Boosting the Accuracy of Differentially Private Histograms Through Consistency

Michael Hay[†], Vibhor Rastogi[‡], Gerome Miklau[†], Dan Suciu[‡]

† University of Massachusetts Amherst
{mhay,miklau}@cs.umass.edu

‡ University of Washington
{vibhor,suciu}@cs.washington.edu

ABSTRACT

We show that it is possible to significantly improve the accuracy of a general class of histogram queries while satisfying differential privacy. Our approach carefully chooses a set of queries to evaluate, and then exploits consistency constraints that should hold over the noisy output. In a post-processing phase, we compute the consistent input most likely to have produced the noisy output. The final output is differentially-private and consistent, but in addition, it is often much more accurate. We show, both theoretically and experimentally, that these techniques can be used for estimating the degree sequence of a graph very precisely, and for computing a histogram that can support arbitrary range queries accurately.

1. INTRODUCTION

Recent work in differential privacy [8] has shown that it is possible to analyze sensitive data while ensuring strong privacy guarantees. Differential privacy is typically achieved through random perturbation: the analyst issues a query and receives a noisy answer. To ensure privacy, the noise is carefully calibrated to the *sensitivity* of the query. Informally, query sensitivity measures how much a small change to the database—such as adding or removing a person's private record—can affect the query answer. Such query mechanisms are simple, efficient, and often quite accurate. In fact, one mechanism has recently been shown to be optimal for a single counting query [9]—i.e., there is no better noisy answer to return under the desired privacy objective.

However, analysts typically need to compute multiple statistics on a database. Differentially private algorithms extend nicely to a set of queries, but there can be difficult trade-offs among alternative strategies for answering a workload of queries. Consider the analyst of a private student database who requires answers to the following queries: the total number of students, x_t, the number of students x_A, x_B, x_C, x_D, x_F receiving grades A, B, C, D, and F respectively, and the number of passing students, x_p (grade D or higher).

Proceedings of the VLDB Endowment, Vol. 3, No. 1

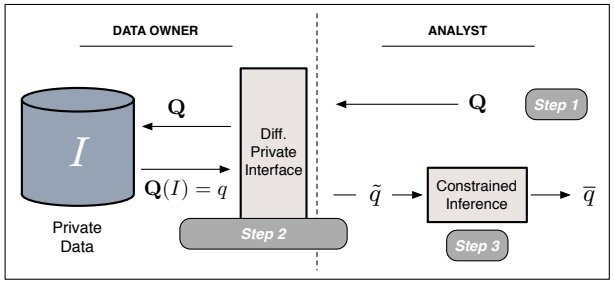


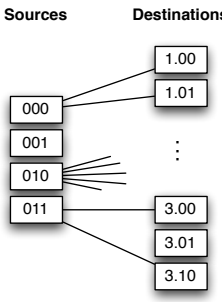
Figure 1: Our approach to querying private data.

Using a differentially private interface, a first alternative is to request noisy answers for just $(x_A, x_B, x_C, x_D, x_F)$ and use those answers to compute answers for x_t and x_p by summation. The sensitivity of this set of queries is 1 because adding or removing one tuple changes exactly one of the five outputs by a value of one. Therefore, the noise added to individual answers is low and the noisy answers are accurate estimates of the truth. Unfortunately, the noise accumulates under summation, so the estimates for x_t and x_p are worse.

A second alternative is to request noisy answers for all queries $(x_t, x_p, x_A, x_B, x_C, x_D, x_F)$. This query set has sensitivity 3 (one change could affect three return values, each by a value of one), and the privacy mechanism must add more noise to each component. This means the estimates for x_A, x_B, x_C, x_D, x_F are worse than above, but the estimates for x_t and x_p may be more accurate. There is another concern, however: inconsistency. The noisy answers are likely to violate the following constraints, which one would naturally expect to hold: $x_t = x_p + x_F$ and $x_p = x_A + x_B + x_C + x_D$. This means the analyst must find a way to reconcile the fact that there are two different estimates for the total number of students and two different estimates for the number of passing students. We propose a technique for resolving inconsistency in a set of noisy answers, and show that doing so can actually increase accuracy. As a result, we show that strategies inspired by the second alternative can be superior in many cases.

Overview of Approach. Our approach, shown pictorially in Figure 1, involves three steps.

First, given a task—such as computing a histogram over student grades—the analyst chooses a set of queries $\mathbf{Q}$ to send to the data owner. The choice of queries will depend on the particular task, but in this work they are chosen so that



(a) Trace data

Query Definitions:

$\mathbf{L}$: $\langle C_{000}, C_{001}, C_{010}, C_{011}\rangle$
$\mathbf{H}$: $\langle C_{0**}, C_{00*}, C_{01*}, C_{000}, C_{001}, C_{010}, C_{011}\rangle$
$\mathbf{S}$: $sort(\mathbf{L})$

True answer	*Private output*	*Inferred answer*
$\mathbf{L}(I) = \langle 2, 0, 10, 2\rangle$	$\tilde{\mathbf{L}}(I) = \langle 3, 1, 11, 1\rangle$	
$\mathbf{H}(I) = \langle 14, 2, 12, 2, 0, 10, 2\rangle$	$\tilde{\mathbf{H}}(I) = \langle 13, 3, 11, 4, 1, 12, 1\rangle$	$\overline{\mathbf{H}}(I) = \langle 14, 3, 11, 3, 0, 11, 0\rangle$
$\mathbf{S}(I) = \langle 0, 2, 2, 10\rangle$	$\tilde{\mathbf{S}}(I) = \langle 1, 2, 0, 11\rangle$	$\overline{\mathbf{S}}(I) = \langle 1, 1, 1, 11\rangle$

(b) Query variations

Figure 2: (a) Illustration of sample data representing a bipartite graph of network connections; (b) Definitions and sample values for alternative query sequences: L counts the number of connections for each *source*, H provides a hierarchy of range counts, and S returns an ordered degree sequence for the implied graph.

constraints hold among the answers. For example, rather than issue $(x_A, x_B, x_C, x_D, x_F)$, the analyst would formulate the query as $(x_t, x_p, x_A, x_B, x_C, x_D, x_F)$, which has consistency constraints. The query set $\mathbf{Q}$ is sent to the data owner.

In the second step, the data owner answers the set of queries, using a standard differentially-private mechanism [8], as follows. The queries are evaluated on the private database and the true answer $\mathbf{Q}(I)$ is computed. Then random independent noise is added to each answer in the set, where the data owner scales the noise based on the sensitivity of the query set. The set of noisy answers $\tilde{q}$ is sent to the analyst. Importantly, because this step is unchanged from [8], it offers the same differential privacy guarantee.

The above step ensures privacy, but the set of noisy answers returned may be inconsistent. In the third and final step, the analyst post-processes the set of noisy answers to resolve inconsistencies among them. We propose a novel approach for resolving inconsistencies, called *constrained inference*, that finds a new set of answers $\overline{q}$ that is the "closest" set to $\tilde{q}$ that also satisfies the consistency constraints.

For two histogram tasks, our main technical contributions are efficient techniques for the third step and a theoretical and empirical analysis of the accuracy of $\overline{q}$. The surprising finding is that $\overline{q}$ can be significantly more accurate than $\tilde{q}$.

We emphasize that the constrained inference step has *no impact* on the differential privacy guarantee. The analyst performs this step without access to the private data, using only the constraints and the noisy answers, $\tilde{q}$. The noisy answers $\tilde{q}$ are the output of a differentially private mechanism; any post-processing of the answers cannot diminish this rigorous privacy guarantee. The constraints are properties of the *query*, not the database, and therefore known by the analyst *a priori*. For example, the constraint $x_p = x_A + x_B + x_C + x_D$ is simply the definition of x_p.

Intuitively, however, it would seem that if noise is added for privacy and then constrained inference reduces the noise, some privacy has been lost. In fact, our results show that existing techniques add more noise than is strictly necessary to ensure differential privacy. The extra noise provides no quantifiable gain in privacy but does have a significant cost in accuracy. We show that constrained inference can be an effective strategy for boosting accuracy.

The increase in accuracy we achieve depends on the input database and the privacy parameters. For instance, for some databases and levels of noise the perturbation may tend to produce answers that do not violate the constraints. In this case the inference step would not improve accuracy. But we show that our inference process never reduces accuracy and give conditions under which it will boost accuracy. In practice, we find that many real datasets have data distributions for which our techniques significantly improve accuracy.

Histogram tasks. We demonstrate this technique on two specific tasks related to histograms. For relational schema $R(A, B, \dots)$, we choose one attribute A on which histograms are built (called the *range* attribute). We assume the domain of A, *dom*, is ordered.

We explain these tasks using sample data that will serve as a running example throughout the paper, and is also the basis of later experiments. The relation $R(src, dst)$, shown in Fig. 2, represents a trace of network communications between a source IP address (*src*) and a destination IP address (*dst*). It is bipartite because it represents flows through a gateway router from internal to external addresses.

In a conventional histogram, we form disjoint intervals for the range attribute and compute counting queries for each specified range. In our example, we use *src* as the range attribute. There are four source addresses present in the table. If we ask for counts of all unit-length ranges, then the histogram is simply the sequence $\langle 2, 0, 10, 2\rangle$ corresponding to the (out) degrees of the source addresses $\langle 000, 001, 010, 011\rangle$.

Our first histogram task is an **unattributed histogram**, in which the intervals themselves are irrelevant to the analysis and so we report only a multiset of frequencies. For the example histogram, the multiset is $\{0, 2, 2, 10\}$. An important instance of an unattributed histogram is the degree sequence of a graph, a crucial measure that is widely studied [17]. If the tuples of R represent queries submitted to a search engine, and A is the search term, then an unattributed histogram shows the frequency of occurrence of all terms (but not the terms themselves), and can be used to study the distribution.

For our second histogram task, we consider more conventional sequences of counting queries in which the intervals studied may be irregular and overlapping. In this case, simply returning unattributed counts is insufficient. And because we cannot predict ahead of time all the ranges of interest, our goal is to compute privately a set of statistics sufficient to support arbitrary interval counts and thus any histogram. We call this a **universal histogram**.

Continuing the example, a universal histogram allows the analyst to count the number of packets sent from any single address (e.g., the counts from source address 010 is 10), or from any range of addresses (e.g., the total number of packets is 14, and the number of packets from a source address matching prefix $01*$ is 12).

While a universal histogram can be used compute an unattributed histogram, we distinguish between the two because we show the latter can be computed much more accurately.

Contributions. For both unattributed and universal histograms, we propose a strategy for boosting the accuracy of existing differentially private algorithms. For each task, (1) we show that there is an efficiently-computable, closed-form expression for the *consistent* query answer closest to a private randomized output; (2) we prove bounds on the error of the inferred output, showing under what conditions inference boosts accuracy; (3) we demonstrate significant improvements in accuracy through experiments on real data sets. Unattributed histograms are extremely accurate, with error at least an order of magnitude lower than existing techniques. Our approach to universal histograms can reduce error for larger ranges by 45-98%, and improves on all ranges in some cases.

2. BACKGROUND

In this section, we introduce the concept of query sequences and how they can be used to support histograms. Then we review differential privacy and show how queries can be answered under differential privacy. Finally, we formalize our constrained inference process.

All of the tasks considered in this paper are formulated as *query sequences* where each element of the sequence is a simple count query on a range. We write intervals as $[x, y]$ for $x, y \in dom$, and abbreviate interval $[x, x]$ as $[x]$. A counting query on range attribute A is:

$$c([x, y]) = \text{Select count(*) From R Where } x \leq \text{R.A} \leq y$$

We use $\mathbf{Q}$ to denote a generic query sequence (please see Appendix A for an overview of notational conventions). When $\mathbf{Q}$ is evaluated on a database instance I, the output, $\mathbf{Q}(I)$, includes one answer to each counting query, so $\mathbf{Q}(I)$ is a vector of non-negative integers. The i^{th} query in $\mathbf{Q}$ is $\mathbf{Q}[i]$.

We consider the common case of a histogram over unit-length ranges. The conventional strategy is to simply compute counts for all unit-length ranges. This query sequence is denoted $\mathbf{L}$:

$$\mathbf{L} = \langle\, c([x_1]), \ldots c([x_n])\,\rangle, x_i \in dom$$

Example 1. *Using the example in Fig 2, we assume the domain of src contains just the 4 addresses shown. Query* $\mathbf{L}$ *is* $\langle c([000]), c([001]), c([010]), c([011])\rangle$ *and* $\mathbf{L}(I) = \langle 2, 0, 10, 2\rangle$.

2.1 Differential Privacy

Informally, an algorithm is differentially private if it is insensitive to small changes in the input. Formally, 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 2.1 (ϵ-differential privacy). *Algorithm A is ϵ-differentially private if for all instances I, any $I' \in nbrs(I)$, and any subset of outputs $S \subseteq Range(A)$, the following holds:*

$$Pr[A(I) \in S] \leq \exp(\epsilon) \times Pr[A(I') \in S]$$

where the probability is taken over the randomness of the A.

Differential privacy has been defined inconsistently in the literature. The original concept, called ϵ-indistinguishability [8], defines neighboring databases using hamming distance rather than symmetric difference (i.e., I' is obtained from I by *replacing* a tuple rather than adding/removing a tuple). The choice of definition affects the calculation of query sensitivity. We use the above definition (from Dwork [6]) but observe that our results also hold under indistinguishability, due to the fact that ϵ-differential privacy (as defined above) implies 2ϵ-indistinguishability.

To answer queries under differential privacy, we use the Laplace mechanism [8], which achieves differential privacy by adding noise to query answers, where the noise is sampled from the Laplace distribution. The magnitude of the noise depends on the query's *sensitivity*, defined as follows (adapting the definition to the query sequences considered in this paper).

Definition 2.2 (Sensitivity). *Let $\mathbf{Q}$ be a sequence of counting queries. The sensitivity of $\mathbf{Q}$, denoted $S_{\mathbf{Q}}$, is*

$$\Delta_{\mathbf{Q}} = \max_{I, I' \in nbrs(I)} \left\| \mathbf{Q}(I) - \mathbf{Q}(I') \right\|_1$$

Throughout the paper, we use $||\mathbf{X} - \mathbf{Y}||_p$ to denote the L_p distance between vectors $\mathbf{X}$ and $\mathbf{Y}$.

Example 2. *The sensitivity of query $\mathbf{L}$ is 1. Database I' can be obtained from I by adding or removing a single record, which affects exactly one of the counts in $\mathbf{L}$ by exactly 1.*

Given query $\mathbf{Q}$, the Laplace mechanism first computes the query answer $\mathbf{Q}(I)$ and then adds random noise independently to each answer. The noise is drawn from a zero-mean Laplace distribution with scale σ. As the following proposition shows, differential privacy is achieved if the Laplace noise is scaled appropriately to the sensitivity of $\mathbf{Q}$ and the privacy parameter ϵ.

Proposition 1 (Laplace mechanism [8]). *Let $\mathbf{Q}$ be a query sequence of length d, and let $\langle Lap(\sigma)\rangle^d$ denote a d-length vector of i.i.d. samples from a Laplace with scale σ. The randomized algorithm $\tilde{\mathbf{Q}}$ that takes as input database I and outputs the following vector is ϵ-differentially private:*

$$\tilde{\mathbf{Q}}(I) = \mathbf{Q}(I) + \langle Lap(\Delta_{\mathbf{Q}}/\epsilon)\rangle^d$$

We apply this technique to the query $\mathbf{L}$. Since, $\mathbf{L}$ has sensitivity 1, the following algorithm is ϵ-differentially private:

$$\tilde{\mathbf{L}}(I) = \mathbf{L}(I) + \langle \text{Lap}(1/\epsilon)\rangle^n$$

We rely on Proposition 1 to ensure privacy for the query sequences we propose in this paper. We emphasize that the proposition holds for *any* query sequence $\mathbf{Q}$, regardless of correlations or constraints among the queries in $\mathbf{Q}$. Such dependencies are accounted for in the calculation of sensitivity. (For example, consider the correlated sequence $\mathbf{Q}$ that consists of the *same* query repeated k times, then the sensitivity of $\mathbf{Q}$ is k times the sensitivity of the query.)

We present the case where the analyst issues a single query sequence $\mathbf{Q}$. To support multiple query sequences, the protocol that computes a ϵ_i-differentially private response to the i^{th} sequence is $(\sum \epsilon_i)$-differentially private.

To analyze the accuracy of the randomized query sequences proposed in this paper we quantify their error. $\tilde{\mathbf{Q}}$ can be considered an estimator for the true value $\mathbf{Q}(I)$. We use the common Mean Squared Error as a measure of accuracy.

DEFINITION 2.3 (ERROR). *For a randomized query sequence $\tilde{\mathbf{Q}}$ whose input is $\mathbf{Q}(I)$, the error$(\tilde{\mathbf{Q}})$ is $\sum_i \mathbb{E}(\tilde{\mathbf{Q}}[i] - \mathbf{Q}[i])^2$ Here $\mathbb{E}$ is the expectation taken over the possible randomness in generating $\tilde{\mathbf{Q}}$.*

For example, $error(\tilde{\mathbf{L}}) = \sum_i \mathbb{E}(\tilde{\mathbf{L}}[i] - \mathbf{L}[i])^2$ which simplifies to: $n\,\mathbb{E}[Lap(1/\epsilon)^2] = n\,Var(Lap(1/\epsilon)) = 2n/\epsilon^2$.

2.2 Constrained Inference

While $\tilde{\mathbf{L}}$ can be used to support unattributed and universal histograms under differential privacy, the main contribution of this paper is the development of more accurate query strategies based on the idea of constrained inference. The specific strategies are described in the next sections. Here, we formulate the constrained inference problem.

Given a query sequence $\mathbf{Q}$, let $\gamma_{\mathbf{Q}}$ denote a set of constraints which must hold among the (true) answers. The constrained inference process takes the randomized output of the query, denoted $\tilde{q} = \tilde{\mathbf{Q}}(I)$, and finds the sequence of query answers $\overline{q}$ that is "closest" to $\tilde{q}$ and also satisfies the constraints of $\gamma_{\mathbf{Q}}$. Here closest is determined by L_2 distance, and the result is the *minimum L_2 solution*:

DEFINITION 2.4 (MINIMUM L_2 SOLUTION). *Let $\mathbf{Q}$ be a query sequence with constraints $\gamma_{\mathbf{Q}}$. Given a noisy query sequence $\tilde{q} = \tilde{\mathbf{Q}}(I)$, a minimum L_2 solution, denoted $\overline{q}$, is a vector $\overline{q}$ that satisfies the constraints $\gamma_{\mathbf{Q}}$ and at the same time minimizes $||\tilde{q} - \overline{q}||_2$.*

We use $\overline{\mathbf{Q}}$ to denote the two step randomized process in which the data owner first computes $\tilde{q} = \tilde{\mathbf{Q}}(I)$ and then computes the minimum L_2 solution from $\tilde{q}$ and $\gamma_{\mathbf{Q}}$. (Alternatively, the data owner can release $\tilde{q}$ and the latter step can be done by the analyst.) Importantly, the inference step has no impact on privacy, as stated below. (Proofs appear in the Appendix.)

PROPOSITION 2. *If $\tilde{\mathbf{Q}}$ satisfies ϵ-differential privacy, then $\overline{\mathbf{Q}}$ satisfies ϵ-differential privacy.*

3. UNATTRIBUTED HISTOGRAMS

To support unattributed histograms, one could use the query sequence $\mathbf{L}$. However, it contains "extra" information—the attribution of each count to a particular range—which is irrelevant for an *unattributed* histogram. Since the association between $\mathbf{L}[i]$ and i is not required, any permutation of the unit-length counts is a correct response for the unattributed histogram. We formulate an alternative query that asks for the counts of $\mathbf{L}$ in rank order. As we will show, ordering does not increase sensitivity, but it does introduce inequality constraints that can be exploited by inference.

Formally, let a_i refer to the answer to $\mathbf{L}[i]$ and let $\mathcal{U} = \{a_1, \ldots, a_n\}$ be the multiset of answers to queries in $\mathbf{Q}$. We write $rank_i(\mathcal{U})$ to refer to the i^{th} smallest answer in $\mathcal{U}$. Then the query $\mathbf{S}$ is defined as

$$\mathbf{S} = \langle rank_1(\mathcal{U}), \ldots, rank_n(\mathcal{U}) \rangle$$

EXAMPLE 3. *In the example in Fig 2, we have $\mathbf{L}(I) = \langle 2, 0, 10, 2 \rangle$ while $\mathbf{S}(I) = \langle 0, 2, 2, 10 \rangle$. Thus, the answer $\mathbf{S}(I)$ contains the same counts as $\mathbf{L}(I)$ but in* sorted *order.*

To answer $\mathbf{S}$ under differential privacy, we must determine its sensitivity.

PROPOSITION 3. *The sensitivity of $\mathbf{S}$ is 1.*

By Propositions 1 and 3, the following algorithm is ϵ-differentially private:

$$\tilde{\mathbf{S}}(I) \quad = \quad \mathbf{S}(I) + \langle \mathrm{Lap}(1/\epsilon) \rangle^n$$

Since the same magnitude of noise is added to $\mathbf{S}$ as to $\mathbf{L}$, the accuracy of $\tilde{\mathbf{S}}$ and $\tilde{\mathbf{L}}$ is the same. However, $\mathbf{S}$ implies a powerful set of constraints. Notice that the ordering occurs *before* noise is added. Thus, the analyst knows that the returned counts are ordered according to the true rank order. If the returned answer contains out-of-order counts, this must be caused by the addition of random noise, and they are inconsistent. Let $\gamma_{\mathbf{S}}$ denote the set of inequalities $\mathbf{S}[i] \leq \mathbf{S}[i+1]$ for $1 \leq i < n$. We show next how to exploit these constraints to boost accuracy.

3.1 Constrained Inference: Computing $\overline{\mathbf{S}}$

As outlined in the introduction, the analyst sends query $\mathbf{S}$ to the data owner and receives a noisy answer $\tilde{s} = \tilde{\mathbf{S}}(I)$, the output of the differentially private algorithm $\tilde{\mathbf{S}}$ evaluated on the private database I. We now describe a technique for post-processing $\tilde{s}$ to find an answer that is consistent with the ordering constraints.

Formally, given $\tilde{s}$, the objective is to find an $\overline{s}$ that minimizes $||\tilde{s} - \overline{s}||_2$ subject to the constraints $\overline{s}[i] \leq \overline{s}[i+1]$ for $1 \leq i < n$. The solution has a surprisingly elegant closed-form. Let $\tilde{s}[i, j]$ be the subsequence of $j - i + 1$ elements: $\langle \tilde{s}[i],\ \tilde{s}[i+1],\ \ldots, \tilde{s}[j] \rangle$. Let $\tilde{M}[i, j]$ be the mean of these elements, i.e. $\tilde{M}[i, j] = \sum_{k=i}^{j} \tilde{s}[k]/(j - i + 1)$.

THEOREM 1. *Denote $L_k = \min_{j \in [k,n]} \max_{i \in [1,j]} \tilde{M}[i, j]$ and $U_k = \max_{i \in [1,k]} \min_{j \in [i,n]} \tilde{M}[i, j]$. The minimum L_2 solution $\overline{s}$, is unique and given by: $\overline{s}[k] = L_k = U_k$.*

Since we first stated this result in a technical report [12], we have learned that this problem is an instance of isotonic regression (i.e., least squares regression under ordering constraints on the estimands). The statistics literature gives several characterizations of the solution, including the above min-max formulas (cf. Barlow et al. [3]), as well as linear time algorithms for computing it (cf. Barlow et al. [2]).

EXAMPLE 4. *We give three examples of $\tilde{s}$ and its closest ordered sequence $\overline{s}$. First, suppose $\tilde{s} = \langle 9, 10, 14 \rangle$. Since $\tilde{s}$ is already ordered, $\overline{s} = \tilde{s}$. Second, $\tilde{s} = \langle 9, 14, 10 \rangle$, where the last two elements are out of order. The closest ordered sequence is $\overline{s} = \langle 9, 12, 12 \rangle$. Finally, let $\tilde{s} = \langle 14, 9, 10, 15 \rangle$. The sequence is in order except for $\tilde{s}[1]$. While changing the first element from 14 to 9 would make it ordered, its distance from $\tilde{s}$ would be $(14 - 9)^2 = 25$. In contrast, $\overline{s} = \langle 11, 11, 11, 15 \rangle$ and $||\tilde{s} - \overline{s}||_2 = 14$.*

3.2 Utility Analysis: the Accuracy of $\overline{\mathbf{S}}$

Prior work in isotonic regression has shown inference cannot hurt, i.e., the accuracy of $\overline{\mathbf{S}}$ is no lower than $\tilde{\mathbf{S}}$ [13].

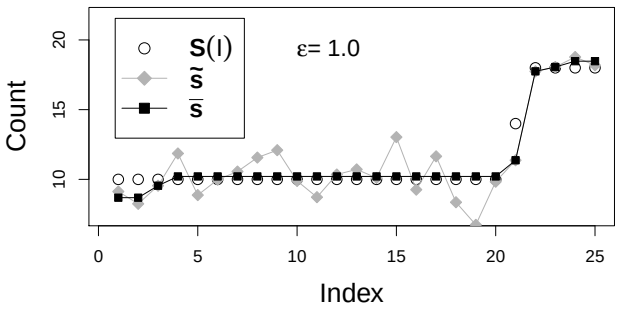


Figure 3: Example of how $\overline{s}$ reduces the error of $\tilde{s}$.

However, we are not aware of any results that give conditions for which $\overline{\mathbf{S}}$ is more accurate than $\tilde{\mathbf{S}}$. Before presenting a theoretical statement of such conditions, we first give an illustrative example.

EXAMPLE 5. *Figure 3 shows a sequence $\mathbf{S}(I)$ along with a sampled $\tilde{s}$ and inferred $\overline{s}$. While the values in $\tilde{s}$ deviate considerably from $\mathbf{S}(I)$, $\overline{s}$ lies very close to the true answer. In particular, for subsequence $[1, 20]$, the true sequence $\mathbf{S}(I)$ is uniform and the constrained inference process effectively averages out the noise of $\tilde{s}$. At the twenty-first position, which is a unique count in $\mathbf{S}(I)$, and constrained inference does not refine the noisy answer, i.e., $\overline{s}[21] = \tilde{s}[21]$.*

Fig 3 suggests that $error(\overline{\mathbf{S}})$ will be low for sequences in which many counts are the same (Fig 7 in Appendix C gives another intuitive view of the error reduction). The following theorem quantifies the accuracy of $\overline{\mathbf{S}}$ precisely. Let n and d denote the number of values and the number of distinct values in $\mathbf{S}(I)$ respectively. Let $n_1, n_2, \ldots, n_d$ be the number of times each of the d distinct values occur in $\mathbf{S}(I)$ (thus $\sum_i n_i = n$).

THEOREM 2. *There exist constants c_1 and c_2 independent of n and d such that*

$$error(\overline{\mathbf{S}}) \leq \sum_{i=1}^{d} \frac{c_1 \log^3 n_i + c_2}{\epsilon^2}$$

Thus $error(\overline{\mathbf{S}}) = O(d \log^3 n/\epsilon^2)$ whereas $error(\tilde{\mathbf{S}}) = \Theta(n/\epsilon^2)$.

The above theorem shows that constrained inference can boost accuracy, and the improvement depends on properties of the input $\mathbf{S}(I)$. In particular, if the number of distinct elements d is 1, then $error(\overline{\mathbf{S}}) = O(\log^3 n/\epsilon^2)$, while $error(\tilde{\mathbf{S}}) = \Theta(n/\epsilon^2)$. On the other hand, if $d = n$, then $error(\overline{\mathbf{S}}) = O(n/\epsilon^2)$ and thus both $error(\overline{\mathbf{S}})$ and $error(\tilde{\mathbf{S}})$ scale linearly in n. For many practical applications, $d \ll n$, which makes $error(\overline{\mathbf{S}})$ significantly lower than $error(\tilde{\mathbf{S}})$. In Sec. 5, experiments on real data demonstrate that the error of $\overline{\mathbf{S}}$ can be orders of magnitude lower than that of $\tilde{\mathbf{S}}$.

4. UNIVERSAL HISTOGRAMS

While the query sequence $\mathbf{L}$ is the conventional strategy for computing a universal histogram, this strategy has limited utility under differential privacy. While accurate for small ranges, the noise in the unit-length counts accumulates under summation, so for larger ranges, the estimates can easily become useless.

We propose an alternative query sequence that, in addition to asking for unit-length intervals, asks for interval counts of larger granularity. To ensure privacy, slightly more noise must be added to the counts. However, this strategy has the property that any range query can be answered via a linear combination of only a small number of noisy counts, and this makes it much more accurate for larger ranges.

Our alternative query sequence, denoted $\mathbf{H}$, consists of a sequence of hierarchical intervals. Conceptually, these intervals are arranged in a tree T. Each node $v \in T$ corresponds to an interval, and each node has k children, corresponding to k equally sized subintervals. The root of the tree is the interval $[x_1, x_n]$, which is recursively divided into subintervals until, at leaves of the tree, the intervals are unit-length, $[x_1], [x_2], \ldots, [x_n]$. For notational convenience, we define the height of the tree ℓ as the number of *nodes*, rather than edges, along the path from a leaf to the root. Thus, $\ell = \log_k n + 1$. To transform the tree into a sequence, we arrange the interval counts in the order given by a breadth-first traversal of the tree.

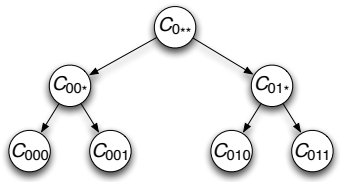


Figure 4: The tree T associated with query H for the example in Fig. 2 for $k = 2$.

EXAMPLE 6. *Continuing from the example in Fig 2, we describe $\mathbf{H}$ for the src domain. The intervals are arranged into a binary ($k = 2$) tree, as shown in Fig 4. The root is associated with the interval $[0**]$, which is evenly subdivided among its children. The unit-length intervals at the leaves are $[000], [001], [010], [011]$. The height of the tree is $\ell = 3$.*

The intervals of the tree are arranged into the query sequence $\mathbf{H} = \langle C_{0**}, C_{00*}, C_{01*}, C_{000}, C_{001}, C_{010}, C_{011} \rangle$. Evaluated on instance I from Fig. 2, the answer is $\mathbf{H}(I) = \langle 14, 2, 12, 2, 0, 10, 2 \rangle$.

To answer $\mathbf{H}$ under differential privacy, we must determine its sensitivity. As the following proposition shows, $\mathbf{H}$ has a larger sensitivity than $\mathbf{L}$.

PROPOSITION 4. *The sensitivity of $\mathbf{H}$ is ℓ.*

By Propositions 1 and 4, the following algorithm is ϵ-differentially private:

$$\tilde{\mathbf{H}}(I) = \mathbf{H}(I) + \langle \text{Lap}(\ell/\epsilon) \rangle^m$$

where m is the length of sequence $\mathbf{H}$, equal to the number of counts in the tree.

To answer a range query using $\tilde{\mathbf{H}}$, a natural strategy is to sum the fewest number of sub-intervals such that their union equals the desired range. However, one challenge with this approach is inconsistency: in the corresponding tree of noisy answers, there may be a parent count that does not equal to the sum of its children. This can be problematic: for example, an analyst might ask one interval query and then ask for a sub-interval and receive a *larger* count.

We next look at how to use the arithmetic constraints between parent and child counts (denoted $\gamma_{\mathbf{H}}$) to derive a consistent, and more accurate, estimate $\overline{\mathbf{H}}$.

4.1 Constrained Inference: Computing $\overline{\mathbf{H}}$

The analyst receives $\tilde{h} = \tilde{\mathbf{H}}(I)$, the noisy output from the differentially private algorithm $\tilde{\mathbf{H}}$. We now consider the problem of finding the minimum L_2 solution: to find the $\overline{h}$ that minimizes $||\tilde{h} - \overline{h}||_2$ and also satisfies the consistency constraints $\gamma_{\mathbf{H}}$.

This problem can be viewed as an instance of linear regression. The unknowns are the true counts of the unit-length intervals. Each answer in $\tilde{h}$ is a fixed linear combination of the unknowns, plus random noise. Finding $\overline{h}$ is equivalent to finding an estimate for the unit-length intervals. In fact, $\overline{h}$ is the familiar least squares solution.

Although the least squares solution can be computed via linear algebra, the hierarchical structure of this problem instance allows us to derive an intuitive closed form solution that is also more efficient to compute, requiring only linear time. Let T be the tree corresponding to $\tilde{h}$; abusing notation, for node $v \in T$, we write $\tilde{h}[v]$ to refer to the interval associated with v.

First, we define a possibly inconsistent estimate $z[v]$ for each node $v \in T$. The consistent estimate $\overline{h}[v]$ is then described in terms of the $z[v]$ estimates. $z[v]$ is defined recursively from the leaves to the root. Let l denote the height of node v and $succ(v)$ denote the set of v's children.

$$z[v] = \begin{cases} \tilde{h}[v], \text{ if v is a leaf node} \\ \frac{k^l - k^{l-1}}{k^l - 1}\tilde{h}[v] + \frac{k^{l-1}-1}{k^l-1}\sum_{u \in succ(v)} z[u], \text{ o.w.} \end{cases}$$

The intuition behind $z[v]$ is that it is a weighted average of two estimates for the count at v; in fact, the weights are inversely proportional to the variance of the estimates.

The consistent estimate $\overline{h}$ is defined recursively from the root to the leaves. At the root r, $\overline{h}[r]$ is simply $z[r]$. As we descend the tree, if at some node u, we have $\overline{h}[u] \neq \sum_{w \in succ(u)} z[w]$, then we adjust the values of each descendant by dividing the difference $\overline{h}[u] - \sum_{w \in succ(u)} z[w]$ equally among the k descendants. The following theorem states that this is the minimum L_2 solution.

THEOREM 3. *Given the noisy sequence $\tilde{h} = \tilde{\mathbf{H}}(I)$, the unique minimum L_2 solution, $\overline{h}$, is given by the following recurrence relation. Let u be v's parent:*

$$\overline{h}[v] = \begin{cases} z[v], & \text{if } v \text{ is the root} \\ z[v] + \frac{1}{k}(\overline{h}[u] - \sum_{w \in succ(u)} z[w]), & \text{o.w.} \end{cases}$$

Theorem 3 shows that the overhead of computing $\overline{\mathbf{H}}$ is minimal, requiring only two linear scans of the tree: a bottom up scan to compute z and then a top down scan to compute the solution $\overline{h}$ given z.

4.2 Utility Analysis: the Accuracy of $\overline{\mathbf{H}}$

We measure utility as accuracy of range queries, and we compare three strategies: $\tilde{\mathbf{L}}$, $\tilde{\mathbf{H}}$, and $\overline{\mathbf{H}}$. We start by comparing $\tilde{\mathbf{L}}$ and $\tilde{\mathbf{H}}$.

Given range query $q = c([x, y])$, we derive an estimate for the answer as follows. For $\tilde{\mathbf{L}}$, the estimate is simply the sum of the noisy unit-length intervals in the range: $\tilde{\mathbf{L}}_q = \sum_{i=x}^{y} \tilde{\mathbf{L}}[i]$. The error of each count is $2/\epsilon^2$, and so the error for the range is $error(\tilde{\mathbf{L}}_q) = O((y-x)/\epsilon^2)$.

For $\tilde{\mathbf{H}}$, we choose the natural strategy of summing the fewest sub-intervals of $\tilde{\mathbf{H}}$. Let $r_1, \ldots, r_t$ be the roots of disjoint subtrees of T such that the union of their ranges equals $[x, y]$. Then $\tilde{\mathbf{H}}_q$ is defined as $\tilde{\mathbf{H}}_q = \sum_{i=1}^{t} \tilde{\mathbf{H}}[r_i]$. Each noisy count has error equal to $2\ell^2/\epsilon^2$ (equal to the variance of the added noise) and the number of subtrees is at most 2ℓ (at most two per level of the tree), thus $error(\tilde{\mathbf{H}}_q) = O(\ell^3/\epsilon^2)$.

There is clearly a tradeoff between these two strategies. While $\tilde{\mathbf{L}}$ is accurate for small ranges, error grows linearly with the size of the range. In contrast, the error of $\tilde{\mathbf{H}}$ is poly-logarithmic in the size of the domain (recall that $\ell = \Theta(\log n)$). Thus, while $\tilde{\mathbf{H}}$ is less accurate for small ranges, it is much more accurate for large ranges. If the goal of a universal histogram is to bound worst-case or total error for all range queries, then $\tilde{\mathbf{H}}$ is the preferred strategy.

We now compare $\tilde{\mathbf{H}}$ to $\overline{\mathbf{H}}$. Since $\overline{\mathbf{H}}$ is consistent, range queries can be easily computed by summing the unit-length counts. In addition to being consistent, it is also more accurate. In fact, it is in some sense optimal: among the class of strategies that (a) produce unbiased estimates for range queries and (b) derive the estimate from linear combinations of the counts in $\tilde{h}$, there is no strategy with lower mean squared error than $\overline{\mathbf{H}}$.

THEOREM 4. *(i) $\overline{\mathbf{H}}$ is a linear unbiased estimator, (ii) $error(\overline{\mathbf{H}}_q) \leq error(E_q)$ for all q and for all linear unbiased estimators E, (iii) $error(\overline{\mathbf{H}}_q) = O(\ell^3/\epsilon^2)$ for all q, and (iv) there exists a query q s.t. $error(\overline{\mathbf{H}}_q) \leq \frac{3}{2(\ell-1)(k-1)-k} error(\tilde{\mathbf{H}}_q)$.*

Part (iv) of the theorem shows that $\overline{\mathbf{H}}$ can more accurate than $\tilde{\mathbf{H}}$ on some range queries. For example, in a height 16 binary tree—the kind of tree used in the experiments—there is a query q where $\overline{\mathbf{H}}_q$ is more accurate than $\tilde{\mathbf{H}}_q$ by a factor of $\frac{2(\ell-1)(k-1)-k}{3} = 9.33$.

Furthermore, the fact that $\overline{\mathbf{H}}$ is consistent can lead to additional advantages when the domain is sparse. We propose a simple extension to $\overline{\mathbf{H}}$: after computing $\overline{h}$, if there is a subtree rooted at v such that $\overline{h}[v] \leq 0$, we simply set the count at v and all children of v to be zero. This is a heuristic strategy; incorporating non-negativity constraints into inference is left for future work. Nevertheless, we show in experiments, that this small change can greatly reduce error in sparse regions and can lead to $\overline{\mathbf{H}}$ being more accurate than $\tilde{\mathbf{L}}$ even at small ranges.

5. EXPERIMENTS

We evaluate our techniques on three real datasets (explained in detail in Appendix C): `NetTrace` is derived from an IP-level network trace collected at a major university; `Social Network` is a graph derived from friendship relations in an online social network site; `Search Logs` is a dataset of search query logs over time from Jan. 1, 2004 to the present. Source code for the algorithms is available at the first author's website.

5.1 Unattributed Histograms

The first set of experiments evaluates the accuracy of constrained inference on unattributed histograms. We compare $\overline{\mathbf{S}}$ to the baseline approach $\tilde{\mathbf{S}}$. Since $\tilde{s} = \tilde{\mathbf{S}}(I)$ is likely to be inconsistent—out-of-order, non-integral, and possibly negative—we consider a second baseline technique, denoted $\tilde{\mathbf{S}}_{\mathbf{r}}$, which enforces consistency by sorting $\tilde{s}$ and rounding each count to the nearest non-negative integer.

We evaluate the performance of these estimators on three queries from the three datasets. On `NetTrace`: the query

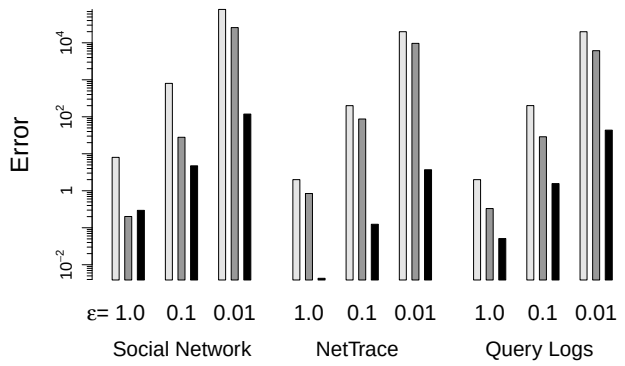


Figure 5: Error across varying datasets and ϵ. Each triplet of bars represents the three estimators: $\tilde{\mathbf{S}}$ (light gray), $\tilde{\mathbf{S}}_\mathbf{r}$ (gray), and $\overline{\mathbf{S}}$ (black).

returns the number of internal hosts to which each external host is connected ($\approx$ 65K external hosts); On `Social Network`, the query returns the degree sequence of the graph ($\approx$ 11K nodes). On `Search Logs`, the query returns the search frequency over a 3-month period of the top 20K keywords; position i in the answer vector is the number of times the i^{th} ranked keyword was searched.

To evaluate the utility of an estimator, we measure its squared error. Results report the average squared error over 50 random samples from the differentially-private mechanism (each sample produces a new $\tilde{s}$). We also show results for three settings of $\epsilon = \{1.0, 0.1, 0.01\}$; smaller ϵ means more privacy, hence more random noise.

Fig 5 shows the results of the experiment. Each bar represents average performance for a single combination of dataset, ϵ, and estimator. The bars represent, from left-to-right, $\tilde{\mathbf{S}}$ (light gray), $\tilde{\mathbf{S}}_\mathbf{r}$ (gray), and $\overline{\mathbf{S}}$ (black). The vertical axis is average squared error on a log-scale. The results indicate that the proposed approach reduces the error by at least an order of magnitude across all datasets and settings of ϵ. Also, the difference between $\tilde{\mathbf{S}}_\mathbf{r}$ and $\overline{\mathbf{S}}$ suggests that the improvement is due not simply to enforcing integrality and non-negativity but from the way consistency is enforced through constrained inference (though $\overline{\mathbf{S}}$ and $\tilde{\mathbf{S}}_\mathbf{r}$ are comparable on `Social Network` at large ϵ). Finally, the relative accuracy of $\overline{\mathbf{S}}$ improves with decreasing ϵ (more noise). Appendix C provides intuition for how $\overline{\mathbf{S}}$ reduces error.

5.2 Universal Histograms

We now evaluate the effectiveness of constrained inference for the more general task of computing a universal histogram and arbitrary range queries. We evaluate three techniques for supporting universal histograms $\tilde{\mathbf{L}}$, $\tilde{\mathbf{H}}$, and $\overline{\mathbf{H}}$. For all three approaches, we enforce integrality and non-negativity by rounding to the nearest non-negative integer. With $\overline{\mathbf{H}}$, rounding is done as part of the inference process, using the approach described in Sec 4.2.

We evaluate the accuracy over a set of range queries of varying size and location. The range sizes are 2^i for $i = 1, \ldots, \ell - 2$ where ℓ is the height of the tree. For each fixed size, we select the location uniformly at random. We report the average error over 50 random samples of $\tilde{l}$ and $\tilde{h}$, and for each sample, 1000 randomly chosen ranges.

We evaluate the following histogram queries: On `NetTrace`, the number of connections for each external host.

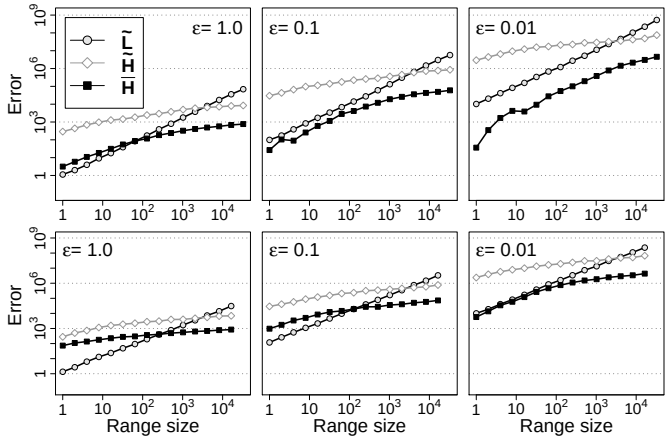


Figure 6: A comparison estimators $\tilde{\mathbf{L}}$ (circles), $\tilde{\mathbf{H}}$ (diamonds), and $\overline{\mathbf{H}}$ (squares) on two real-world datasets (top `NetTrace`, bottom `Search Logs`).

This is similar to the query in Sec 5.1 except that here the association between IP address and count is retained. On `Search Logs`, the query reports the temporal frequency of the query term "Obama" from Jan. 1, 2004 to present. (A day is evenly divided into 16 units of time.)

Fig 6 shows the results for both datasets and varying ϵ. The top row corresponds to `NetTrace`, the bottom to `Search Logs`. Within a row, each plot shows a different setting of $\epsilon \in \{1.0, 0.1, 0.01\}$. For all plots, the x-axis is the size of the range query, and the y-axis is the error, averaged over sampled counts and intervals. Both axes are in log-scale.

First, we compare $\tilde{\mathbf{L}}$ and $\tilde{\mathbf{H}}$. For unit-length ranges, $\tilde{\mathbf{L}}$ yields more accurate estimates. This is unsurprising since it is a lower sensitivity query and thus less noise is added for privacy. However, the error of $\tilde{\mathbf{L}}$ increases linearly with the size of the range. The average error of $\tilde{\mathbf{H}}$ increases slowly with the size of the range, as larger ranges typically require summing a greater number of subtrees. For ranges larger than about 2000 units, the error of $\tilde{\mathbf{L}}$ is higher than $\tilde{\mathbf{H}}$; for the largest ranges, the error of $\tilde{\mathbf{L}}$ is 4-8 times larger than that of $\tilde{\mathbf{H}}$ (note the log-scale).

Comparing $\overline{\mathbf{H}}$ against $\tilde{\mathbf{H}}$, the error of $\overline{\mathbf{H}}$ is uniformly lower across all range sizes, settings of ϵ, and datasets. The relative performance of the estimators depends on ϵ. At smaller ϵ, the estimates of $\overline{\mathbf{H}}$ are more accurate relative to $\tilde{\mathbf{H}}$ and $\tilde{\mathbf{L}}$. Recall that as ϵ decreases, noise increases. This suggests that the relative benefit of statistical inference increases with the uncertainty in the observed data.

Finally, the results show that $\overline{\mathbf{H}}$ can have lower error than $\tilde{\mathbf{L}}$ over small ranges, even for leaf counts. This may be surprising since for unit-length counts, the scale of the noise of $\overline{\mathbf{H}}$ is *larger* than that of $\tilde{\mathbf{L}}$ by a factor of $\log n$. The reduction in error is because these histograms are sparse. When the histogram contains sparse regions, $\overline{\mathbf{H}}$ can effectively identify them because it has noisy observations at higher levels of the tree. In contrast, $\tilde{\mathbf{L}}$ has only the leaf counts; thus, even if a range contains no records, $\tilde{\mathbf{L}}$ will assign a positive count to roughly half of the leaves in the range.

6. RELATED WORK

Dwork has written comprehensive reviews of differential privacy [6, 7]; below we highlight results closest to this work.

The idea of post-processing the output of a differentially private mechanism to ensure consistency was introduced in Barak et al. [1], who proposed a linear program for making a set of marginals consistent, non-negative, and integral. However, unlike the present work, the post-processing is not shown to improve accuracy.

Blum et al. [4] propose an efficient algorithm to publish synthetic data that is useful for range queries. In comparison with our hierarchical histogram, the technique of Blum et al. scales slightly better (logarithmic versus poly-logarithmic) in terms of domain size (with all else fixed). However, our hierarchical histogram achieves lower error for a fixed domain, and importantly, the error does not grow as the size of the database increases, whereas with Blum et al. it grows with $O(N^{2/3})$ with N being the number of records (details in Appendix E).

The present work first appeared as an arXiv preprint [12], and since then a number of related works have emerged, including additional work by the authors. The technique for unattributed histograms has been applied to accurately and efficiently estimate the degree sequences of large social networks [11]. Several techniques for histograms over hierarchical domains have been developed. Xiao et al. [20] propose an approach based on the Haar wavelet, which is conceptually similar to the **H** query in that it is based on a tree of queries where each level in the tree is an increasingly fine-grained summary of the data. In fact, that technique has error equivalent to a binary **H** query, as shown by Li et al. [14], who represent both techniques as applications of the matrix mechanism, a framework for computing workloads of linear counting queries under differential privacy. We are aware of ongoing work by McSherry et al. [16] that combines hierarchical querying with statistical inference, but differs from **H** in that it is adaptive. Chan et al. [5] consider the problem of continual release of aggregate statistics over streaming private data, and propose a differentially private counter that is similar to **H**, in which items are hierarchically aggregated by arrival time. The **H** and wavelet strategy are specifically designed to support range queries. Strategies for answering more general workloads of queries under differential privacy are emerging, in both the offline [10, 14] and online [18] settings.

7. CONCLUSIONS

Our results show that by transforming a differentially-private output so that it is consistent, we can boost accuracy. Part of the innovation is devising a query set so that useful constraints hold. Then the challenge is to apply the constraints by searching for the closest consistent solution. Our query strategies for histograms have closed-form solutions for efficiently computing a consistent answer.

Our results show that conventional differential privacy approaches can add more noise than is strictly required by the privacy condition. We believe that using constraints may be an important part of finding optimal strategies for query answering under differential privacy. More discussion of the implications of our results, and possible extensions, is included in Appendix B.

Acknowledgments. Hay was supported by the Air Force Research Laboratory (AFRL) and IARPA, under agreement number FA8750-07-2-0158. Hay and Miklau were supported by NSF CNS 0627642, NSF DUE-0830876, and NSF IIS 0643681. Rastogi and Suciu were supported by NSF IIS-0627585. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the AFRL and IARPA, or the U.S. Government.

8. 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] R. E. Barlow, D. J. Bartholomew, J. M. Bremner, and H. D. Brunk. *Statistical Inference Under Order Restrictions*. John Wiley and Sons Ltd, 1972.

[3] R. E. Barlow and H. D. Brunk. The isotonic regression problem and its dual. *JASA*, 67(337):140–147, 1972.

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

[5] T.-H. H. Chan, E. Shi, and D. Song. Private and continual release of statistics. In *ICALP*, 2010.

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

[7] C. Dwork. A firm foundation for private data analysis. *CACM, To appear*, 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, C. Li, G. Miklau, and D. Jensen. Accurate estimation of the degree distribution of private networks. In *ICDM*, 2009.

[12] M. Hay, V. Rastogi, G. Miklau, and D. Suciu. Boosting the accuracy of differentially-private queries through consistency. *CoRR*, abs/0904.0942, April 2009.

[13] J. T. G. Hwang and S. D. Peddada. Confidence interval estimation subject to order restrictions. *Annals of Statistics*, 1994.

[14] C. Li, M. Hay, V. Rastogi, G. Miklau, and A. McGregor. Optimizing histogram queries under differential privacy. In *PODS*, 2010.

[15] F. McSherry. Privacy integrated queries: An extensible platform for privacy-preserving data analysis. In *SIGMOD*, 2009.

[16] F. McSherry, K. Talwar, and O. Williams. Maximum likelihood data synthesis. Manuscript, 2009.

[17] M. E. J. Newman. The structure and function of complex networks. *SIAM Review*, 45(2):167–256, 2003.

[18] A. Roth and T. Roughgarden. Interactive privacy via the median mechanism. In *STOC*, 2010.

[19] S. D. Silvey. *Statistical Inference*. Chapman-Hall, 1975.

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

APPENDIX

A. NOTATIONAL CONVENTIONS

The table below summarizes notational conventions used in the paper.

$\mathbf{Q}$	Sequence of counting queries
$\mathbf{L}$	Unit-**L**ength query sequence
$\mathbf{H}$	**H**ierarchical query sequence
$\mathbf{S}$	**S**orted query sequence
$\gamma_{\mathbf{Q}}$	Constraint set for query $\mathbf{Q}$
$\tilde{\mathbf{Q}}, \tilde{\mathbf{L}}, \tilde{\mathbf{H}}, \tilde{\mathbf{S}}$	Randomized query sequence
$\overline{\mathbf{H}}, \overline{\mathbf{S}}$	Randomized query sequence, returning minimum L_2 solution
I	Private database instance
$\mathbf{L}(I), \mathbf{H}(I), \mathbf{S}(I)$	Output sequence (truth)
$\tilde{l} = \tilde{\mathbf{L}}(I), \tilde{h} = \tilde{\mathbf{H}}(I), \tilde{s} = \tilde{\mathbf{S}}(I)$	Output sequence (noisy)
$\overline{h} = \overline{\mathbf{H}}(I), \overline{s} = \overline{\mathbf{S}}(I)$	Output sequence (inferred)

B. DISCUSSION OF MAIN RESULTS

Here we provide a supplementary discussion of the results, review the insights gained, and discuss future directions.

Unattributed histograms. The choice of the sorted query $\mathbf{S}$, instead of $\mathbf{L}$, is an unqualified benefit, because we gain from the inequality constraints on the output, while the sensitivity of $\mathbf{S}$ is no greater than that of $\mathbf{L}$. Among other applications, this allows for extremely accurate estimation of degree sequences of a graph, improving error by an order of magnitude on the baseline technique. The accuracy of the estimate depends on the input sequence. It works best for sequences with duplicate counts, which matches well the degree sequences of social networks encountered in practice.

Future work specifically oriented towards degree sequence estimation could include a constraint enforcing that the output sequence is *graphical*, i.e. the degree sequence of some graph.

Universal histograms. The choice of the hierarchical counting query $\mathbf{H}$, instead of $\mathbf{L}$, offers a trade off because the sensitivity of $\mathbf{H}$ is greater than that of $\mathbf{L}$. It is interesting that for some data sets and privacy levels, the effect of the $\mathbf{H}$ constraints outweighs the increased noise that must be added. In other cases, the algorithms based on $\mathbf{H}$ provide greater accuracy for all but the smallest ranges. We note that in many practical settings, domains are large and sparse. The sparsity implies that no differentially private technique can yield meaningful answers for unit-length queries because the noise necessary for privacy will drown out the signal. So while $\tilde{\mathbf{L}}$ sometimes has higher accuracy for small range queries, this may not have practical relevance since the relative error of the answers renders them useless.

In future work we hope to extend the technique for universal histograms to multi-dimensional range queries, and to investigate optimizations such as higher branching factors.

Across both histogram tasks, our results clearly show that it is possible to achieve greater accuracy without sacrificing privacy. The existence of our improved estimators $\overline{\mathbf{S}}$ and $\overline{\mathbf{H}}$ show that there is another differentially private noise distribution that is more accurate than independent Laplace noise. This does not contradict existing results because the original differential privacy work showed only that calibrating Laplace noise to the sensitivity of a query was *sufficient* for privacy, not that it was necessary. Only recently has the optimality of this construction been studied (and proven only for single queries) [9]. Finding the optimal strategy for answering a set of queries under differential privacy is an important direction for future work, especially in light of emerging private query interfaces [15].

A natural goal is to describe directly the improved noise distributions implied by $\overline{\mathbf{S}}$ and $\overline{\mathbf{H}}$, and build a privacy mechanism that samples from it. This could, in theory, avoid the inference step altogether. But it is seems quite difficult to discover, describe, and sample these improved noise distributions, which will be highly dependent on a particular query of interest. Our approach suggests that constraints and constrained inference can be an effective path to discovering new, more accurate noise distributions that satisfy differential privacy. As a practical matter, our approach does not necessarily burden the analyst with the constrained inference process because the server can implement the post-processing step. In that case it would appear to the analyst as if the server was sampling from the improved distribution.

While our focus has been on histogram queries, the techniques are probably not limited to histograms and could have broader impact. However, a general formulation may be challenging to develop. There is a subtle relationship between constraints and sensitivity: reformulating a query so that it becomes highly constrained may similarly increase its sensitivity. A challenge is finding queries such as $\mathbf{S}$ and $\mathbf{H}$ that have useful constraints but remain low sensitivity. Another challenge is the computational efficiency of constrained inference, which is posed here as a constrained optimization problem with a quadratic objective function. The complexity of solving this problem will depend on the nature of the constraints and is NP-Hard in general. Our analysis shows that the constraint sets of $\mathbf{S}$ and $\mathbf{H}$ admit closed-form solutions that are efficient to compute.

C. ADDITIONAL EXPERIMENTS

This section provides detailed descriptions of the datasets, and additional results for unattributed histograms.

`NetTrace` is derived from an IP-level network trace collected at a major university. The trace monitors traffic at the gateway between internal IP addresses and external IP addresses. From this data, we derived a bipartite connection graph where the nodes are hosts, labeled by their IP address, and an edge connotes the transmission of at least one data packet. Here, differential privacy ensures that individual connections remain private.

`Social Network` is a graph derived from friendship relations on an online social network site. The graph is limited to a population of roughly $11,000$ students from a single university. Differential privacy implies that friendships will not be disclosed. The size of the graph (number of students) is assumed to be public knowledge.[1]

`Search Logs` is a dataset of search query logs over time from Jan. 1, 2004 to the present. For privacy reasons, it is difficult to obtain such data. Our dataset is derived from a search engine interface that publishes summary statistics for specified query terms. We combined these summary statistics with a second dataset, which contains actual search

[1]This is not a critical assumption and, in fact, the number of students can be estimated privately within $\pm 1/\epsilon$ in expectation. Our techniques can be applied directly to either the true count or a noisy estimate.

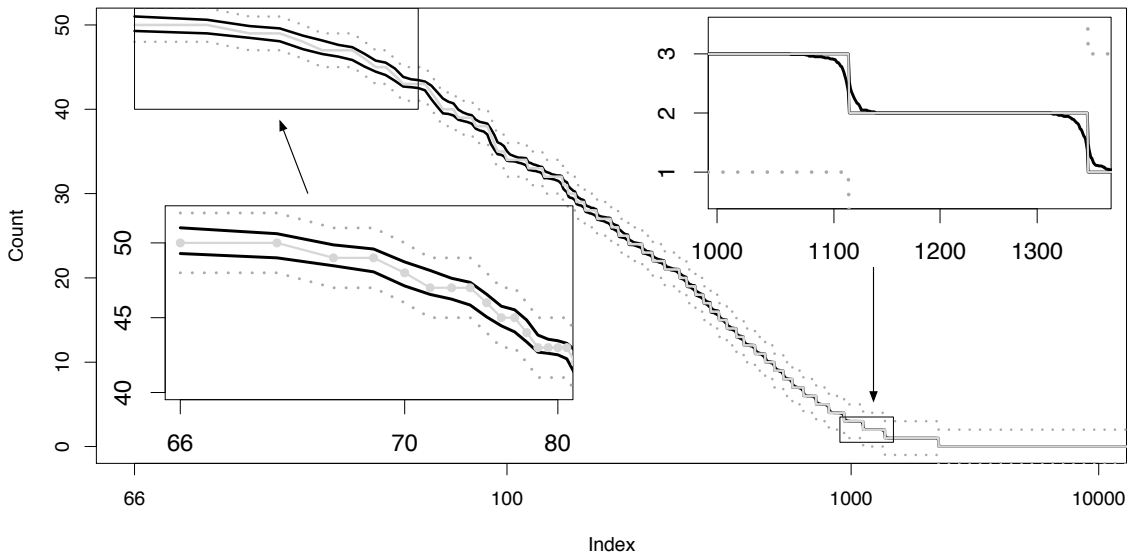


Figure 7: On NetTrace, $\mathbf{S}(I)$ (solid gray), the average error of $\overline{\mathbf{S}}$ (solid black) and $\tilde{\mathbf{S}}$ (dotted gray), for $\epsilon = 1.0$.

query logs but for a much shorter time period, to produce a synthetic data set. In the experiments, ground truth refers to this synthetic dataset. Differential privacy guarantees that the output will prevent the association of an individual entity (user, host) with a particular search term.

Unattributed histograms. Figure 7 provides some intuition for how inference is able to reduce error. Shown is a portion of the unattributed histogram of NetTrace: the sequence is sorted in *descending* order along the x-axis and the y-axis indicates the count. The solid gray line corresponds to ground truth: a long horizontal stretch indicates a subsequence of uniform counts and a vertical drop indicates a decrease in count. The graphic shows only the middle portion of the unattributed histogram—some very large and very small counts are omitted to improve legibility. The solid black lines indicate the error of $\overline{\mathbf{S}}$ averaged over 200 random samples of $\tilde{\mathbf{S}}$ (with $\epsilon = 1.0$); the dotted gray lines indicate the expected error of $\tilde{\mathbf{S}}$.

The inset graph on the left reveals larger error at the beginning of the sequence, when each count occurs once or only a few times. However, as the counts become more concentrated (longer subsequences of uniform count), the error diminishes, as shown in the right inset. Some error remains around the points in the sequence where the count changes, but the error is reduced to zero for positions in the middle of uniform subsequences.

Figure 7 illustrates that our approach reduces or eliminates noise in precisely the parts of the sequence where the noise is unnecessary for privacy. Changing a tuple in the database cannot change a count in the middle of a uniform subsequence, only at the end points. These experimental results also align with Theorem 2, which states that the error of $\overline{\mathbf{S}}$ is a function of the number of distinct counts in the sequence. In fact, the experimental results suggest that the theorem also holds locally for subsequences with a small number of distinct counts. This is an important result since the typical degree sequences that arise in real data, such as the power-law distribution, contain very large uniform subsequences.

D. PROOFS

Proof of Proposition 2. For any output $\overline{q}$, then let $\mathcal{S}(\overline{q})$ denote the set of noisy answers such that if $\tilde{q} \in \mathcal{S}(\overline{q})$ then the minimum L_2 solution given $\tilde{q}$ and $\gamma_{\mathbf{Q}}$ is $\overline{q}$. For any I and $I' \in nbrs(I)$, the following shows that $\overline{\mathbf{Q}}$ is ϵ-differentially private:

$$\begin{aligned} Pr[\overline{\mathbf{Q}}(I) = \overline{q}] &= Pr[\tilde{\mathbf{Q}}(I) \in \mathcal{S}(\overline{q})] \\ &\leq \exp(\epsilon)\ Pr[\tilde{\mathbf{Q}}(I') \in \mathcal{S}(\overline{q})] \\ &= \exp(\epsilon)\ Pr[\overline{\mathbf{Q}}(I') = \overline{q}] \end{aligned}$$

where the inequality is because $\tilde{\mathbf{Q}}$ is ϵ-differentially private. □

Proof of Proposition 3. Given a database I, suppose we add a record to it to obtain I'. The added record affects one count in $\mathbf{L}$, i.e., there is exactly one i such that $\mathbf{L}(I)[i] = x$ and $\mathbf{L}(I')[i] = x+1$, and all other counts are the same. The added record affects $\mathbf{S}$ as follows. Let j be the largest index such that $\mathbf{S}(I)[j] = x$, then the added record increases the count at j by one: $\mathbf{S}(I')[j] = x+1$. Notice that this change does not affect the sort order—i.e., in $\mathbf{S}(I')$, the j^{th} value remains in sorted order: $\mathbf{S}(I')[j-1] \leq x$, $\mathbf{S}(I')[j] = x+1$, and $\mathbf{S}(I')[j+1] \geq x+1$. All other counts in $\mathbf{S}$ are the same, and thus the L_1 distance between $\mathbf{S}(I)$ and $\mathbf{S}(I')$ is 1. □

Proof of Proposition 4. If a tuple is added or removed from the relation, this affects the count for every range that includes it. There are exactly ℓ ranges that include a given tuple: the range of a single leaf and the ranges of the nodes along the path from that leaf to the root. Therefore, adding/removing a tuple changes exactly ℓ counts each by exactly 1. Thus, the sensitivity is equal to ℓ, the height of the tree. □

D.1 Proofs of Theorems 1 & 2

These proofs are omitted due to space limitations, but can be found in the full version [12].

D.2 Proof of Theorem 3

We first restate the theorem below.

THEOREM 3. *Given the noisy sequence $\tilde{h} = \tilde{\mathbf{H}}(I)$, the unique minimum L_2 solution, $\overline{h}$, is given by the following recurrence relation. Let u be v's parent:*

$$\overline{h}[v] = \begin{cases} z[v], & \text{if } v \text{ is the root} \\ z[v] + \frac{1}{k}(\overline{h}[u] - \sum_{w \in succ(u)} z[w]), & o.w. \end{cases}$$

PROOF. We first show that $\overline{h}[r] = z[r]$ for the root node r. By definition of a minimum L_2 solution, the sequence $\overline{h}$ satisfies the following constrained optimization problem. Let $succZ[u] = \sum_{w \in succ(u)} z[w]$.

$$\text{minimize } \sum_v (\overline{h}[v] - \tilde{h}[v])^2$$
$$\text{subject to } \forall v, \sum_{u \in succ(v)} \overline{h}[u] = \overline{h}[v]$$

Denote $leaves(v)$ to be the set of leaf nodes in the subtree rooted at v. The above optimization problem can be rewritten as the following unconstrained minimization problem.

$$\text{minimize } \sum_v \big((\sum_{l \in leaves(v)} \overline{h}[l]) - \tilde{h}[v] \big)^2$$

For finding the minimum, we take derivative w.r.t $\overline{h}[l]$ for each l and equate it to 0. We thus get the following set of equations for the minimum solution.

$$\forall l, \sum_{v: l \in leaves(v)} 2\big((\sum_{l' \in leaves(v)} \overline{h}[l']) - \tilde{h}[v] \big) = 0$$

Since $\sum_{l' \in leaves(v)} \overline{h}[l'] = \overline{h}[v]$, the above set of equations can be rewritten as: $\forall l, \sum_{v: l \in leaves(v)} \overline{h}[v] = \sum_{v: l \in leaves(v)} \tilde{h}[v]$

For a leaf node l, we can think of the above equation for l as corresponding to a path from l to the root r of the tree. The equation states that sum of the sequences $\overline{h}$ and $\tilde{h}$ over the nodes along the path are the same. We can sum all the equations to obtain the following equation.

$$\sum_v \sum_{l \in leaves(v)} \overline{h}[v] = \sum_v \sum_{l \in leaves(v)} \tilde{h}[v]$$

Denote $level(i)$ as the set of nodes at height i of the tree. Thus root belongs to $level(\ell - 1)$ and leaves in $level(0)$. Abbreviating LHS (RHS) for the left (right) hand side of the above equation, we observe the following.

$$\begin{aligned} LHS &= \sum_v \sum_{l \in leaves(v)} \overline{h}[v] \\ &= \sum_{i=0}^{\ell-1} \sum_{v \in level(i)} \sum_{l \in leaves(v)} \overline{h}[v] \\ &= \sum_{i=0}^{\ell-1} \sum_{v \in level(i)} k^i \overline{h}[v] = \sum_{i=0}^{\ell-1} k^i \sum_{v \in level(i)} \overline{h}[v] \\ &= \sum_{i=0}^{\ell-1} k^i \overline{h}[r] = \frac{k^\ell - 1}{k - 1} \overline{h}[r] \end{aligned}$$

Here we use the fact that $\sum_{v \in level(i)} \overline{h}[v] = \overline{h}[r]$ for any level i. This is because $\overline{h}$ satisfies the constraints of the tree. In a similar way, we also simplify the RHS.

$$\begin{aligned} RHS &= \sum_v \sum_{l \in leaves(v)} \tilde{h}[v] \\ &= \sum_{i=0}^{\ell-1} \sum_{v \in level(i)} \sum_{l \in leaves(v)} \tilde{h}[v] \\ &= \sum_{i=0}^{\ell-1} \sum_{v \in level(i)} k^i \tilde{h}[v] = \sum_{i=0}^{\ell-1} k^i \sum_{v \in level(i)} \tilde{h}[v] \end{aligned}$$

Note that we cannot simplify the RHS further as $\tilde{h}[v]$ may not satisfy the constraints of the tree. Finally equating LHS and RHS we get the following equation.

$$\overline{h}[r] = \frac{k-1}{k^\ell - 1} \sum_{i=0}^{\ell-1} k^i \sum_{v \in level(i)} \tilde{h}[v]$$

Further, it is easy to expand $z[r]$ and check that

$$z[r] = \frac{k-1}{k^\ell - 1} \sum_{i=0}^{\ell-1} k^i \sum_{v \in level(i)} \tilde{h}[v]$$

Thus we get $\overline{h}[r] = z[r]$. For nodes v other than the r, assume that we have computed $\overline{h}[u]$ for $u = pred(v)$. Denote $H = \overline{h}[u]$. Once H is fixed, we can argue that the value of $\overline{h}[v]$ will be independent of the values of $\tilde{h}[w]$ for any w not in the subtree of u.

For nodes $w \in subtree(u)$ the L_2 minimization problem is equivalent to the following one.

$$\text{minimize } \sum_{w \in subtree(u)} (\overline{h}[w] - \tilde{h}[w])^2$$
$$\text{subject to } \forall w \in subtree(u), \sum_{w' \in succ(w)} \overline{h}[w'] = \overline{h}[w]$$
$$\text{and } \sum_{v \in succ(u)} \overline{h}[v] = H$$

Again using nodes $l \in leaves(u)$, we convert this minimization into the following one.

$$\text{minimize } \sum_{w \in subtree(U)} \big((\sum_{l \in leaves(w)} \overline{h}[l]) - \tilde{h}[w] \big)^2$$
$$\text{subject to } \sum_{l \in leaves(u)} \overline{h}[u] = H$$

We can now use the method of Lagrange multipliers to find the solution of the above constrained minimization problem. Using λ as the Lagrange parameter for the constraint $\sum_{l \in leaves(u)} \overline{h}[u] = H$, we get the following sets of equations.

$$\forall l \in leaves(u), \sum_{w: l \in leaves(w)} 2\big(\overline{h}[w] - \tilde{h}[w]\big) = -\lambda$$

Adding the equations for all $l \in leaves(u)$ and solving for λ we get $\lambda = -\frac{H - succZ[u]}{n(u) - 1}$. Here $n(u)$ is the number of nodes in $subtree(u)$. Finally adding the above equations for only leaf nodes $l \in leaves(v)$, we get

$$\begin{aligned}\overline{h}[v] &= z[v] - (n(v)-1)\cdot\lambda \\ &= z[v] + \frac{n(v)-1}{n(u)-1}(H - succZ[u]]) \\ &= z[v] + \frac{1}{k}(\overline{h}[u] - succZ[u])\end{aligned}$$

This completes the proof. □

D.3 Proof of Theorem 4

First, the theorem is restated.

THEOREM 4. *(i) $\overline{\mathbf{H}}$ is a linear unbiased estimator, (ii) $error(\overline{\mathbf{H}}_q) \leq error(E_q)$ for all q and for all linear unbiased estimators E, (iii) $error(\overline{\mathbf{H}}_q) = O(\ell^3/\epsilon^2)$ for all q, and (iv) there exists a query q s.t. $error(\overline{\mathbf{H}}_q) \leq \frac{3}{2(\ell-1)(k-1)-k} error(\tilde{\mathbf{H}}_q)$.*

PROOF. For (i), the linearity of $\overline{\mathbf{H}}$ is obvious from the definition of z and $\overline{h}$. To show $\overline{\mathbf{H}}$ is unbiased, we first show that z is unbiased, i.e. $\mathbb{E}(z[v]) = h[v]$. We use induction: the base case is if v is a leaf node in which case $\mathbb{E}(z[v]) = \mathbb{E}(\tilde{h}[v]) = h[v]$. If v is not a leaf node, assume that we have shown z is unbiased for all nodes $u \in succ(v)$. Thus

$$\mathbb{E}(succZ[v]) = \sum_{u \in succ(v)} \mathbb{E}(z[u]) = \sum_{u \in succ(v)} h[u] = h[v]$$

Thus $succZ[v]$ is an unbiased estimator for $h[v]$. Since $z[v]$ is a linear combination of $\tilde{h}[v]$ and $succZ[v]$, which are both unbiased estimators, $z[v]$ is also unbiased. This completes the induction step proving that z is unbiased for all nodes. Finally, we note that $\overline{h}[v]$ is a linear combination of $\tilde{h}[v]$, $z[v]$, and $succZ[v]$, all of which are unbiased estimators. Thus $\overline{h}[v]$ is also unbiased proving (i).

For (ii), we shall use the Gauss-Markov theorem [19]. We shall treat the sequence $\tilde{h}$ as the set of observed variables, and l, the sequence of original leaf counts, as the set of unobserved variables. It is easy to see that for all nodes v

$$\tilde{h}[v] = \sum_{u \in leaves(v)} l[u] + noise(v)$$

Here $noise(v)$ is the Laplacian random variable, which is independent for different nodes v, but has the same variance for all nodes. Hence $\tilde{h}$ satisfies the hypothesis of Gauss-Markov theorem. (i) shows that $\overline{h}$ is a linear unbiased estimator. Further, $\overline{h}$ has been obtained by minimizing the L_2 distance with $\tilde{h}[v]$. Hence, $\overline{h}$ is the Ordinary Least Squares (OLS) estimator, which by the Gauss-Markov theorem has the least *error*. Since it is the OLS estimator, it minimizes the *error* for estimating any linear combination of the original counts, which includes in particular the given range query q.

For (iii), we note that any query q can be answered by summing at most $k\ell$ nodes in the tree. Since for any node v, $error(\overline{\mathbf{H}}[v]) \leq error(\tilde{\mathbf{H}}[v]) = 2\ell^2/\epsilon^2$, we get

$$error(\overline{\mathbf{H}}[q]) \leq k\ell(2\ell^2/\epsilon^2) = O(\ell^3/\epsilon^2)$$

For (iv), denote l_1 and l_2 to be the leftmost and rightmost leaf nodes in the tree. Denote r to be the root. We consider the query q that asks for the sum of all leaf nodes except for l_1 and l_2. Then from (i) $error(\overline{\mathbf{H}}(q))$ is less than the *error* of the estimate $\tilde{h}[r] - \tilde{h}[l_1] - \tilde{h}[l_2]$, which is $6\ell^2/\epsilon^2$. But, on the other hand, $\tilde{\mathbf{H}}$ will require summing $2(k-1)(\ell-1)-k$ noisy counts in total—$2(k-1)$ at each level of the tree, except for the root and the level just below the root, only $k-2$ counts are summed. Thus $error(\tilde{\mathbf{H}}_q) = 2(2(k-1)(\ell-1)-k)\ell^2/\epsilon^2$. Thus

$$error(\overline{\mathbf{H}}_q) \leq \frac{3error(\tilde{\mathbf{H}}_q)}{2(\ell-1)(k-1)-k}$$

This completes the proof. □

E. COMPARISON WITH BLUM ET AL.

We compare a binary $\tilde{\mathbf{H}}_q$ against the binary search equi-depth histogram of Blum et al. [4] in terms of (ϵ, δ)-usefulness as defined by Blum et al. Since ϵ is used in the usefulness definition, we use α as the parameter for α-differential privacy.

Let N be the number of records in the database. An algorithm is (ϵ, δ)-useful for a class of queries if with probability at least $1-\delta$, for every query in the class, the absolute error is at most ϵN.

For any range query q, the absolute error of $\tilde{\mathbf{H}}_q$ is $|\tilde{\mathbf{H}}_q(I) - \mathbf{H}_q(I)| = |Y|$ where $Y = \sum_{i=1}^{c} \gamma_i$, each $\gamma_i \sim Lap(\ell/\alpha)$, and c is the number of subtrees in $\tilde{\mathbf{H}}_q$, which is at most 2ℓ. We use Corollary 1 from [5] to bound the error of a sum of Laplace random variables. With $\nu = \sqrt{c\ell^2/\alpha^2}\sqrt{2\ln\frac{2}{\delta'}}$, we obtain

$$Pr\left[|Y| \leq \frac{16\ell^{\frac{3}{2}}\ln\frac{2}{\delta'}}{\alpha}\right] \geq 1-\delta'$$

The above is for a single range query. To bound the error for all $\binom{n}{2}$ range queries, we use a union bound. Set $\delta' = \frac{\delta}{n^2}$. Then $\tilde{\mathbf{H}}$ is (ϵ, δ)-useful provided that $\epsilon \geq \left(16\ell^{\frac{3}{2}}\ln\frac{2n^2}{\delta}\right)/\alpha$. As in Blum et al., we can also fix ϵ and bound the size of the database. $\tilde{\mathbf{H}}$ is (ϵ, δ)-useful when

$$N \geq \frac{16\ell^{\frac{3}{2}}\ln\frac{2n^2}{\delta}}{\alpha\epsilon} = O\left(\frac{\log^{\frac{3}{2}} n \left(\log n + \log\frac{1}{\delta}\right)}{\alpha\epsilon}\right)$$

In comparison, the technique of Blum et al. is (ϵ, δ)-useful for range queries when

$$N \geq O\left(\frac{\log n\left(\log\log n + \log\frac{1}{\epsilon\delta}\right)}{\alpha\epsilon^3}\right)$$

Both techniques scale at most poly-logarithmically with the size of the domain. However, the $\tilde{\mathbf{H}}$ scales better with ϵ, achieving the same utility guarantee with a database that is smaller by a factor of $O(1/\epsilon^2)$.

The above comparison reveals a distinction between the techniques: for $\tilde{\mathbf{H}}_q$ the bound on absolute error is independent of database size, i.e., it only depends on ϵ, α, and the size of range. However, for the Blum et al. approach, the absolute error increases with the size of the database at a rate of $O(N^{2/3})$.