Optimal error of query sets under the differentially-private matrix mechanism

Chao Li
University of Massachusetts
Amherst, MA, USA
chaoli@cs.umass.edu

Gerome Miklau
University of Massachusetts
Amherst, MA, USA
INRIA Saclay
France
miklau@cs.umass.edu

ABSTRACT

A common goal of privacy research is to release synthetic data that satisfies a formal privacy guarantee and can be used by an analyst in place of the original data. To achieve reasonable accuracy, a synthetic data set must be tuned to support a specified set of queries accurately, sacrificing fidelity for other queries.

This work considers methods for producing synthetic data under differential privacy and investigates what makes a set of queries "easy" or "hard" to answer. We consider answering sets of linear counting queries using the matrix mechanism [18], a recent differentially-private mechanism that can reduce error by adding complex correlated noise adapted to a specified workload.

Our main result is a novel lower bound on the minimum total error required to simultaneously release answers to a set of workload queries. The bound reveals that the hardness of a query workload is related to the spectral properties of the workload when it is represented in matrix form. The bound is most informative for (ϵ, δ)-differential privacy but also applies to ϵ-differential privacy.

Categories and Subject Descriptors

H.2.8 [**Database Applications**]: Statistical databases; F.2 [**Analysis of Algorithms and Problem Complexity**]: General

General Terms

Algorithms, Theory

Keywords

Differential Privacy, Matrix Mechanism, Lower Bound

1. INTRODUCTION

Differential privacy [9] is a rigorous privacy standard offering participants in a data set the appealing guarantee that released query answers will be nearly indistinguishable whether or not their data is included. The earliest methods for achieving differential privacy were interactive: an analyst submits a query to the server and receives a noisy query answer. Further queries may be submitted, but increasing noise will be added and the server may eventually refuse to answer subsequent queries.

To avoid some of the challenges of the interactive model, differential privacy has often been adapted to a non-interactive setting where a common goal has been to release a synthetic data set that the analyst can use in place of the original data. There are a number of appealing benefits to releasing a private synthetic database: the analyst need not carefully divide their task into individual queries and can use familiar data processing techniques on the synthetic data; the privacy budget will not be exhausted before the queries of interest have been answered; and data processing can be carried out using the resources of the analyst without revealing tasks to the data owner.

There are limits, however, to private synthetic data generation. When a synthetic dataset is released, the server no longer controls how many questions the analyst computes from the data. Dinur and Nissim showed that accurately answering "too many" queries of a certain type is incompatible with any reasonable notion of privacy, allowing reconstruction of the database with high probability [7].

This tempers the hopes of private synthetic data to some degree, suggesting that if a synthetic dataset is to be private, then it can be accurate only for a specific class of queries and may need to sacrifice accuracy for other queries. A number of methods have been proposed for releasing accurate synthetic data for specific sets of queries [6, 18, 16, 27, 28, 4, 1, 25, 30]. These results show that it is still possible to achieve many of the benefits of synthetic data if the released data is targeted to a *workload* of queries that are of interest to the analyst.

In general, efficient differentially private algorithms for answering sets of queries with minimum error are not known. The goal of our work is to develop tools that can explain what we informally term the *error complexity* of a given workload, which should measure, for fixed privacy parameters, the accuracy with which we can simultaneously answer all queries in the workload.

Such tools can help us to answer a number of natural questions that arise in the context of private synthetic data generation. Why is it possible to answer one set of queries more accurately than another? What properties of the queries, or of their relationship to one another, influence this? Can lower error be achieved by specializing the query set more closely to the task at hand? Does the combination of mul-

EDBT/ICDT'13, March 18-22, 2013, Genoa, Italy.

tiple users' workloads severely impact the accuracy possible for the combined workload?

Naive approaches to understanding the "hardness" of a query workload are unsatisfying. For example, one may naturally expect that the greater the number of queries in the workload, the larger the error in simultaneously answering them. Yet the number of queries in a workload is usually an inadequate measure of its hardness. Query workload sensitivity [9] is another natural approach. Sensitivity measures the maximum change in all query answers due to an insertion or deletion of a single database record. Basic differentially private mechanisms (e.g. the Laplace mechanism) add noise to each query in proportion to sensitivity, and in such cases sensitivity does in fact determine error rates. But better mechanisms can reduce error when answering multiple queries (with no cost to privacy), so that sensitivity alone fails to be a reliable measure.

In this paper we seek a better understanding of workload error complexity by reasoning formally about the minimum error achievable for a workload, regardless of the underlying database. We pursue this goal in the context of a class differentially private algorithms: namely those that are instances of the matrix mechanism (so named because workloads are represented as matrices and analyzed algebraically). The matrix mechanism can be used to answer sets of linear counting queries, a general class of queries which includes all predicate counting queries, histogram queries, marginals, data cubes, and others.

The matrix mechanism [18] exploits the relationships between queries in the workload to construct a complex, correlated noise distribution that offers lower error than standard mechanisms. It encompasses a range of possible approaches to differentially-private query answering because it must be instantiated with a set of queries, called the "strategy", which it uses to derive accurate answers to the workload queries. This makes the mechanism quite general, since any strategy can be selected. It includes as a special case a number of recently-proposed techniques for answering various subsets of the class of linear queries including range-count queries, sets of low order marginals, sets of data cubes [1, 27, 16, 6, 19, 30, 29]. Each of the above techniques can be seen as selecting strategies (either manually or adaptively) that work well for given workloads. The lower bound presented in this work provides a theoretical method of evaluating the quality of the approaches since it provides a lower bound on the error attained by the best possible strategy.

We use the optimal error achievable under the matrix mechanism as a proxy for workload error complexity. It is computationally infeasible to compute the optimal strategy for an arbitrary workload, making the assessment of workload complexity a challenge. Nevertheless, we are able to resolve this challenge through the following contributions:

- Our main result is a novel lower bound on the minimum total error required to simultaneously release answers to a set of workload queries. The bound reveals that the "hardness" of a query workload is related to the eigenvalues of the workload when it is represented in matrix form.
- Under (ϵ, δ)-differential privacy, we characterize two important classes of workloads for which our lower bound is tight. As a consequence, it is possible to directly construct a minimum error mechanism for workloads in these classes. We also analyze the cases for which our bound is not tight, including when the bound is adapted to ϵ-differential privacy.
- We compare our lower bound to corresponding bounds on the achieved error of recently-proposed mechanisms [26, 14, 12, 11].

Note that our lower bound on error is a conditional bound: it holds for the class of mechanisms defined by the matrix mechanism, but not necessarily for all differentially private mechanisms. Nevertheless, we believe this conditional bound serves as a widely useful tool. First, it helps to resolve a number of open questions about the quality of previously-proposed mechanisms that are instances of the matrix mechanism [18, 27, 1, 6, 19, 30]. Second, emerging techniques that can outperform this bound tend to exploit special properties of the input database. Therefore, the achievable error of those mechanisms no longer reflects only properties of the analysis task (as embodied by the workload) but instead reflects the interaction of the task with the database. Third, the data-independence of the matrix mechanism makes deployment particularly efficient since the noise distribution is fixed for all input databases once a strategy has been selected. Our bound therefore helps to clarify the utility possible using data-independent mechanisms and reveals when other methods may be required.

The organization of the paper is as follows. Section 2 reviews definitions and Section 3 presents the matrix mechanism and properties of workload error. In Section 4 we present the lower bound and its proof. Section 5 evaluates the tightness and looseness of the bound. We compare our bound with error rates of data-dependent mechanisms in Section 6. To aid intuition, we include throughout the paper a series of examples in which we compute our lower bound on workloads of interest and report concrete error rates. Due to lack of space, most of proofs are omitted and can be found in the full version of this paper [20].

2. DEFINITIONS & BACKGROUND

In this section we describe our representation of query workloads as matrices, formally define differential privacy, and review linear algebra notation.

2.1 Data model & linear queries

The queries considered in this paper are all counting queries over a single relation. Let the database I be an instance of a single-relation schema $R(\mathbb{A})$, with attributes $\mathbb{A} = \{A_1, A_2, \ldots, A_m\}$. The crossproduct of the attribute domains, written $dom(\mathbb{A}) = dom(A_1) \times \cdots \times dom(A_m)$, is the set of all possible tuples that may occur in I.

In order to express our queries, we encode the instance I as a vector $\mathbf{x}$ consisting of *cell counts*, each counting the number of tuples in I satisfying a distinct logical cell condition.

Definition 2.1 (Cell Conditions). *A cell condition is a Boolean formula which evaluates to True or False on any tuple in $dom(\mathbb{A})$. A collection of cell conditions $\Phi = \phi_1, \phi_2 \ldots \phi_n$ is an ordered list of pairwise unsatisfiable cell conditions: each tuple in $dom(\mathbb{A})$ will satisfy at most one ϕ_i.*

The data vector is formed from cell counts corresponding to a collection of cell conditions.

DEFINITION 2.2 (DATA VECTOR). *Given instance I and a collection of cell conditions $\Phi = \phi_1, \phi_2 \dots \phi_n$, the data vector $\mathbf{x}$ is the length-n column vector consisting of the non-negative integral counts $x_i = |\{t \in I \mid \phi_i(t) \text{ is True}\}|$.*

We may choose to fully represent instance I by defining the vector $\mathbf{x}$ with one cell for every element of $dom(\mathbb{A})$. Then $\mathbf{x}$ is a bit vector of size $|dom(\mathbb{A})|$ with nonzero counts for each tuple present in I. (This is also a vector representation of the full contingency table built from I.) Alternatively, it may be sufficient to partially represent I by the cell counts in $\mathbf{x}$, for example by focusing on a subset of the attributes of $\mathbb{A}$ that are relevant to a particular workload of interest. Because workloads are finite, it is always sufficient to consider a finite list of cell conditions, even if attribute domains are infinite.

EXAMPLE 2.3. *Consider the relational schema $R = (name, gradyear, gender, gpa)$ describing students and suppose we wish to form queries over gradyear and gender only, where $dom(gender) = \{M, F\}$ and $dom(gradyear) = \{2011, 2012, 2013, 2014\}$. Then we can define 8 cell conditions which result from all combinations of gradyear and gender. These are enumerated in Table 1(a).*

Given a data vector $\mathbf{x}$, queries are expressed as linear combinations of the cell counts in $\mathbf{x}$.

DEFINITION 2.4 (LINEAR COUNTING QUERY). *A* linear counting 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 counting query $\mathbf{q}$ on $\mathbf{x}$ is the vector product $\mathbf{qx} = q_1x_1 + \dots + q_nx_n$.*

If $\mathbf{q}$ consists exclusively of coefficients in $\{0, 1\}$, then $\mathbf{q}$ is called a *predicate counting query*. In this case, $\mathbf{q}$ counts the number of tuples in I that satisfy the union of the cell conditions corresponding to the nonzero coefficients in $\mathbf{q}$.

2.2 Query workloads

A *workload* is a finite set of linear queries. A workload is represented as a matrix, each row of which is a single linear counting query.

DEFINITION 2.5 (QUERY MATRIX). *A* query matrix *is a collection of m unique linear counting queries, arranged by rows to form an $m \times n$ matrix.*

Note that cell condition ϕ_i defines the meaning of the i^{th} position of $\mathbf{x}$, and accordingly, it determines the meaning of the i^{th} column of $\mathbf{W}$. Unless otherwise noted, we assume all workloads are defined over the same fixed set of cell conditions.

EXAMPLE 2.6. *The matrix in Table 1(b) shows a workload of five queries. The first four are predicate queries. Table 1(c) describes the meaning of the queries w.r.t. the cell conditions in Table 1(a).*

We assume that workloads consist of unique queries, without duplicates. If workload $\mathbf{W}$ is an $m \times n$ query matrix, the answers for $\mathbf{W}$ are represented as a length m column vector of numerical query results, which can be computed by multiplying matrix $\mathbf{W}$ by the data vector $\mathbf{x}$.

Note that it is critical that the analyst include in the workload *all* queries of interest. In the absence of noise introduced by the privacy mechanism, it might be reasonable for the analyst to request answers to a small set of counting queries, from which other queries of interest could be computed. (E.g., it would be sufficient to recover $\mathbf{x}$ itself by choosing the workload defined by the identity matrix.) But because the analyst will receive private, noisy estimates to the workload queries, the error of queries computed from their combination is often increased. Our privacy mechanism is designed to optimize error across the entire set of desired queries, so all queries should be included.

As a concrete example, in Table 1(b), $\mathbf{q}_4$ can be computed as $(\mathbf{q}_2 - \mathbf{q}_3)$ but is nevertheless included in the workload. This reflects the fact that we wish to simultaneously answer all included queries with minimum aggregate error, treating each equally. It is also possible to scale individual rows by a positive scalar value, which has the effect of reducing the error of that query.

The cell conditions are used to define the semantics of the queries in a workload, while the workload properties we study are primarily determined by features of their matrix representation. This can lead to a few representational inconsistencies we would like to avoid. First, while a workload $\mathbf{W}$ is meant to represent a *set* of queries, as a matrix it has a specified order of its rows. Further, for any workload $\mathbf{W}$ defined by cell conditions $\Phi = \phi_1 \dots \phi_n$, consider any permutation Φ' of Φ. Then there is a different matrix $\mathbf{W}'$, defined on Φ' and constructed from $\mathbf{W}$ by applying the permutation to its columns, that is semantically equivalent to $\mathbf{W}$. We will verify later that our analysis of workloads is representation independent for both rows and columns. We use the following definition:

DEFINITION 2.7 (REPRESENTATION INDEPENDENCE). *A numerical measure ρ on a workload matrix is* row (resp. column) representation independent *if, given a workload matrix $\mathbf{W}$, and any workload $\mathbf{W}'$ which results from permuting the rows (columns) of $\mathbf{W}$, $\rho(\mathbf{W}) = \rho(\mathbf{W}')$.*

A related issue arises in the specification of the cell conditions. If a workload $\mathbf{W}$ is defined by cell conditions in Φ, then we can always consider extending Φ by adding additional cell conditions not relevant to the queries in $\mathbf{W}$. We can then define a semantically equivalent workload $\mathbf{W}'$ on Φ' which will consist of columns of zeroes for each of the new cell conditions. To address this issue in later sections we rely on the following definition:

DEFINITION 2.8 (COLUMN PROJECTION). *Given an $m \times n$ workload $\mathbf{W}$ defined by cell conditions $\Phi = \phi_1 \dots \phi_n$, and an ordered subset $\Psi \subseteq \Phi$ consisting of p selected cell conditions, the column projection of $\mathbf{W}$ w.r.t Ψ is a new $m \times p$ workload consisting only of the columns of $\mathbf{W}$ included in Ψ.*

In the rest of the paper, μ denotes a subset of cell conditions, $|\mu|$ denotes its cardinality and $\mathcal{U}_n$ to denote all possible subsets of n cell conditions. $\mu(\mathbf{W})$ is the column projection of $\mathbf{W}$ w.r.t. μ.

EXAMPLE 2.9. *The column projection of the workload in Table 1(b) w.r.t. cell conditions $\{\phi_1, \phi_3, \phi_5, \phi_7\}$, which consists of queries only over* Male *students, is the following:*

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

Table 1: For schema $R = (name, gradyear, gender, gpa)$, (a) shows 8 cell conditions on attributes *gradyear* and *gender*. The database vector x (not shown) will accordingly consist of 8 counts; (b) shows a sample workload matrix W consisting of five queries, each described in (c).

(a) Cell conditions Φ

$\phi_1 : gradyear = 2011 \wedge gender = M$
$\phi_2 : gradyear = 2011 \wedge gender = F$
$\phi_3 : gradyear = 2012 \wedge gender = M$
$\phi_4 : gradyear = 2012 \wedge gender = F$
$\phi_5 : gradyear = 2013 \wedge gender = M$
$\phi_6 : gradyear = 2013 \wedge gender = F$
$\phi_7 : gradyear = 2014 \wedge gender = M$
$\phi_8 : gradyear = 2014 \wedge gender = F$

(b) A query matrix $\mathbf{W}$

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

(c) Counting queries defined by rows of $\mathbf{W}$

$\mathbf{q}_1$: all students;
$\mathbf{q}_2$: students with $gradyear \in [2011, 2012]$;
$\mathbf{q}_3$: female students with $gradyear \in [2011, 2012]$;
$\mathbf{q}_4$: male students with $gradyear \in [2011, 2012]$;
$\mathbf{q}_5$: difference between 2013 grads and 2014 grads.

Common workloads. We introduce notation for two common workloads that contain all queries of a certain type. AllRange(d) denotes the workload consisting of all range-count queries over d cell conditions (typically derived from a single ordered attribute A_i with $|dom(A_i)| = d$). There are $\frac{d}{2}(d+1)$ queries in workload AllRange(d). Over k ordered attributes with domain sizes $d_1 \dots d_k$, we similarly define the set of all k-dimensional range-count queries, denoted AllRange($d_1, \dots d_k$).

AllPredicate(d) is the much larger workload consisting of all predicate counting queries over d cell conditions. There are 2^d queries in AllPredicate(d).

Example 2.10. *The workload of queries counting the students who have graduated in any interval of years drawn from* $\{2011, 2012, 2013, 2014\}$ *is denoted* AllRange(4) *and consists of 10 one-dimensional range queries over the* gradyear *attribute.*

Example 2.11. *The set of all two dimensional range-count queries over* gradyear *and* gender *is written* AllRange(4, 2)*, where the possible "ranges" for gender are simply M, F, or $(M \vee F)$. This workload consists of* 30 *queries. The workload of all predicate queries over over* gradyear *and* gender *is* AllPredicate(8)*, consisting of all* 256 *predicate queries over a domain of size 8.*

2.3 Differential privacy & basic mechanisms

Standard ϵ-differential privacy [9] places a bound (controlled by ϵ) on the difference in the probability of query answers for any two *neighboring* databases. For database instance I, we denote by $nbrs(I)$ the set of databases formed by adding or removing exactly one tuple from I. Approximate differential privacy [8, 22], is a relaxation in which the ϵ bound on query answer probabilities may be violated with small probability, controlled by δ.

Definition 2.12 (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 $\mathcal{K}$.

When $\delta = 0$, this definition describes standard ϵ-differential privacy.

Both definitions can be satisfied by adding random noise to query answers. The magnitude of the required noise is determined by the *sensitivity* of a set of queries: the maximum change in a vector of query answers over any two neighboring databases. However, the two privacy definitions differ in the measurement of sensitivity and in their noise distributions. Standard differential privacy can be achieved by adding Laplace noise calibrated to the L_1 sensitivity of the queries [9]. Approximate differential privacy can be achieved by adding Gaussian noise calibrated to the L_2 sensitivity of the queries [8, 22]. This small difference in the sensitivity metric—from L_1 to L_2—has important consequences for the theory underlying our analysis and, unless otherwise noted, *stated results apply only to approximate differential privacy.* Sec 4.3 contains a comparison of these two definitions as they pertain to the matrix mechanism and the results of this paper.

Since our query workloads are represented as matrices, we express the sensitivity of a workload as a matrix norm. Notice that for neighboring databases I and I', $|(I - I') \cap (I' - I)| = 1$ and recall that all cell conditions are mutually unsatisfiable. It follows that the corresponding data vectors $\mathbf{x}$ and $\mathbf{x}'$ differ in exactly one component, by exactly one. We extend our notation and write $\mathbf{x}' \in nbrs(\mathbf{x})$. The L_2 sensitivity of $\mathbf{W}$ is equal to the maximum L_2 norm of the columns of $\mathbf{W}$. Below, cols($\mathbf{W}$) is the set of column vectors W_i of $\mathbf{W}$.

Definition 2.13 (L_2 Query matrix sensitivity). *The L_2 sensitivity of a query matrix $\mathbf{W}$ is denoted $\Delta_{\mathbf{W}}$ and defined as follows:*

$$\Delta_{\mathbf{W}} \stackrel{\text{def}}{=} \max_{\mathbf{x}' \in nbrs(\mathbf{x})} ||\mathbf{W}\mathbf{x} - \mathbf{W}\mathbf{x}'||_2 = \max_{W_i \in cols(\mathbf{W})} ||W_i||_2$$

The classic differentially private mechanism adds independent noise calibrated to the sensitivity of a query workload. We use Normal$(\sigma)^m$ to denote a column vector consisting of m independent samples drawn from a Gaussian distribution with mean 0 and scale σ.

Proposition 2.14. (Gaussian mechanism [8, 22]) *Given an $m \times n$ query matrix $\mathbf{W}$, the randomized algorithm $\mathcal{G}$ that outputs the following vector is (ϵ, δ)-differentially private:*

$$\mathcal{G}(\mathbf{W}, \mathbf{x}) = \mathbf{W}\mathbf{x} + Normal(\sigma)^m$$

where $\sigma = \Delta_{\mathbf{W}} \sqrt{2 \ln(2/\delta)}/\epsilon$

Recall that $\mathbf{W}\mathbf{x}$ is a vector of the true answers to each query in $\mathbf{W}$. The algorithm above adds independent Gaussian noise (scaled by the sensitivity of $\mathbf{W}$, ϵ, and δ) to each query answer. Thus $\mathcal{G}(\mathbf{W}, \mathbf{x})$ is a length-m column vector containing a noisy answer for each linear query in $\mathbf{W}$.

2.4 Linear algebra notation

Throughout the paper, we use the notation of linear algebra and employ standard techniques of matrix analysis. Recall that for a matrix $\mathbf{A}$, $\mathbf{A}^T$ is its transpose, $\mathbf{A}^{-1}$ is its

inverse, and trace($\mathbf{A}$) is the sum of values on the main diagonal. The Frobenius norm of $\mathbf{A}$ is denoted $||\mathbf{A}||_F$ and defined as the square root of the squared sum of all entries in $\mathbf{A}$, or, equivalently, $\sqrt{\text{trace}(\mathbf{A}^T\mathbf{A})}$. We use $diag(c_1, \ldots c_n)$ 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. An orthogonal matrix $\mathbf{Q}$ is a square matrix whose rows and columns are orthogonal unit vectors, and for which $\mathbf{Q}^T = \mathbf{Q}^{-1}$.

We will also rely on the notion of a positive semidefinite matrix. A symmetric square matrix $\mathbf{A}$ is called positive semidefinite if for any vector $\mathbf{x}$, $\mathbf{x}^T\mathbf{A}\mathbf{x} \geq 0$. If $\mathbf{A}$ is positive semidefinite, denoted $\mathbf{A} \succeq 0$, all its diagonal entries are non-negative as well. In particular, for any matrix $\mathbf{A}$, $\mathbf{A}^T\mathbf{A}$ is a positive semidefinite matrix.

In addition, we use $\mathbf{A}^+$ to represent the Moore-Penrose pseudoinverse of a matrix $\mathbf{A}$, a generalization of the matrix inverse defined as follows:

DEFINITION 2.15. (MOORE-PENROSE PSEUDOINVERSE [2]) *Given a $m \times n$ matrix $\mathbf{A}$, a matrix $\mathbf{A}^+$ is the Moore-Penrose pseudoinverse of $\mathbf{A}$ if it satisfies each of the following:*

$$\mathbf{A}\mathbf{A}^+\mathbf{A} = \mathbf{A}, \qquad \mathbf{A}^+\mathbf{A}\mathbf{A}^+ = \mathbf{A}^+,$$
$$(\mathbf{A}\mathbf{A}^+)^T = \mathbf{A}\mathbf{A}^+, \qquad (\mathbf{A}^+\mathbf{A})^T = \mathbf{A}^+\mathbf{A}.$$

We include some important properties of the Moore-Penrose pseudoinverse in the following theorem.

THEOREM 2.16. ([2]) *The Moore-Penrose pseudoinverse satisfies the following properties:*

1. *Given any matrix $\mathbf{A}$, there exists a unique matrix that is the Moore-Penrose pseudoinverse of $\mathbf{A}$.*
2. *Given a vector $\mathbf{y}$, we have $||\mathbf{y} - \mathbf{A}\mathbf{x}||_2 \geq ||\mathbf{y} - \mathbf{A}\mathbf{A}^+\mathbf{y}||_2$ for any vector $\mathbf{x}$.*
3. *For any satisfiable linear system $\mathbf{B}\mathbf{A} = \mathbf{W}$, $\mathbf{W}\mathbf{A}^+$ is a solution to the linear system and $||\mathbf{W}\mathbf{A}^+||_F \leq ||\mathbf{B}||_F$ for any solution $\mathbf{B}$ to the linear system.*

Throughout the paper, we use the singular value decomposition of a matrix, which is a classic tool of matrix analysis. If $\mathbf{W}$ is an $m \times n$ matrix, the singular value decomposition (SVD) of $\mathbf{W}$ is a factorization of the form $\mathbf{W} = \mathbf{Q}_\mathbf{W}\mathbf{\Lambda}_\mathbf{W}\mathbf{P}_\mathbf{W}^T$ such that $\mathbf{Q}_\mathbf{W}$ is an $m\times m$ orthogonal matrix, $\mathbf{\Lambda}_\mathbf{W}$ is a $m\times n$ diagonal matrix containing the singular values of $\mathbf{W}$ and $\mathbf{P}_\mathbf{W}$ is an $n \times n$ orthogonal matrix. When $m > n$, the diagonal matrix $\mathbf{\Lambda}_\mathbf{W}$ consists of an $n \times n$ diagonal submatrix combined with $\mathbf{0}^{(m-n)\times n}$. In addition, we also consider the eigenvalue decomposition of matrix $\mathbf{W}^T\mathbf{W}$ and the square root of $\mathbf{W}^T\mathbf{W}$. The eigenvalue decomposition of $\mathbf{W}^T\mathbf{W}$ has the form $\mathbf{W}^T\mathbf{W} = \mathbf{P}_\mathbf{W}\mathbf{D}_\mathbf{W}\mathbf{P}_\mathbf{W}^T$, where $\mathbf{P}_\mathbf{W}$ is the same matrix as the singular value decomposition of $\mathbf{W}$ and $\mathbf{D}'_\mathbf{W}$ is an $n\times n$ diagonal matrix such that $\mathbf{D}_\mathbf{W} = \mathbf{\Lambda}_\mathbf{W}^T\mathbf{\Lambda}_\mathbf{W}$. The square root of $\mathbf{W}^T\mathbf{W}$, denoted as $\sqrt{\mathbf{W}^T\mathbf{W}}$, is a matrix $\mathbf{W}'$ such that $(\mathbf{W}')^2 = \mathbf{W}^T\mathbf{W}$, which can also be represented as the singular values and singular vectors of $\mathbf{W}$: $\mathbf{W}' = \mathbf{P}_\mathbf{W}\mathbf{\Lambda}_\mathbf{W}\mathbf{P}_\mathbf{W}^T$.

3. THE (ϵ, δ)-MATRIX MECHANISM

In this section we define the class of algorithms that can be constructed using the matrix mechanism, we define optimal error of a workload with respect to this class, and we develop notions of workload equivalence and containment consistent with our error measures.

3.1 The extended matrix mechanism

The matrix mechanism [18] has a form similar to the Gaussian mechanism in Prop. 2.14, but adds a more complex noise vector. It uses a different set of queries (the strategy matrix $\mathbf{A}$) to construct this vector. The intuitive justification for this mechanism is that it is equivalent to the following three-step process: (1) the queries in the strategy are submitted to the Gaussian mechanism; (2) an estimate $\hat{\mathbf{x}}$ for $\mathbf{x}$ is derived by computing the $\hat{\mathbf{x}}$ that minimizes the squared sum of errors (this step consists of standard linear regression and requires that $\mathbf{A}$ be full rank to ensure a unique solution); (3) noisy answers to the workload queries are then computed as $\mathbf{W}\hat{\mathbf{x}}$.

We present an extended version of the matrix mechanism which relaxes the requirement that $\mathbf{A}$ be full rank. Instead it is sufficient that all queries in $\mathbf{W}$ can be represented as linear combinations of queries in $\mathbf{A}$. Then estimating $\hat{\mathbf{x}}$ is not necessary, and step (2) and (3) can be combined.

PROPOSITION 3.1. (EXTENDED (ϵ,δ)-MATRIX MECHANISM) *Given an $m \times n$ query matrix $\mathbf{W}$, and a $p \times n$ strategy matrix $\mathbf{A}$ such that $\mathbf{W}\mathbf{A}^+\mathbf{A} = \mathbf{W}$, the following randomized algorithm $\mathcal{M}_\mathbf{A}$ is (ϵ, δ)-differentially private:*

$$\mathcal{M}_\mathbf{A}(\mathbf{W}, \mathbf{x}) \quad = \quad \mathbf{W}\mathbf{x} + \mathbf{W}\mathbf{A}^+ \mathit{Normal}(\sigma)^m.$$

where $\sigma = \Delta_\mathbf{A}\sqrt{2\ln(2/\delta)}/\epsilon$.

Here condition $\mathbf{W}\mathbf{A}^+\mathbf{A} = \mathbf{W}$ guarantees that the queries in $\mathbf{A}$ can represent all queries in $\mathbf{W}$. When $\mathbf{A}$ has full rank, $\mathbf{A}^+ = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T$, which coincides with the original definition [18].

Like the Gaussian mechanism, the matrix mechanism computes the true answer vector $\mathbf{W}\mathbf{x}$ and adds noise to each component. But a key difference is that the scale of the Gaussian noise is *calibrated to the sensitivity of the strategy matrix $\mathbf{A}$, not that of the workload.* In addition, the noise added to the list of query answers is no longer independent, because the vector of independent Gaussian samples is transformed by the matrix $\mathbf{W}\mathbf{A}^+$.

The matrix mechanism can reduce error, particularly for large or complex workloads, by avoiding redundancy in the set of desired workload queries. Intuitively, some workloads consist of queries that ask the same (or similar) questions of the database multiple times, which incurs a significant cost to the privacy budget under the Gaussian mechanism. By choosing the right strategy matrix for a workload, it is possible to remove the redundancy from the queries submitted to the privacy mechanism and derive more accurate answers to the workload queries.

Fundamental to the performance of the matrix mechanism, is the choice of the strategy matrix which instantiates it. One naive approach is to minimize sensitivity. The full rank strategy matrix with least sensitivity is the identity matrix, $\mathbf{I}$, which has sensitivity 1. With $\mathbf{A} = \mathbf{I}$, the matrix mechanism privately computes the individual counts in $\mathbf{x}$ and then uses them to estimate any desired workload query. At the other extreme, the workload itself can be used as the strategy, setting $\mathbf{A} = \mathbf{W}$. In this case, there is no benefit in sensitivity over the Gaussian mechanism[1].

For many workloads, neither of these basic strategies offer optimal error. Recent research has shown that for specific

[1]Although there is no benefit in sensitivity when $\mathbf{A} = \mathbf{W}$, the matrix mechanism still has lower error than the Gaussian mechanism for some workloads by combining related query answers into a more accurate consistent result.

workloads, there exist strategies that can offer much better error rates. For example, if $\mathbf{W} = \text{AllRange}(n)$, two strategies were recently proposed. A *hierarchical* strategy [16] includes the total sum over the whole domain, the count of each half of the domain, and so on, terminating with counts of individual elements of the domain. The *wavelet* strategy [27] consists of the matrix describing the Haar wavelet transformation. Informally, both strategies achieve low error because they each have low sensitivity, $O(\log n)$, and every range query can be expressed as a linear combination of just a few strategy queries.[2] Although both of these strategy matrices offer significant improvements in error for range query workloads, neither is optimal. We will use our lower bound to evaluate the quality of these proposed strategies in Sec 4. Other recently-proposed techniques can be seen as attempts to compute approximately optimal strategy matrices adapted to the input workload [1, 6, 19, 30, 29].

3.2 Measuring & minimizing error

We measure the error of individual query answers using mean squared error. For a workload of queries, the error is defined as the *total* of individual query errors.

Definition 3.2 (Query and Workload Error [18]). *Let $\hat{\mathbf{w}}$ be the estimate for query $\mathbf{w}$ under the matrix mechanism using query strategy $\mathbf{A}$. That is, $\hat{\mathbf{w}} = \mathcal{M}_{\mathbf{A}}(\mathbf{w}, \mathbf{x})$. The mean squared error of the estimate for $\mathbf{w}$ using strategy $\mathbf{A}$ is:*

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

Given a workload $\mathbf{W}$, the total mean squared error of answering $\mathbf{W}$ using strategy $\mathbf{A}$ is: $\text{Error}_{\mathbf{A}}(\mathbf{W}) = \sum_{\mathbf{w}_i \in \mathbf{W}} \text{Error}_{\mathbf{A}}(\mathbf{w}_i)$.

The query answers returned by the matrix mechanism are linear combinations of noisy strategy query answers to which independent Gaussian noise has been added. Thus, as the following proposition shows (extending the corresponding proposition from [18]), we can directly compute the error for any linear query $\mathbf{w}$ or workload of queries $\mathbf{W}$:

Proposition 3.3 (Total Error). *Given a workload $\mathbf{W}$, the total error of answering $\mathbf{W}$ using the extended (ϵ, δ)-matrix mechanism with query strategy $\mathbf{A}$ is:*

$$\text{Error}_{\mathbf{A}}(\mathbf{W}) = P(\epsilon, \delta)\Delta_{\mathbf{A}}^2 \, ||\mathbf{W}\mathbf{A}^+||_F^2 \qquad (1)$$

where $P(\epsilon, \delta) = \frac{2\log(2/\delta)}{\epsilon^2}$.

We call a mechanism *data-independent* if its error for workload $\mathbf{W}$ is independent of the data vector $\mathbf{x}$. Proposition 3.3 shows that the matrix mechanism is data-independent.

The optimal strategy for a workload $\mathbf{W}$ is defined to be the one that minimizes total error (the same measure used in [18]).

Definition 3.4 (Minimum Total Error [18]). *Given a workload $\mathbf{W}$, the minimum total error is:*

$$\text{MinError}(\mathbf{W}) = \min_{\mathbf{A}:\, \mathbf{W}\mathbf{A}^+\mathbf{A} = \mathbf{W}} \text{Error}_{\mathbf{A}}(\mathbf{W}). \qquad (2)$$

Our work investigates the error complexity of workloads as it is represented by $\text{MinError}(\mathbf{W})$. This reflects the hardness of the workload assuming we use an algorithm that is an instance of the matrix mechanism and that a total squared error measure is used. Understanding error for this class of mechanisms is a first step towards more general lower bounds and helps to assess the quality of a number of previously-proposed algorithms included in this class [18, 27, 1, 6, 19, 30, 29].

The strategy matrix that minimizes total error can be computed using a semi-definite program (SDP) [18]. However, finding the solutions of the program with standard SDP solvers takes $O(n^8)$ time, where n is the number of cell conditions, making it infeasible for realistic applications. Efficient approximation algorithms for this problem have been investigated recently [19, 30]. Yet, approximate—or even exact—solutions to this problem do not provide much general insight into the main goal of this paper: to understand the properties of workloads that determine the magnitude of $\text{MinError}(\mathbf{W})$. The bound we will present in Sec. 4 is important because it closely approximates $\text{MinError}(\mathbf{W})$, it is easily computable, and it reveals the connection between minimum error and spectral properties.

Nevertheless, our results do not preclude the existence of different mechanisms capable of answering a workload $\mathbf{W}$ with lower error. In particular, our error analysis does not include data-dependent algorithms [26, 14, 24, 5]. The error rates for these mechanisms no longer reflect properties of the workload alone, but instead some combination of the workload and properties of the input data, and call for a substantially different analysis. In our case, $\mathbf{x}$ (i.e., the vector of cell counts corresponding to the database) does not appear in (1) above. This means that our minimum error strategy depends on the workload alone, independent of a particular database instance.

3.3 Equivalence & containment for workloads

Next we develop a notion of equivalence and containment of workloads with respect to error. We will verify that the error bounds presented in the next section satisfy these relationships in most cases.

The special form of the expression for total error in Prop. 3.3 means that there are many workloads that are equivalent from the standpoint of error. For two workloads $\mathbf{W}_1$ and $\mathbf{W}_2$, if $\mathbf{W}_1^T\mathbf{W}_1 = \mathbf{W}_2^T\mathbf{W}_2$, then any strategy $\mathbf{A}$ that can represent the queries of $\mathbf{W}_1$ can also represent the queries of $\mathbf{W}_2$, and vice versa. In addition, $\mathbf{W}_1^T\mathbf{W}_1 = \mathbf{W}_2^T\mathbf{W}_2$ implies $||\mathbf{W}_1\mathbf{A}^+||_F^2 = ||\mathbf{W}_2\mathbf{A}^+||_F^2$ for any strategy $\mathbf{A}$. We therefore define the following notion of *equivalence* of two workloads:

Definition 3.5 (Workload Equivalence). *An $m_1 \times n_1$ workload $\mathbf{W}_1$ and an $m_2 \times n$ workload $\mathbf{W}_2$ are* equivalent, *denoted $\mathbf{W}_1 \equiv \mathbf{W}_2$, if $\mathbf{W}_1^T\mathbf{W}_1 = \mathbf{W}_2^T\mathbf{W}_2$.*

The following conditions on pairs of workloads imply that they have equivalent minimum error:

Proposition 3.6 (Equivalence Conditions). *Given an $m_1 \times n_1$ workload $\mathbf{W}_1$ and an $m_2 \times n_2$ workload $\mathbf{W}_2$, each of the following conditions implies that* $\text{MinError}(\mathbf{W}_1) = \text{MinError}(\mathbf{W}_2)$*:*

(i) $\mathbf{W}_1 \equiv \mathbf{W}_2$

(ii) $\mathbf{W}_1 = \mathbf{Q}\mathbf{W}_2$ *for some orthogonal matrix* $\mathbf{Q}$.

(iii) $\mathbf{W}_2$ *results from permuting the rows of* $\mathbf{W}_1$.

(iv) $\mathbf{W}_2$ *results from permuting the columns of* $\mathbf{W}_1$.

[2] The approaches in [16, 27] were originally proposed in the context of ϵ-differential privacy, but their behavior is similar under (ϵ, δ)-differential privacy.

(v) $\mathbf{W}_2$ *results from the column projection of* $\mathbf{W}_1$ *on all of its nonzero columns.*

It follows from this proposition that MinError is row and column representation independent, and behaves well under the projection of extraneous columns.

Defining a notion of containment for workload matrices is more complex than simple inclusion of rows. Even if the rows of $\mathbf{W}_1$ are not present in $\mathbf{W}_2$, it could be that $\mathbf{W}_1$ is in fact contained in $\mathbf{W}_2$ when expressed using an alternate basis. The following definition considers this possibility:

Definition 3.7 (Workload Containment). *An* $m_1 \times n$ *workload* $\mathbf{W}_1$ *is contained in an* $m_2 \times n$ *workload* $\mathbf{W}_2$*, denoted* $\mathbf{W}_1 \subseteq \mathbf{W}_2$*, if there exists a* $\mathbf{W}_2' \equiv \mathbf{W}_2$ *and the rows of* $\mathbf{W}_1$ *are contained in* $\mathbf{W}_2'$*.*

The following proposition shows two conditions which imply inequality of error among workloads:

Proposition 3.8 (Error inequality). *Given an* $m_1 \times n_1$ *workload* $\mathbf{W}_1$ *and an* $m_2 \times n_2$ *workload* $\mathbf{W}_2$*, each of the following conditions implies that*

$$\textsc{MinError}(\mathbf{W}_1) \leq \textsc{MinError}(\mathbf{W}_2) :$$

(i) $\mathbf{W}_1 \subseteq \mathbf{W}_2$

(ii) $\mathbf{W}_1$ *is a column projection of* $\mathbf{W}_2$*.*

4. THE SINGULAR VALUE BOUND

In this section we state and prove our main result: a lower bound on MinError($\mathbf{W}$), the optimal error of a workload $\mathbf{W}$ under the extended (ϵ, δ)-matrix mechanism. The bound shows that the hardness of a workload is a function of its eigenvalues. We describe the measure and its properties in Section 4.1 and prove that it is a lower bound in Section 4.2. In Section 4.3 we briefly discuss the challenge of adapting this bound to ϵ-differential privacy.

4.1 The Singular Value Bound

We first present the simplest form of our bound, which is based on computing the square of the sum of eigenvalues of the workload matrix:

Definition 4.1 (Singular Value Bound). *Given an* $m \times n$ *workload* $\mathbf{W}$*, its singular value bound, denoted* svdb($\mathbf{W}$)*, is:*

$$\textsc{svdb}(\mathbf{W}) = \frac{1}{n}(\lambda_1 + \ldots + \lambda_n)^2,$$

where $\lambda_1, \ldots, \lambda_n$ *are the singular values of* $\mathbf{W}$*.*

The following theorem guarantees that the singular value bound is a valid lower bound to the minimal error of a workload. The proof is presented in detail in Sec. 4.2.

Theorem 4.2. *Given an* $m \times n$ *workload* $\mathbf{W}$*,*

$$\textsc{MinError}(\mathbf{W}) \geq P(\epsilon, \delta)\textsc{svdb}(\mathbf{W}),$$

where $P(\epsilon, \delta) = \frac{2\log(2/\delta)}{\epsilon^2}$*.*

In the rest of paper, we refer to svdb($\mathbf{W}$) as the "SVD bound". For any workload $\mathbf{W}$, the SVD bound is determined by $\mathbf{W}^T\mathbf{W}$ and can be computed directly from it (which can be more efficient):

Proposition 4.3. *Given* $n \times n$ *matrix* $\mathbf{W}^T\mathbf{W}$*.*

$$\textsc{svdb}(\mathbf{W}) = \frac{1}{n}(\sum_{i=1}^{n} \sqrt{d_i})^2.$$

where $d_1, \ldots, d_n$ *are the eigenvalues of* $\mathbf{W}^T\mathbf{W}$*.*

The SVD bound satisfies equivalence properties analogous to (i), (ii), (iii), and (iv) in Prop. 3.6 and inequality (i) in Prop. 3.8. However, it does not satisfy properties related to column projection, as shown in the following counter-example.

Example 4.4. *Consider a* $2 \times n$ *workload* $\mathbf{W}$ *consisting of queries* $[1, 0, \ldots, 0]$ *and* $[t, t, \ldots, t]$*. Let* μ *be the column projection w.r.t. the first cell condition of* $\mathbf{W}$*. When* $n > 8$ *and* $t < 1/8$*,* svdb($\mathbf{W}$) < svdb($\mu(\mathbf{W})$)*.*

According to Prop 3.8, column projections reduce the minimum error. Therefore, the SVD bound on any column projection of $\mathbf{W}$ also constitutes a lower bound for the minimum error of $\mathbf{W}$. Because of this we extend the simple SVD bound in the following way. Recall that $\mathcal{U}_n$ is the set of all column projections.

Definition 4.5. *Given an* $m \times n$ *workload* $\mathbf{W}$ *and* $U \subseteq \mathcal{U}_n$*. The singular value bound of* $\mathbf{W}$ *w.r.t.* U*, denoted by* $\textsc{svdb}_U(\mathbf{W})$ *is defined as*

$$\textsc{svdb}_U(\mathbf{W}) = \max_{\mu \in U} \textsc{svdb}(\mu(\mathbf{W})).$$

In particular, if $U = \mathcal{U}_n$*, we call this bound the* supreme singular value bound*, denoted* $\overline{\textsc{svdb}}(\mathbf{W})$*.*

According to Prop. 3.8 and Thm. 4.2, for any $U \subseteq \mathcal{U}_n$, $\textsc{svdb}_U(\mathbf{W})$ provides a lower bound on MinError($\mathbf{W}$).

Corollary 4.6. *Given an* $m \times n$ *workload* $\mathbf{W}$*, and for any* $U \subseteq \mathcal{U}_n$

$$\begin{aligned}\textsc{MinError}(\mathbf{W}) &\geq \max_{\mu \in U} \textsc{MinError}(\mu(\mathbf{W})) \\ &\geq P(\epsilon, \delta)\textsc{svdb}_U(\mathbf{W}),\end{aligned}$$

where $P(\epsilon, \delta) = \frac{2\log(2/\delta)}{\epsilon^2}$*.*

The supreme SVD bound satisfies all of the error equivalence and containment properties, analogous to those of Prop. 3.6 and Prop. 3.8, as stated below:

Theorem 4.7. *Given an* $m_1 \times n_1$ *workload* $\mathbf{W}_1$ *and an* $m_2 \times n_2$ *workload* $\mathbf{W}_2$*, the following conditions imply that* $\overline{\textsc{svdb}}(\mathbf{W}_1) = \overline{\textsc{svdb}}(\mathbf{W}_2)$*:*

(i) $\mathbf{W}_1 \equiv \mathbf{W}_2$

(ii) $\mathbf{W}_1 = \mathbf{Q}\mathbf{W}_2$ *for some orthogonal matrix* $\mathbf{Q}$*.*

(iii) $\mathbf{W}_2$ *results from permuting the rows of* $\mathbf{W}_1$*.*

(iv) $\mathbf{W}_2$ *results from permuting the columns of* $\mathbf{W}_1$*.*

(v) $\mathbf{W}_2$ *results from column projection of* $\mathbf{W}_1$ *on all of its nonzero columns.*

Theorem 4.8. *Given an* $m_1 \times n_1$ *workload* $\mathbf{W}_1$ *and an* $m_2 \times n_2$ *workload* $\mathbf{W}_2$*, the following conditions imply that* $\overline{\textsc{svdb}}(\mathbf{W}_1) \leq \overline{\textsc{svdb}}(\mathbf{W}_2)$*:*

(i) $\mathbf{W}_1 \subseteq \mathbf{W}_2$

(ii) $\mathbf{W}_1$ *is a column projection of* $\mathbf{W}_2$*.*

While Theorems 4.7 and 4.8 show that $\overline{\text{SVDB}}(\mathbf{W})$ matches all the properties of MinError($\mathbf{W}$), we often wish to avoid considering all possible column projections as required in the computation of $\overline{\text{SVDB}}(\mathbf{W})$. In many cases, using SVDB($\mathbf{W}$) as our lower bound provides good results. In other cases, we can choose an appropriate set of column projections to get a good approximation to the supreme SVD bound. We provide empirical evidence for this in the following example, along with an application of our bound to range and predicate workloads which have been studied in prior work. The bound allows us to evaluate, for the first time, how well existing solutions approximate the minimum achievable error under (ϵ, δ)-differential privacy.

Example 4.9. *In Table 2 we consider three workloads, each consisting of all multi-dimensional range queries for a different dimension set, along with a workload of all predicate queries. We report* SVDB($\mathbf{W}$) *and its ratio with* SVDB$_U$($\mathbf{W}$) *where U contains projections onto all possible ranges over the domain, showing that they are virtually indistinguishable.*

We also compute the actual error introduced by several well-known strategies: the identity strategy, the hierarchical strategy [16], and the wavelet strategy [27], as well as a strategy generated by the Eigen-design mechanism [19]. These results reveal the quality of these approaches by their ratio to SVDB($\mathbf{W}$). *For example, from the table we can conclude that the Eigen-design mechanism and wavelet strategies have error at most* 1.5 *to* 3 *times the optimal for range workloads, but perform worse on the predicate queries. The identity strategy is far from optimal on low dimensional range queries, but better on high dimensional range queries and predicate queries.*

4.2 Proof of the SVD bound

We now describe the proof of Theorem 4.2. The key to the proof is an important property of the optimal strategy for the (ϵ, δ) matrix mechanism. As shown in Lemma 4.10, among the optimal strategies for a workload $\mathbf{W}$, there is always a strategy $\mathbf{A}$ that has the same sensitivity for every cell condition (i.e. in every column). We use $\mathcal{A}_{\mathbf{W}}$ to denote the set that contains all strategies that satisfy $\mathbf{W}\mathbf{A}^+\mathbf{A} = \mathbf{W}$ and have the same sensitivity for every cell condition.

Recall that the sensitivity of strategy $\mathbf{A}$ (Def. 2.13) is the maximum L_2 column norm of $\mathbf{A}$. The square of the sensitivity is also equal to the maximum diagonal entry of $\mathbf{A}^T\mathbf{A}$. By using Lemma 4.10, the sensitivity of $\mathbf{A}$ can instead be computed in terms of the trace of the matrix $\mathbf{A}^T\mathbf{A}$ and minimizing the error of $\mathbf{W}$ with this alternative expression of the sensitivity leads to the SVD bounds. Ultimately, to achieve the SVD bounds, a strategy $\mathbf{A}$ must simultaneously (i) minimize the error of $\mathbf{W}$ with the sensitivity computed in terms of the trace($\mathbf{A}^T\mathbf{A}$), and (ii) have $\mathbf{A} \in \mathcal{A}_{\mathbf{W}}$. Such a strategy may not exist for every possible $\mathbf{W}$ and therefore the SVD bounds only serve as lower bounds to the minimal error of $\mathbf{W}$.

Lemma 4.10. *Given a workload $\mathbf{W}$, there exists a strategy $\mathbf{A} \in \mathcal{A}_{\mathbf{W}}$ such that* Error$_{\mathbf{A}}$($\mathbf{W}$) = MinError($\mathbf{W}$).

Lemma 4.11. *Let $\mathbf{D}$ be a diagonal matrix with non-negative diagonal entries and $\mathbf{P}$ be an orthogonal matrix whose column equals to $\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_n$.*

$$\textit{trace}(\mathbf{D}) \leq \sum_{i=1}^{n} ||\mathbf{D}\mathbf{p}_i||_2.$$

Theorem 4.2 can hence be proved using the lemmas above.

Proof. For a given workload $\mathbf{W}$, according to Lemma 4.10, it has an optimal strategy matrix $\mathbf{A} \in \mathcal{A}_{\mathbf{W}}$, whose sensitivity can then be computed as $\Delta^2_{\mathbf{A}} = \frac{1}{n}||\mathbf{A}||^2_F$.

Let $\mathbf{W} = \mathbf{Q}_{\mathbf{W}}\mathbf{\Lambda}_{\mathbf{W}}\mathbf{P}_{\mathbf{W}}$ and $\mathbf{A} = \mathbf{Q}_{\mathbf{A}}\mathbf{\Lambda}_{\mathbf{A}}\mathbf{P}_{\mathbf{A}}$ be the singular decomposition of $\mathbf{W}$ and $\mathbf{A}$, respectively. We have:

$$\begin{aligned}
&\min_{\mathbf{A}:\,\mathbf{W}\mathbf{A}^+\mathbf{A}=\mathbf{W}} \Delta^2_{\mathbf{A}}||\mathbf{W}\mathbf{A}^+||^2_F \\
=\;& \min_{\mathbf{A}\in\mathcal{A}_{\mathbf{W}}} \frac{1}{n}||\mathbf{A}||^2_F||\mathbf{W}\mathbf{A}^+||^2_F \\
=\;& \frac{1}{n}\min_{(\mathbf{\Lambda}_{\mathbf{A}}\mathbf{P}_{\mathbf{A}})\in\mathcal{A}_{\mathbf{W}}} ||\mathbf{\Lambda}_{\mathbf{A}}||^2_F||\mathbf{\Lambda}_{\mathbf{W}}\mathbf{P}_{\mathbf{W}}\mathbf{P}^T_{\mathbf{A}}\mathbf{\Lambda}^+_{\mathbf{A}}||^2_F \\
\geq\;& \frac{1}{n}\min_{\substack{\mathbf{\Lambda}_{\mathbf{A}},\mathbf{P}_{\mathbf{A}} \\ \mathbf{\Lambda}_{\mathbf{W}}\mathbf{\Lambda}^+_{\mathbf{A}}\mathbf{\Lambda}^=_{\mathbf{A}}\mathbf{\Lambda}_{\mathbf{W}}}} ||\mathbf{\Lambda}_{\mathbf{A}}||^2_F||\mathbf{\Lambda}_{\mathbf{W}}\mathbf{P}_{\mathbf{W}}\mathbf{P}^T_{\mathbf{A}}\mathbf{\Lambda}^+_{\mathbf{A}}||^2_F \qquad (3) \\
\geq\;& \frac{1}{n}\min_{\mathbf{P}_{\mathbf{A}}}(\sum_{i=1}^{n}||\mathbf{\Lambda}_{\mathbf{W}}\mathbf{p}_i||_2)^2 \qquad (4) \\
\geq\;& \frac{1}{n}(\sum_{i=1}^{n}\lambda_i)^2, \qquad (5)
\end{aligned}$$

where $\mathbf{p}_i$ is the i-th column of matrix $\mathbf{P}_{\mathbf{W}}\mathbf{P}^T_{\mathbf{A}}$, the inequality in (4) is based on the Cauchy-Schwarz inequality and the inequality in (5) comes from Lemma 4.11.

The equal sign in (4) is satisfied if and only if $\mathbf{\Lambda}_{\mathbf{A}} \propto \sqrt{\mathbf{\Lambda}_{\mathbf{W}}}$. Therefore to achieve equality in (4) and (5) simultaneously, we need $\mathbf{A} \propto \mathbf{Q}\sqrt{\mathbf{\Lambda}_{\mathbf{W}}}\mathbf{P}_{\mathbf{W}}$ for any orthogonal matrix $\mathbf{Q}$. Moreover, (3) is true if and only if $\mathbf{A} \in \mathcal{A}_{\mathbf{W}}$, which may not be satisfied when $\mathbf{A} \propto \mathbf{Q}\sqrt{\mathbf{\Lambda}_{\mathbf{W}}}\mathbf{P}_{\mathbf{W}}$, therefore the SVD bound only gives an lower bound to the minimum total error. □

Intuitively, the SVD bound is based on the assumption that the error can be evenly distributed to all the cells, which may not be achievable in all the cases. The supreme SVD bound considers only the case that the error can be evenly distributed to some of the cells and therefore may be tighter than the SVD bound.

4.3 Bounding MinError($\mathbf{W}$) Under the ϵ-Matrix Mechanism

The SVD bound is defined for the (ϵ, δ)-matrix mechanism, so it is natural to consider extending these results to the ϵ-matrix mechanism. Prop. 3.3 can be adopted to the ϵ-matrix mechanism, which uses Laplace noise, an alternative privacy parameter, $P(\epsilon) = 1/\epsilon^2$, and measures the sensitivity of $\mathbf{A}$ as the largest L_1 norm of the columns of $\mathbf{A}$. For any vector, its L_1 norm is always greater than or equal to its L_2 norm. Given a workload $\mathbf{W}$ and a strategy matrix $\mathbf{A}$, $P(\epsilon)\Delta^2_{\mathbf{A}}||\mathbf{W}\mathbf{A}^+||^2_F$ provides a lower bound to Error$_{\mathbf{A}}$($\mathbf{W}$) under the ϵ-matrix mechanism. Therefore, error under the ϵ-matrix mechanism is also bounded below by SVDB($\mathbf{W}$).

When the number of queries in a workload is no more than the domain size, Bhaskara et al. [3] presented the following lower bound of error for any data-independent ϵ-differential privacy mechanism.

Theorem 4.12 ([3]). *Given an $m \times n$ workload $\mathbf{W}$ with $m \leq n$, let convex body $\mathbf{K} = \mathbf{W}\mathbf{B}^n_1$, where $\mathbf{B}^m_1$ is the m-dimensional L_1 ball. Let $\mathbf{P}_1, \ldots, \mathbf{P}_t$ be projection operators to a collection of t mutually orthogonal subspaces of $\mathbb{R}^m$ of dimension $m_1, \ldots, m_t$ respectively. Then the error of an-*

Example Workload, $\mathbf{W}$	$\text{SVDB}(\mathbf{W})$	$\text{SVDB}_U(\mathbf{W})$	Error, as ratio to $P(\epsilon,\delta)\text{SVDB}(\mathbf{W})$			
			Identity	Hierarchical	Wavelet	Eigen Design
AllRange(2048)	3.034×10^7	1.001	47.25	1.776	1.545	1.028
AllRange(64, 32)	2.261×10^7	1.000	12.11	2.996	1.899	1.107
AllRange(2, 2, 2, 2, 2, 2, 2, 2, 2, 2)	5.242×10^5	1.000	2.000	2.000	2.000	1.000
AllPredicate(1024)	4.885×10^{156}	1.000	1.884	3.464	6.292	1.000

Table 2: Four example workloads, their singular value bounds, and their error rates under common strategies and strategies proposed in prior work.

swering $\mathbf{W}$ *under any data-independent* ϵ*-differentially private mechanism must be at least*

$$\Omega\left(\sum_i \frac{m_i^3}{\epsilon^2} Vol_{m_i}(\mathbf{P}_i\mathbf{K})^{2/m_i}\right),$$

where $Vol_{m_i}(\mathbf{P}_i\mathbf{K})$ *is the volume of the convex body* $\mathbf{P}_i\mathbf{K}$ *in* m_i *dimensional space.*

In particular, when $\mathbf{P}_i$ are the projections to the singular vectors of $\mathbf{W}$, we can formulate the bound above using singular values of $\mathbf{W}$.

Corollary 4.13. *Given an* $m \times n$ *workload* $\mathbf{W}$ *with* $m \leq n$*, the error of answering* $\mathbf{W}$ *under any data-independent* ϵ*-differentially private mechanism must be at least*

$$\Omega\left(\sum_i^n \frac{\lambda_i^2}{\epsilon^2}\right),$$

where $\lambda_1, \ldots, \lambda_n$ *are singular values of* $\mathbf{W}$*.*

When $m \leq n$, we can compare the lower bound in Corollary 4.13 with the SVD bound under the ϵ-matrix mechanism. It is clear that the bound in Corollary 4.13 is tighter unless all singular values of $\mathbf{W}$ are equal. When $m > n$, the quality of the SVD bound under the ϵ-matrix mechanism is not yet known. The discussion on the tightness and looseness of the SVD bound in the next section is based on the (ϵ, δ)-matrix mechanism and cannot be extended to the ϵ-matrix mechanism directly.

5. ANALYSIS OF THE SVD BOUND

In this section, we analyze the accuracy of the SVD bound as an approximation of the minimum error for a workload. We study the sufficient and necessary conditions under which the SVD bound is tight. In addition, we show the minimum error is equal to the bound over a specific class of workloads called variable-agnostic workloads and then generalize the result to the widely-studied class of data cube workloads. For both classes, strategies that achieve the minimum error can be constructed, as a by-product of the proof of the SVD bound.

We then show that the bound may be loose, underestimating the minimal error for some workloads. The worst case of looseness of the SVD bound is presented in Section 5.2, along with a formal estimate of the quality of the bound. We conclude this section with an example demonstrating empirically that error rates close to the lower bound can be achieved for workloads consisting of multi-dimensional range queries.

5.1 The Tightness of the SVD Bound

The circumstances under which the SVD bound is tight arise directly from inspection of the proof presented in Sec. 4.2. In particular, we noted the conditions that make the inequalities in equations (3), (4) and (5) actually equal. Those conditions are equivalent to a straightforward property of $\mathbf{W}^T\mathbf{W}$:

Theorem 5.1. *Given workload* $\mathbf{W}$*,* $\text{SVDB}(\mathbf{W})$ *is tight if and only if the diagonal entries of* $\sqrt{\mathbf{W}^T\mathbf{W}}$ *are all equal.*

There are workloads that satisfy the condition in Thm 5.1. Here we present one such special class of workloads, called *variable-agnostic workloads*, in which the queries on each cell are fully symmetric and swapping any two cells does not change $\mathbf{W}^T\mathbf{W}$.

Definition 5.2 (Variable-agnostic workload). *A workload* $\mathbf{W}$ *is* variable-agnostic *if* $\mathbf{W}^T\mathbf{W}$ *is unchanged when we swap any two columns of* $\mathbf{W}$*.*

For any variable-agnostic workload $\mathbf{W}$, $\mathbf{W}^T\mathbf{W}$ has the following special form: for some constants a and b such that $a > b$, all diagonal entries of $\mathbf{W}^T\mathbf{W}$ are equal to a and the remaining entries of $\mathbf{W}^T\mathbf{W}$ are equal to b.

The following theorem shows that any variable-agnostic workload $\mathbf{W}$ satisfies the condition in Thm 5.1. Furthermore, we also demonstrate the closed form expression of the SVD bound in case that n is a power of 2.

Theorem 5.3. *The SVD bound is tight for any variable-agnostic workload* $\mathbf{W}$*. In addition, when* $n = 2^k$ *for any nonnegative integer* k*,* $\text{SVDB}(\mathbf{W}) = \frac{1}{n}(\sqrt{a + (n-1)b} + (n-1)\sqrt{a-b})^2$*, where* a *is the value of diagonal entries of* $\mathbf{W}^T\mathbf{W}$ *and* b *is the value of off-diagonal entries of* $\mathbf{W}^T\mathbf{W}$*.*

As a concrete example, the workload AllPredicate(n) is variable-agnostic, and therefore we can construct its optimal strategy and compute the error rate directly.

Corollary 5.4. *The SVD bound is tight for the workload* AllPredicate(n)*. In addition, when* $n = 2^k$ *for any nonnegative integer* k*,* $\text{SVDB}(\text{AllPredicate}(n)) = \frac{2^{n-2}}{n}(n-1+\sqrt{n+1})^2$*.*

For variable-agnostic workloads, using a naive strategy like the identity matrix or the workload itself results in total error equal to na and the ratio by which the error is reduced using the strategy in Thm. 5.3 is approximately $1 - \frac{b}{a}$. In the case of AllPredicate(n), the ratio is at least as low as 0.5, which occurs when n is very large.

Another family of workloads for which the SVD bound is tight are those consisting of sets of data cube queries. A data cube workload consists of one or more cuboids, each of which contains all aggregation queries on all possible values of the cross-product of a set of attributes. Here we also consider the case that each cuboid can have its own weight, so that higher weighted queries will be estimated more accurately than lower weighted ones.

THEOREM 5.5. *The SVD bound is tight for any weighted data cube workload* $\mathbf{W}$.

Data cube workloads (a special case of marginal workloads) have been studied by the differential privacy community in both theory and practice [1, 6, 17]. Barak et al. [1] use the Fourier basis as a strategy for workloads consisting of marginals while Ding et al. [6] proposed an approximation algorithm for data cube workloads. Thm. 5.5 shows that under (ϵ, δ)-differential privacy we can now directly compute the optimal strategy, obviating the need to use an approximation algorithm or blindly relying on the Fourier basis for workloads of this type. The result in [17], however, involves data-dependent techniques and the comparison between [17] to the SVD bound relies on a thorough analysis of the spectral properties of data cube workloads, which is a direction of future work.

5.2 The Looseness of the SVD Bound

The SVD bound can also underestimate the minimum error when the workload is highly skewed. For example, the SVD bound does not work well when the sensitivity of one column in the workload is overwhelmingly larger than others. Recall the workload in Example 4.4, when $t \to 0$, the SVD bound will underestimate the total error by a factor of n. This is caused by the underestimate of the sensitivity of $\mathbf{A}$ considered in equation (3) in the proof of Thm. 4.2.

Since the proof of Thm. 4.2 constructs a concrete strategy, one way to measure the looseness of the SVD bound is to estimate its ratio to the actual error introduced by this strategy. Note that the sensitivity of the strategy is the only part of the SVD bound that is underestimated. The square of the sensitivity is the maximum diagonal entry of matrix $\mathbf{A}^T\mathbf{A}$, rather than the estimate given by $\text{trace}(\mathbf{A}^T\mathbf{A})/n$. The ratio between the actual sensitivity and the estimated sensitivity bounds the looseness of the SVD bound, as shown by the following theorem.

THEOREM 5.6. *Given an* $m \times n$ *workload* $\mathbf{W}$. *Let* d_0 *be the maximum diagonal entry of* $\sqrt{\mathbf{W}^T\mathbf{W}}$.

$$\text{MINERROR}(\mathbf{W}) \leq \frac{nd_0 P(\epsilon,\delta)\text{SVDB}(\mathbf{W})}{trace(\sqrt{\mathbf{W}^T\mathbf{W}})},$$

where $P(\epsilon, \delta) = \frac{2\log(2/\delta)}{\epsilon^2}$.

According to Thm. 5.6, the approximate ratio of the SVD bound corresponds to the ratio between d_0, the largest diagonal entry of $\sqrt{\mathbf{W}^T\mathbf{W}}$ and the trace of $\sqrt{\mathbf{W}^T\mathbf{W}}$, which is equal to the sum of all singular values of $\mathbf{W}$. This ratio, although it is upper-bounded by the ratio between the largest singular value of $\mathbf{W}$ and the sum of all singular values of $\mathbf{W}$, is much closer to 1 than the ratio between singular values. As a consequence, the skewness in singular values does not always lead to a bad approximation ratio for the SVD bound. For example, for variable-agnostic workloads, the largest singular value can be arbitrarily larger than the rest of the singular values, while the SVD bound is tight. Instead, the cases where the SVD bound has high approximation ratio, such as the one in Example 4.4, are due to the skewness of singular value of $\mathbf{W}$ and the particular distribution of singular vectors. The supreme SVD bound can help us to avoid some of these worst cases, but there is no guarantee of the quality of the bound with more sophisticated cases.

Nevertheless, for many common workloads, empirical evidence suggests that the SVD bound is quite close to the minimal error. The following example provides a comparison between the SVD bound and achievable error for a few common workloads.

EXAMPLE 5.7. *Returning to Table 2, we observe empirical evidence that for range and predicate workloads, there are strategies that come quite close to the SVD bound. The last column of Table 2 lists the error for the Eigen-design mechanism [19], which attempts to find approximately optimal strategies for any given workload by computing optimal weights for the eigenvectors of the workload. This algorithm is able to find a strategy whose error is within a factor of* 1.028 *and* 1.107 *of optimal for* ALLRANGE(2048) *and* ALLRANGE(64, 32)*, respectively.*

6. COMPARISON OF MECHANISMS

The matrix mechanism is a data-independent mechanism: the noise distribution (and therefore error) depends only on the workload and not on the particular input data. This makes it possible to process the workload once and apply the mechanism efficiently to any dataset. On the other hand, data-independent mechanisms lack the flexibility to exploit specific properties of individual datasets. In this section, we use the SVD bound to compare the error bounds of the matrix mechanism with error bounds of other mechanisms that are data-dependent.

6.1 Asymptotic Estimation of the SVD bound

Before the comparison, we first convert the SVD bound into an error measure that can be directly related to other bounds in the literature. We assume all queries in the workload have sensitivity at most one and estimate the SVD bound as a function of the domain size n and the number of queries m. Recall that the error in previous sections is defined as the total mean squared error of the queries. We introduce a new measure of error which bounds the maximum absolute error of the workload queries by α with high probability (controlled by β).

DEFINITION 6.1 ((α, β)-ACCURATE [12]). *Given a workload* $\mathbf{W}$*, an algorithm* $\mathcal{K}$ *is* (α, β)*-accurate if, for any uniformly drawn data vector* $\mathbf{x}$*, with a probability of at least* $1 - \beta$*,* $\max_{\mathbf{q}\in\mathbf{W}} |\mathcal{K}(\mathbf{q}, \mathbf{x}) - \mathbf{qx}| \leq \alpha$.

Since the SVD bound measures total error (rather than max error), here we modify the (α, β)-accuracy by bounding the root mean squared error of the workload.

DEFINITION 6.2 (RMS-(α, β)-ACCURATE). *Given a workload* $\mathbf{W}$*, an algorithm* $\mathcal{K}$ *is RMS-*(α, β)*-accurate if, for any uniformly drawn data vector* $\mathbf{x}$*, with a probability of at least* $1 - \beta$*,* $\sqrt{\sum_{\mathbf{q}\in\mathbf{W}} ||\mathcal{K}(\mathbf{q}, \mathbf{x}) - \mathbf{qx}||^2/|\mathbf{W}|} \leq \alpha$.

THEOREM 6.3. *Given an* $m \times n$ *workload* $\mathbf{W}$*, if the* SVDB($\mathbf{W}$) *is asymptotically tight, then there exists a strategy under which the matrix mechanism is RMS-*(α, β)*-accurate, where*

$$\alpha = O\left(\frac{\sqrt{\min(m,n)}\sqrt{\log(2/\delta)\log(\sqrt{\pi/2}/\beta)}}{\epsilon}\right).$$

Recall the discussion in Sec. 5.1 indicates that the SVD bound is tight or almost tight for many common workloads. Thus, it is reasonable to compare the asymptotic estimate of the SVD bound to the error introduced by other mechanisms.

Mechanism		α
1	Median [26]	$O\left(\frac{\sqrt{N}(\log n \log m)^{1/4}\sqrt{t(\log m+t)}}{\sqrt{\epsilon}}\right)$
2	MW [14]	$O\left(\frac{\sqrt{N}(\log n)^{1/4}\sqrt{t(\log m+t)}}{\sqrt{\epsilon}}\right)$
3	IDC [12]	$O\left(\frac{(nN)^{1/4}\sqrt{t(\log m+t)}}{\sqrt{\epsilon}}\right)$
4	Boosting [11]	$\tilde{O}\left(\frac{\sqrt{N\log n}\cdot t^{3/2}\log^{3/2} m}{\epsilon}\right)$
5	SVDB	$O\left(\frac{\sqrt{\min(m,n)}\cdot t}{\epsilon}\right)$

Table 3: For $t \geq 2$, bounds on the α required to achieve $(\epsilon, \exp(-t))$-differential privacy and accuracy measures of: $(\alpha, \exp(-t))$-accuracy (mechanisms 1-4); RMS-$(\alpha, \exp(-t))$-accuracy (mechanism 5).

6.2 Comparison of Error Bounds

Here we compare our SVD bound with other error bounds from data-dependent mechanisms. We include four competitors each representing fundamentally different mechanisms. The median mechanism [26] discards candidate data vectors that are inconsistent with historical query answers. The multiplicative weights mechanism (MW) [14] and the iterative database construction method (IDC) [12] repeatedly update an estimated data vector according to query answers. The boosting method [11] maintains a distribution of queries according to the quality of their answers and repeatedly samples queries from the distribution so as to improve their answers. The (α, β)-accuracy under (ϵ, δ)-differential privacy for the median and the multiplicative weight mechanism follows the result in [12].

Table 3 summarizes error bounds of different data dependent approaches. In particular, the comparison is over $(\epsilon, \exp(-t))$-differential privacy and $(\alpha, \exp(-t))$-accuracy. The workload $\mathbf{W}$ we considered contains m queries with sensitivity no larger than 1. The database is of size N, which means the sum of all x_i's in the data vector is N.

Observing the values of α in Table 3, the matrix mechanism has a greater dependence on ϵ compared with the median, the multiplicative weights and the iterative database construction methods. In addition, since the matrix mechanism is data-independent, it cannot take advantage of the input dataset so that it always assumes $n = N$. However, when N is sufficiently large ($\Theta(n)$) and $m = O(n)$, the SVD bound is smaller than the error of the Boosting method and can outperform other competitors when $m = \Omega(\exp(t/\epsilon))$.

6.3 Data-dependency & the matrix mechanism

Although the techniques of the matrix mechanism are data-independent, they can be deployed in a data-dependent way, blurring the distinction between mechanism types. The differentially private domain compression technique [21] may be applied to reduce the domain size n to $\Theta(N)$ with an additional $O(\log n)$ noise, which suggests a method for improving the error dependency of the matrix mechanism on n.

Further, the optimal strategy matrix used in the matrix mechanism represents the fundamental building blocks of the workload and the matrix mechanism reduces error by using the strategy queries as differentially private observations, instead of the workload queries. Recent data-dependent approaches can benefit from the same approach. In fact, [13] selects Fourier basis vectors adaptively in a data dependent manner, but could benefit from selecting from a more efficient strategy matrix. Therefore, the SVDB bound can serve as a baseline accuracy measure, which may be improved by data-dependent query selection.

7. RELATED WORK

The original description of the matrix mechanism [18] focuses primarily on ϵ-differential privacy, with a brief consideration of (ϵ, δ)-differential privacy. A number of proposed mechanisms can be formulated as instances of the matrix mechanism: techniques for accurately answering range queries are presented in [27, 16]; low-order marginals are studied in [1] using a Fourier transformation as the strategy (combined with other techniques for achieving integral consistency) as well as in [29] by optimally scaling a manually-chosen set of strategy queries; an algorithm for generating good strategies for answering sets of data cube queries is introduced in [6]; and an algorithm for computing optimal low-rank strategy matrices is presented in [30]. The lower bound presented in this work provides a theoretical method to evaluate the quality of each of the approaches above, assuming (ϵ, δ)-differential privacy.

In recent work, Nikolov et al. [23] propose an algorithm whose error is within a ratio of $O(\log^2 \text{rank}(\mathbf{W}) \log(1/\delta))$ to the optimal error under *any* data-independent (ϵ, δ)-differentially private mechanism (not limited to instances of the matrix mechanism). Their algorithm is in fact a special case of the (ϵ, δ)-matrix mechanism, so this approximation ratio also bounds the ratio between the SVD bound and the minimum achievable error of *all* possible data-independent (ϵ, δ)-differential private mechanisms.

Blum et al. [4] describe a very general mechanism for synthetic data release, in which error rates are related to the VC dimension of the workload. However, for many workloads of linear queries, VC dimension is too coarse-grained to provide a useful measure of workload error complexity. For example, the VC dimension for any workload of d-dimensional range queries that can not be embedded into $(d-1)$-dimensional spaces is always $d+1$, despite the fact that such workloads could have very different achievable error rates.

Hardt et al. [15] present a lower bound on error for low rank workloads. Similar to the SVD bound, this geometric bound can also be represented as a function of the singular values of the workload. In particular, the bound uses the geometric average of the singular values rather than the algebraic average in the SVD bound. The geometric bound provides a more general guarantee since it is a lower bound on all ϵ-differential privacy mechanisms. But it is not directly comparable with the SVD bound since it bounds the mean absolute error rather than mean squared error in the SVD bound. Lower and upper bounds on answering all k-way marginals with a data dependent mechanism are discussed in [17]. Though it is clear that the SVD bound is tight in the case of all k-way marginals (since it is a special case of data cube) comparison with [17] requires a careful analysis of the singular values of workloads of k-way marginals and is a direction for future investigation.

There are also error bounds from data-dependent mechanisms, some of which we have compared with in Sec. 6. A data-dependent approach for range queries is described in [5]. The median mechanism [26] drops data vectors that are inconsistent with query answers in each step. Dwork et al. [11] samples linear queries in each step and modifies the sample distribution with the new query answers. In [14, 13, 12], the authors recursively update the estimated data vector to re-

duce the error on linear queries. More generally, Dwork et al. provide an error bound using an arbitrary differentially private mechanism [10] but not specifically for linear counting queries. Those analyses lead to smaller error than the matrix mechanism over sparse databases by analyzing the properties of the underlying database. Thus their error bounds reflect the connection between workloads and databases but cannot lead to bounds on error that can be used to characterize the error complexity of workloads.

8. CONCLUSION

We have shown that, for a general class of (ϵ, δ)-differentially private algorithms, the error rate achievable for a set of queries is determined by the spectral properties of the queries when they are represented in matrix form. The result is a lower bound on error which is a simple function of the eigenvalues of the query matrix. The bound can be used to assess the quality of a number of existing differentially private algorithms, to directly construct error-optimal strategies in some cases, to compare the hardness of query sets, and to guide users in the design of query workloads.

Acknowledgements

We appreciate the helpful comments of the anonymous reviewers. Li was supported by NSF CNS-1012748. Miklau was partially supported by NSF CNS-1012748, NSF CNS-0964094, and the European Research Council under the Webdam grant.

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. Ben-Israel and T. Greville. *Generalized inverses: Theory and applications*, volume 15. Springer, 2003.

[3] A. Bhaskara, D. Dadush, R. Krishnaswamy, and K. Talwar. Unconditional differentially private mechanisms for linear queries. In *STOC*, pages 1269–1284, New York, NY, USA, 2012.

[4] A. Blum, K. Ligett, and A. Roth. A learning theory approach to non-interactive database privacy. In *STOC*, pages 609–618, 2008.

[5] G. Cormode, M. Procopiuc, E. Shen, D. Srivastava, and T. Yu. Differentially private spatial decompositions. *ICDE*, pages 20–31, 2012.

[6] B. Ding, M. Winslett, J. Han, and Z. Li. Differentially private data cubes: optimizing noise sources and consistency. In *SIGMOD*, pages 217–228, 2011.

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

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

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

[10] C. Dwork, M. Naor, O. Reingold, G. Rothblum, and S. Vadhan. On the complexity of differentially private data release: efficient algorithms and hardness results. In *STOC*, pages 381–390, 2009.

[11] C. Dwork, G. N. Rothblum, and S. P. Vadhan. Boosting and differential privacy. In *FOCS*, pages 51–60, 2010.

[12] A. Gupta, A. Roth, and J. Ullman. Iterative constructions and private data release. In *TCC*, pages 339–356, 2012.

[13] M. Hardt, K. Ligett, and F. McSherry. A simple and practical algorithm for differentially private data release. In *NIPS*, pages 2348–2356, 2012.

[14] M. Hardt and G. Rothblum. A multiplicative weights mechanism for privacy-preserving data analysis. In *FOCS*, pages 61–70, 2010.

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

[16] M. Hay, V. Rastogi, G. Miklau, and D. Suciu. Boosting the accuracy of differentially-private histograms through consistency. *PVLDB*, 3(1-2):1021–1032, 2010.

[17] S. 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 *STOC*, pages 775–784, 2010.

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

[19] C. Li and G. Miklau. An adaptive mechanism for accurate query answering under differential privacy. *PVLDB*, 5(6):514–525, 2012.

[20] C. Li and G. Miklau. Optimal error of query sets under the differentially-private matrix mechanism. *CoRR*, abs/1202.3399, 2012.

[21] Y. D. Li, Z. Zhang, M. Winslett, and Y. Yang. Compressive mechanism: utilizing sparse representation in differential privacy. In *WPES*, pages 177–182, 2011.

[22] F. McSherry and I. Mironov. Differentially Private Recommender Systems : Building Privacy into the Netflix Prize Contenders. In *SIGKDD*, pages 627–636, 2009.

[23] A. Nikolov, K. Talwar, and L. Zhang. The geometry of differential privacy: the sparse and approximate cases. *CoRR*, arXiv/1212.0297, 2012.

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

[25] V. Rastogi, D. Suciu, and S. Hong. The boundary between privacy and utility in data publishing. In *VLDB*, number 531-542, 2007.

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

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

[28] Y. Xiao, L. Xiong, and C. Yuan. Differentially private data release through multidimensional partitioning. In *SDM*, pages 150–168, 2010.

[29] G. Yaroslavtsev, G. Cormode, C. M. Procopiuc, and D. Srivastava. Accurate and efficient private release of datacubes and contingency tables. In *ICDE*, 2013.

[30] G. Yuan, Z. Zhang, M. Winslett, X. Xiao, Y. Yang, and Z. Hao. Low-rank mechanism: Optimizing batch queries under differential privacy. *PVLDB*, 5(11):1136–1147, 2012.