shape and rotation of the error profiles for $\mathbf{Y}_n$ and $\mathbf{H}_n$ we provide a complete analysis of the decomposition, but leave the details in [12]. Their eigenvalue distributions are remarkably similar. Each has $\log n + 1$ distinct eigenvalues of geometrically increasing frequency. The actual eigenvalues of $\mathbf{H}_n$ are smaller than those of $\mathbf{Y}_n$ by exactly one throughout the increasing sequence, except the largest eigenvalue: it is equal to the second largest eigenvalue in $\mathbf{Y}_n$, but it has a distinct value in $\mathbf{H}_n$. Finally, the smallest eigenvalue of either approach is 1 and the ratio between their corresponding eigenvalues is in the range $[\frac{1}{2}, 2]$.

For sensitivity, it is clear that $\Delta_{\mathbf{I}_n} = 1$ for all n. Intuitively, this sensitivity should be minimal since the columns of I_n are axis aligned and orthogonal, and any rotation of $\mathbf{I}_n$ can only increase the L_1 ball containing the columns of $\mathbf{I}_n$. This intuition can be formalized by considering the relationship between the L_1 norm and the L_2 norm stated in Section 5.2.1. No strategy profile equivalent to I_n can have lower sensitivity, since $\Delta_{I_n} = ||I_n||_2 = 1$.

On the other hand, the sensitivity of $\mathbf{Y}_n$ and $\mathbf{H}_n$ is not minimal, suggesting that there exist strategies that dominate both of them. We have $\Delta_{\mathbf{Y}_n} = \Delta_{\mathbf{H}_n} = \log_2 n + 1$. In addition we find that their L_2 norms are also equal: $||\mathbf{Y}_n||_2 = ||\mathbf{H}_n||_2 = \sqrt{\log_2 n + 1}$. This L_2 norm is a lower bound on the sensitivity of profile equivalent strategies for both $\mathbf{H}_n$ and $\mathbf{Y}_n$. We do not know if there are profile equivalent strategies that achieve this sensitivity lower bound for these strategies. We can, however, improve on the sensitivity of both. As an example, Fig. 4 shows profiles equivalent to $\mathbf{H}_4$ and $\mathbf{Y}_4$ with improved sensitivity. Through our decomposition of $\mathbf{H}_n$ and $\mathbf{Y}_n$ we have derived modest improvements on the sensitivity in the case of arbitrary $n \geq 8$: $\log n + 0.64$ for $\mathbf{H}_n$, which is the sensitivity of its decomposition, and $\log n + 2\sqrt{2} - 4$ for $\mathbf{Y}_n$, which is achieved by applying some minor modifications to its decomposition. We suspect it is possible to find rotations of $\mathbf{H}_n$ and $\mathbf{Y}_n$ that improve more substantially on the sensitivity.

6.2 Error analysis for $\mathbf{I}_n$,$\mathbf{H}_n$ and $\mathbf{Y}_n$

In this section we analyze the total and worst case error for specific workloads of interest. We focus on two typical workloads: $\mathbf{W}_R$, the set of all range queries, and $\mathbf{W}_{01}$, which includes arbitrary predicate queries, since it consists of all linear queries 0-1 queries. Note that attempting to use either of these workloads as strategies leads to poor results: the sensitivity of $\mathbf{W}_R$ is $O(n^2)$ while the sensitivity of $\mathbf{W}_{01}$ is $O(2^n)$.

In the original papers describing $\mathbf{H}_n$ and $\mathbf{Y}_n$ [11, 17], both techniques are shown to have worst case error bounded by $O(\log^3 n)$ on $\mathbf{W}_R$. Both papers resort to experimental analysis to understand the distribution of error across the class of range queries. We note that our results allow error for any query to be analyzed analytically.

It follows from the similarity of eigenvectors and eigenvalues of $\mathbf{H}_n$ and $\mathbf{Y}_n$ that the error profiles are asymptotically equivalent to one another. We thus prove a close equivalence between the error of the two techniques:

THEOREM 9. *For any linear counting query* $\mathbf{w}$,

$$\frac{1}{2}\textsc{Error}_{\mathbf{Y}}(\mathbf{w}) \leq \textsc{Error}_{\mathbf{H}}(\mathbf{w}) \leq 2\textsc{Error}_{\mathbf{Y}}(\mathbf{w}).$$

Next we summarize the maximum and total error for these strategies. The following results tighten known bounds for $\mathbf{W}_R$, and show new bounds for $\mathbf{W}_{01}$. The proof of the following theorem can be found in [12].

THEOREM 10 (MAXIMUM AND TOTAL ERROR). *The maximum and total error on workloads* $\mathbf{W}_R$ *and* $\mathbf{W}_{01}$ *under strategies* $\mathbf{H}_n, \mathbf{Y}_n$, *and* $\mathbf{I}_n$ *is given by:*

MAXERROR	$\mathbf{H}_n$	$\mathbf{Y}_n$	$\mathbf{I}_n$
$\mathbf{W}_R$	$\Theta(\log^3 n)$	$\Theta(\log^3 n)$	$\Theta(n)$
$\mathbf{W}_{01}$	$\Theta(n \log^2 n)$	$\Theta(n \log^2 n)$	$\Theta(n)$

TOTALERROR	$\mathbf{H}_n$	$\mathbf{Y}_n$	$\mathbf{I}_n$
$\mathbf{W}_R$	$\Theta(n^2 \log^3 n)$	$\Theta(n^2 \log^3 n)$	$\Theta(n^3)$
$\mathbf{W}_{01}$	$\Theta(n2^n \log^2 n)$	$\Theta(n2^n \log^2 n)$	$\Theta(n2^n)$

While $\mathbf{H}_n$ and $\mathbf{Y}_n$ achieve similar asymptotic bounds, their error profiles are slightly different (as suggested by previous examples for $n = 4$). As a result, $\mathbf{H}_n$ tends to have lower error for larger range queries, while $\mathbf{Y}_n$ has lower error for unit counts and smaller range queries.

7. RELATED WORK

Since differential privacy was first introduced [8], it has been the subject of considerable research, as outlined in recent surveys [5, 6, 7].

Closest to our work are the two techniques, developed independently, for answering range queries over histograms. Xiao et al. [17] propose an approach based on the Haar wavelet; Hay et al. [11] propose an approach based on hierarchical sums and least squares. The present work unifies these two apparently disparate approaches under a significantly more general framework (Section 3) and uses the framework to compare the approaches (Section 6). While both approaches are instances of the matrix mechanism, the specific algorithms given in these papers are more efficient than a generic implementation of the matrix mechanism employing matrix inversion. Xiao et al. also extend their wavelet approach to nominal attributes and multi-dimensional histograms.

Barak et al. [1] consider a Fourier transformation of the data to estimate low-order marginals over a set of attributes. The main utility goal of [1] is integral consistency: the numbers in the marginals must be non-negative integers and their sums should be consistent across marginals. Their main result shows that it is possible to achieve integral consistency (via Fourier transforms and linear programming) without significant loss in accuracy. We would like to use the framework of the matrix mechanism to further investigate optimal strategies for workloads consisting of low-order marginals.

Blum et al. [2] propose a mechanism for accurately answering queries for an arbitrary workload (aka query class), where the accuracy depends on the VC-dimension of the query class. However, the mechanism is inefficient, requiring exponential runtime. They also propose an efficient strategy for the class of range queries, but this approach is less accurate than the wavelet or hierarchical approaches discussed here (see Hay et al. [11] for comparison).

Some very recent works consider improvements on the Laplace mechanism for multiple queries. Hardt and Talwar [10] consider a very similar task based on sets of linear queries. They propose the k-norm mechanism, which adds noise tailored to the set of linear queries by examining the shape to which the linear queries map the L_1 ball. They also show an interesting lower bound on the noise needed for satisfying differential privacy that matches their upper bound up to polylogarithmic factors assuming the truth of a central conjecture in convex geometry. But the proposed k-norm mechanism can be inefficient in practice because of its requirement of sampling uniformly from high-dimensional convex bodies. Furthermore, the techniques restrict the number of queries to be less than n (the domain size). A notable difference in our approach is that our computational cost is incurred for finding the query strategy. Once a strategy is found, our mechanism is as efficient as the Laplace mechanism. For stable or recurring workloads, optimization needs only to be performed once.

Roth and Roughgarden [15] consider the interactive setting, in which queries arrive over time and must be answered immediately without knowledge of future queries. They propose the median mechanism which improves upon the Laplace mechanism by deriving answers to some queries from the noisy answers already received from the private server. The straightforward implementation of the median mechanism is inefficient and requires sampling from a set of super-polynomial size, while a more efficient polynomial implementation requires weakening the privacy and utility guarantees to average-case notions (i.e., guarantees hold for most but not all input datasets).

The goal of optimal experimental design [14] is to produce the best estimate of an unknown vector from the results of a set of experiments returning noisy observations. Given the noisy observations, the estimate is typically the least squares solution. The goal is to minimize error by choosing a subset of experiments and a frequency for each. A relaxed version of the experimental design problem can be formulated as a semi-definite program [3]. While this problem setting is similar to ours, a difference is that the number and choice of experiments is constrained to a fixed set. In addition, although experimental design problems can include costs associated with individual experiments, modeling the impact of the sensitivity of experiments does not fit most problem formulations. Lastly, the objective function of most experimental design problems targets the accuracy of individual variables (the $\mathbf{x}$ counts), rather than a specified workload computed from those counts.

8. CONCLUSION

We have described the matrix mechanism, which derives answers to a workload of counting queries from the noisy answers to a different set of strategy queries. By designing the strategy queries for the workload, correlated sets of counting queries can be answered more accurately. We show that the optimal strategy can be computed by iteratively solving a pair of semidefinite programs, and we use our framework to understand two recent techniques targeting range queries.

While we have formulated the choice of strategy matrix as an optimization problem, we have not yet generated optimal—or approximately optimal—solutions for specific workloads. Computing such optimal strategies for common workloads would have immediate practical impact as it could boost the accuracy that is efficiently achievable under differential privacy. We also plan to apply our approach to interactive query answering settings, and we would like to understand the conditions under which optimal strategies in our framework can match known lower bounds for differential privacy.

Acknowledgements

We would like to thank the anonymous reviewers for their helpful comments. The NSF supported authors Hay, Li, Miklau through IIS-0643681 and CNS-0627642, author Rastogi through IIS-0627585, and author McGregor through CCF-0953754.

9. 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 *PODS*, 2007.

[2] A. Blum, K. Ligett, and A. Roth. A learning theory approach to non-interactive database privacy. In *STOC*, 2008.

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

[4] J. Dattorro. *Convex optimization & Euclidean distance geometry.* Meboo Publishing USA, 2005.

[5] C. Dwork. Differential privacy: A survey of results. In *TAMC*, 2008.

[6] C. Dwork. The differential privacy frontier. In *TCC*, 2009.

[7] C. Dwork. A firm foundation for privacy. In *To Appear, CACM*, 2010.

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

[9] A. Ghosh, T. Roughgarden, and M. Sundararajan. Universally utility-maximizing privacy mechanisms. In *STOC*, 2009.

[10] M. Hardt and K. Talwar. On the geometry of differential privacy. In *STOC*, 2010.

[11] M. Hay, V. Rastogi, G. Miklau, and D. Suciu. Boosting the accuracy of differentially-private histograms through consistency. In *Proceedings of the VLDB*, 2010. (also available as CoRR abs/0904.0942 2009).

[12] C. Li, M. Hay, V. Rastogi, G. Miklau, and A. McGregor. Optimizing histogram queries under differential privacy. *CoRR*, abs/0912.4742, 2009.

[13] K. Nissim, S. Raskhodnikova, and A. Smith. Smooth sensitivity and sampling in private data analysis. In *STOC*, pages 75–84, 2007.

[14] F. Pukelsheim. *Optimal Design of Experiments.* Wiley & Sons, 1993.

[15] A. Roth and T. Roughgarden. The median mechanism: Interactive and efficient privacy with multiple queries. In *STOC*, 2010.

[16] S. D. Silvey. *Statistical Inference.* Chap. & Hall, 1975.

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