arXiv:0904.0942v2 [cs.DB] 7 Apr 2009

Boosting the Accuracy of Differentially-Private Queries Through Consistency

Michael Hay
University of Massachusetts Amherst
mhay@cs.umass.edu

Vibhor Rastogi
University of Washington
vibhor@cs.washington.edu

Gerome Miklau
University of Massachusetts Amherst
miklau@cs.umass.edu

Dan Suciu
University of Washington
suciu@cs.washington.edu

ABSTRACT

Recent differentially private query mechanisms offer strong privacy guarantees by adding noise to the query answer. For a single counting query, the technique is simple, accurate, and provides optimal utility. However, analysts typically wish to ask multiple queries. In this case, the optimal strategy is not apparent, and alternative query strategies can involve difficult trade-offs in accuracy, and may produce inconsistent answers.

In this work we show that it is possible to significantly improve accuracy for a general class of histogram queries. 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 both private and consistent, but in addition, it is often much more accurate.

We apply our techniques to real datasets and show they can be used for estimating the degree sequence of a graph with extreme precision, and for computing a histogram that can support arbitrary range queries accurately.

1. INTRODUCTION

Recent work in differential privacy [10] 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. The noise is carefully calibrated to the *sensitivity* of the query to ensure privacy. Informally, the sensitivity of a query measures the total change to the query output under small changes to the input database. This query mechanism is simple, efficient, and—for many queries—quite accurate. In addition, for a single query, this standard mechanism has also recently been shown to be optimal in the sense that there is no better noisy answer to return under the desired privacy objective [11].

However, analysts typically need to compute multiple statistics on a database. Differentially private algorithms extend nicely to sets 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 the following quantities: the total number of students, x_t, the number of students receiving each possible grade A, B, C, D, and F (x_A, x_B, x_C, x_D, x_F), as well as the number of passing students, x_p (those receiving at least grade D). When exact answers are available, handling this workload of queries is straightforward. The analyst can simply submit all the queries, or can easily find a subset of queries from which the rest of the answers can be computed.

When using a differentially private interface, more than one strategy is possible. 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 x_t and x_p by summation. The sensitivity of this set of queries is one because adding or removing one tuple can change at most one of the five numbers by a value of one. The noise added to individual queries is therefore low. Unfortunately, the noise accumulates under summation, so the estimates for x_t and x_p are poor.

A second alternative is to request noisy answers for all quantities (x_t, x_p, x_A, x_B, x_C, x_D, x_F). This query has sensitivity 3 (one change can affect 3 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. These query results are related, and we expect the following constraints to hold for any consistent answer to the queries: $x_p = x_A + x_B + x_C + x_D$ and $x_t = x_p + x_F$. Receiving noisy answers that violate these constraints is problematic. For example, x_t and $x_p + x_F$ may be different estimates for the total number of students, and the analyst must find a way to reconcile them.

We propose techniques for resolving inconsistency in a set of queries, 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.

1.1 Overview of Approach

In this paper we propose novel techniques for deriving private but accurate answers to two related histogram tasks, to be described shortly. Our approach to private query answering is the following. First, we choose a set of queries where constraints naturally hold among the query answers. Second, using a standard differentially private algorithm, we derive a set of noisy answers. If the true answer set is x, we write the noisy answer set $\widetilde{x}$. The noisy answer set typically violates the constraints. So, third, we introduce a *constrained inference* step that computes a new set of answers $\overline{x}$ that satisfy the constraints and is closest to $\widetilde{x}$. The inferred answers $\overline{x}$ are differentially private and consistent, and often significantly more accurate than the original output.

The increase in accuracy we achieve depends on the input database and the privacy parameters. Intuitively, there may be some databases and levels of noise for which the perturbation tends not to produce answers that violate the constraints. In this case the inference step cannot help. But we show that our inference process never hurts: $\overline{x}$ is never a worse estimate for the true answer than $\widetilde{x}$ (in expectation). In practice, we find that many real datasets have data distributions for which our techniques offer significant accuracy improvement.

We emphasize that the improvement in accuracy is achieved with no sacrifice of privacy. The constrained inference step is applied by the analyst *after* receiving the output of a differentially private algorithm. In addition, the consistency constraints used in the inference step are derived from properties of the queries themselves and cannot leak information about the private database instance. Since the inference step uses no information from the database other than $\widetilde{x}$, it has no impact on the privacy guarantee. Thus, all techniques proposed in this paper satisfy standard ϵ-differential privacy [9].

Computing consistent answers from differentially private outputs has been discussed before [5] for the special case of contingency tables, although accuracy is not improved. We are able to exploit a larger set of constraints, and more general constraints, which offers a boost in accuracy.

1.2 Histogram tasks

We focus on two tasks related to histograms that illustrate how our inference technique can be applied to diverse constraints. 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 by reference to sample data that will serve as a running example throughout the paper, and is also the basis of later experiments. The relation $R(src, dst)$, illustrated in Fig. 1, 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 from internal to external addresses.

Unattributed histograms. In an unattributed histogram, we form disjoint intervals for the range attribute and compute counting queries for each of the specified ranges. The intervals themselves are irrelevant to the analysis and so we report only a multiset of frequencies.

In our example, we use *src* as the range attribute. There are four source addresses $000, 001, 010, 011$ present in the table. If we ask for counts of all unit-length ranges, then the unattributed histogram is simply the sequence of (out) degrees of the source addresses (in the example: $\langle 2, 0, 10, 2 \rangle$).

Table 1: Notational conventions

$\mathbf{L}$	Unit-**L**ength query sequence
$\mathbf{H}$	**H**ierarchical query sequence
$\mathbf{S}$	**S**orted query sequence
$\tilde{\mathbf{L}}, \tilde{\mathbf{H}}, \tilde{\mathbf{S}}$	Randomized query sequence
$\overline{\mathbf{H}}, \overline{\mathbf{S}}$	Randomized query sequence, returning $\textsc{min}L_2$
$\mathbf{L}(I), \mathbf{H}(I), \mathbf{S}(I)$	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)

Degree sequences are an important instance of an unattributed histogram because they are a crucial measure of a graph that is widely studied [16]. If the tuples of R record queries submitted to a search engine, and A is the search term, then the unattributed histogram shows the frequency of occurrence of all terms (but not the terms themselves), and can be used to study the distribution.

Universal histograms. We also 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 (the counts from source addresses 010 and 011 are 10 and 2, respectively), 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).

These two tasks can enable a wide range of analyses. We note that a universal histogram is more general than an unattributed histogram—the former can be used to derive the latter. But we will see that unattributed histograms can be evaluated much more accurately than would be possible using a universal histogram.

1.3 Queries and constraints for histogram tasks

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:

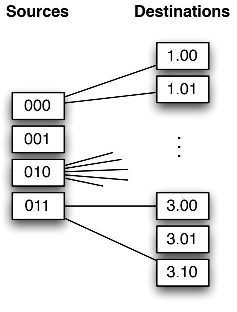
$$c([x, y]) = \mathsf{Select\ count(*)\ From\ R\ Where\ } x \leq \mathsf{R.A} \leq y$$

Our notational conventions are summarized in Table 1. When the query sequence $\mathbf{Q}$ is evaluated on a database instance I, the output, $\mathbf{Q}(I)$, includes one answer to each counting query, that is, $\mathbf{Q}(I)$ is a vector of non-negative integers. The i^{th} counting query in $\mathbf{Q}$ is $\mathbf{Q}[i]$.

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

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

In the example, we assume the domain of *src* contains



(a) Trace data

Query Definitions

$\mathbf{L}$: $\langle C_{000}, C_{001}, C_{010}, C_{011}\rangle$
$\mathbf{H}$: $\langle C_{0**}, C_{00*}, C_{01*}, C_{000}, C_{001}, C_{010}, C_{011}\rangle$
$\mathbf{S}$: $\langle rank_1(U) \ldots rank_n(U)\rangle$ for $U = \{c([x_j]) | x_j \in dom\}$

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

(b) Query variations

Figure 1: (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.

just the 4 addresses present in the table. So we have query sequence $\mathbf{L}$ defined to be $\langle c([000]), c([001]), c([010]), c([011])\rangle$ and $\mathbf{L}(I) = \langle 2, 0, 10, 2\rangle$.

This query implicitly reflects the association between a count and the range for that count, which is not relevant in an unattributed histogram. This extra information can be ignored. But our observation is that, 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 define a new query sequence $\mathbf{S}$, returning the permutation of $\mathbf{L}$ that is in ascending order. If U is the set of all unit length counts $U = \{c([x_j]) | x_j \in dom\}$, we write $rank_i(U)$ to refer to the i^{th} element in sorted, ascending order. We define $\mathbf{S} = \langle rank_1(U) \ldots rank_n(U)\rangle$. In the example, we have $\mathbf{L}(I) = \langle 2, 0, 10, 2\rangle$ while $\mathbf{S}(I) = \langle 0, 2, 2, 10\rangle$.

This small change does not impact privacy because the sensitivity is not changed by ordering, as we will show. But it has a significant impact on accuracy because $\mathbf{S}$ has a powerful set of constraints. We denote the constraint set $\gamma_{\mathbf{S}}$, which consists of all the inequalities $\mathbf{S}[i] < \mathbf{S}[i+1]$ for $1 \leq i < n$. (Note that the constraint set $\gamma_{\mathbf{L}}$ for $\mathbf{L}$ is empty.)

Our first set of results, given in Section 3, include a differentially private algorithm $\tilde{\mathbf{S}}$ and techniques for inferring a more accurate ordered result from its unordered noisy output.

Universal histograms. The query sequence $\mathbf{L}$ described above is the conventional strategy for computing a universal histogram as well. Since it returns all unit-length counts, the result can be used to compute any interval count, $c([x, y])$, by summation. This strategy works well only for small ranges, because the error accumulates as noisy estimates are added. For large ranges, the estimates can easily become useless.

We propose a different query sequence for supporting universal histograms, which consists of hierarchical intervals. The first interval is the entire domain $[x_1, x_n]$. We recursively partitioning so that the next intervals are $[x_1, x_{n/2}]$ and $[x_{n/2+1}, x_n]$, and so on. The new query sequence, denoted $\mathbf{H}$, consists of counting queries for *all* of the $2n-1$ hierarchical intervals (including the n unit-length intervals in $\mathbf{L}$). In the example, $\mathbf{H}$ is a query sequence with 7 elements, beginning with the total count, then a count for each half of the *src* address space, and then a count for each individual address. The complete description of $\mathbf{H}$ is shown in Fig. 1 and we have $\mathbf{H}(I) = \langle 14, 2, 12, 2, 0, 10, 2\rangle$.

The query sequence $\mathbf{H}$ has more counts than $\mathbf{L}$ and greater sensitivity. But it also has a large set of arithmetic constraints, denoted denoted $\gamma_{\mathbf{H}}$, which follow from the relationships between the intervals. For example, we know the following must hold: $c([0**]) = c([00*]) + c([01*])$ and $c([00*]) = c([000]) + c([001])$ and $c([01*]) = c([010]) + c([011])$.

Our second set of results, given in Section 4, include a differentially private algorithm $\tilde{\mathbf{H}}$ and techniques for using its inconsistent output to infer a more accurate consistent result.

Contributions

We propose differentially private algorithms to support unattributed and universal histograms. Our specific contributions include:

- For each task, we provide efficient, closed-form expressions for computing the *consistent* query answer closest to an observed randomized output.
- For each task, we analyze the inferred output theoretically, showing that it never increases error, and in some cases, that it provides an optimal estimate given the input.
- We demonstrate significant improvements in accuracy through experiments on real data sets. Degree sequence estimation is 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 about 45-98%, and improves on all ranges in some cases.

Our results reveal that common differential privacy approaches can introduce more noise than is strictly required by the privacy condition. Exploiting constraints through the proposed inference process is an important method for avoiding this.

2. BACKGROUND

In this section we review differential privacy, establish notational conventions, and formalize our constrained inference process as a minimization problem.

2.1 Differential Privacy

Informally, an algorithm is differentially private if its output is insensitive to small changes in the input. This provides privacy because if similar databases, say differing by a single record, produce indistinguishable outputs, then the adversary cannot use the output to infer the presence of individual records. For database I, let $nbrs(I)$ be the set of neighboring instances, which differ in exactly one tuple: i.e. such that $|(I - I') \cup (I' - I)| = 1$.

Definition 2.1 (ϵ-differential privacy [9]). *An 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:*

$$Pr[A(I) \in S] \leq e^{\epsilon} Pr[A(I') \in S]$$

where Pr is a probability distribution over the randomness of the algorithm.

Dwork et al. [10] present a technique for approximating the answer to any query in a differentially private way. It works by adding random noise to the answer, where the noise distribution is carefully calibrated to the query. The calibration depends on the query *sensitivity*—informally, the maximum amount the query answer can change given any small change to the database. We adapt the following 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 the smallest number such that for all I and $I' \in nbrs(I)$,*

$$\left|\mathbf{Q}(I) - \mathbf{Q}(I')\right| \leq S_{\mathbf{Q}}$$

Since $\mathbf{Q}$ is a query sequence consisting of counting queries, $\mathbf{Q}(I)$ and $\mathbf{Q}(I')$ are each vectors non-negative integers. In the definition above, $|\mathbf{Q}(I) - \mathbf{Q}(I')|$ refers to the L_1 distance between these vectors.

For a query sequence $\mathbf{Q}$ of length d, the following algorithm is shown to be ϵ-differentially private [10] :

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

Here $\langle \text{Lap}(X) \rangle^d$ is used to denote a vector of size d consisting of independent random samples, each taken from a zero-mean Laplace distribution. In the algorithm above, the scale of the Laplace distribution, $S_{\mathbf{Q}}/\epsilon$, is determined by the sensitivity of $\mathbf{Q}$.

Thus, $\tilde{\mathbf{Q}}$ denotes a randomized algorithm that computes the true answer to the query, $\mathbf{Q}(I)$, and adds Laplacian noise independently to each component. We use this technique throughout the paper to construct differentially private algorithms for query sequences.

We briefly review how these existing results are applied to the query sequence $\mathbf{L}$ that is the conventional technique for both unattributed and universal histograms. Recall that $\mathbf{L}$ returns counts for each unit interval in *dom*, the domain of the range attribute, and that $|dom| = n$:

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

The sensitivity of $\mathbf{L}$ is 1 because if two instances differ by one tuple, the result of computing $\mathbf{L}$ differs in exactly one position by a value of one. Therefore, the following algorithm is ϵ-differentially private:

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

Analyzing accuracy. To analyze the accuracy of the randomized query sequences proposed in this paper we quantify their error. A randomized query like $\tilde{\mathbf{Q}}$ is computed using $\mathbf{Q}(I)$ as input. Therefore, $\tilde{\mathbf{Q}}$ can be considered an estimator for the true value $\mathbf{Q}(I)$. Recall that the L_2 distance of two vectors X and Y is defined as $\parallel X - Y \parallel_2 = \sum_i |x_i - y_i|^2$. We use the common Mean Squared Error as a measure accuracy.

Definition 2.3 (Utility). *For a randomized query sequence $\tilde{\mathbf{Q}}$ whose input is $\mathbf{Q}(I)$, the $error(\tilde{\mathbf{Q}})$ is $\mathbb{E} \left\| \tilde{\mathbf{Q}} - \mathbf{Q} \right\|_2$. Here $\mathbb{E}$ is the expectation taken over the possible randomness in generating $\tilde{\mathbf{Q}}$.*

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

2.2 Constrained Inference

For a query sequence $\mathbf{Q}$ with constraint set $\gamma_{\mathbf{Q}}$, the constrained inference process takes the randomized output of the query, $\tilde{q} = \tilde{\mathbf{Q}}(I)$, and finds the "closest" set of query answers, $\tilde{q}$, that satisfy the constraints $\gamma_{\mathbf{Q}}$. Here *closest* is measured according to the L_2 distance, and we call the result the *minimum L_2 solution*:

Definition 2.4 (Minimum L_2 solution). *Let $\mathbf{Q}$ be a query sequence with constraints $\gamma_{\mathbf{Q}}$. A minimum L_2 solution for noisy query sequence $\tilde{q} = \tilde{\mathbf{Q}}(I)$, is a vector $\overline{q}$ that satisfies the constraints $\gamma_{\mathbf{Q}}$ and at the same time minimizes $\parallel \tilde{q} - \overline{q} \parallel_2$.*

We shall show that there is unique "closest" query sequence for both unattributed histograms and universal histograms. In these cases, we denote this unique solution by $\text{MIN}L_2(\tilde{\mathbf{S}}(I), \gamma_{\tilde{\mathbf{S}}})$ or $\text{MIN}L_2(\tilde{\mathbf{H}}(I), \gamma_{\tilde{\mathbf{H}}})$. We also derive closed form expressions for these quantities.

Uniqueness is one of the main reasons we choose to formulate the inference process using L_2 distance. If we minimize L_1 distance, we are not guaranteed a unique solution, and furthermore, some of the possible solutions are less accurate. In addition, we shall show that the minimal L_2 solution forms the estimator with least error in many cases.

Note that all query sequences considered in the paper consist of sets of counting queries. Thus the output sequences are always integral, non-negative values. The output of our randomized queries is neither integral nor non-negative because we add continuous Laplace noise. Integrality and non-negativity are typically achieved by moving negative values to zero, and rounding fractional values to the nearest integer. They usually do not have a large impact on accuracy.

To simplify the theoretical development, we do not include integrality and non-negativity as constraints in any constraint sets $\gamma_{\mathbf{Q}}$, but we do consider them in the experiments in Section 5.

3. UNATTRIBUTED HISTOGRAMS

In this section we describe a query for supporting unattributed histograms. First, we define the basic query $\mathbf{S}$ and contrast it with the conventional query $\mathbf{L}$. After defining its differentially private extension $\tilde{\mathbf{S}}$, we use constrained inference to derive a new estimator $\overline{\mathbf{S}}$ and compare the utility of $\tilde{\mathbf{S}}$ and $\overline{\mathbf{S}}$.

To support unattributed histograms, we can use the query $\mathbf{L}$ which will give a sequence of unit-length counts $\mathbf{L}[1]$, $\mathbf{L}[2], \ldots, \mathbf{L}[n]$, where $n = |dom|$. However for an unattributed histogram, we are interested only in the values of counts $\mathbf{L}[i]$, without requiring the association between $\mathbf{L}[i]$ and i. It follows that any permutation of $\mathbf{L}$ is a correct response for the unattributed histogram.

We define a new query $\mathbf{S}$ which returns the counts in ascending order. If U is the set of all unit length counts $U = \{c([x_j]) | x_j \in dom\}$, we write $rank_i(U)$ to refer to the i^{th} element in sorted, ascending order. We define:

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

Thus $\mathbf{S}[1]$ is the smallest count and $\mathbf{S}[n]$ is the largest count. Perhaps surprisingly, $\mathbf{S}$ has the same sensitivity as $\mathbf{L}$. In fact, we can make a general statement about the effect of sorting by value. Given a query sequence $\mathbf{Q}$, let $\mathbf{Q}' = sort(\mathbf{Q})$ correspond to the query that returns the counts of $\mathbf{Q}$ in sorted order.

Proposition 1. *The sensitivity of $\mathbf{Q}'$ is no larger than the sensitivity of $\mathbf{Q}$.*

Proof. Given two vectors X, Y, let l be the minimum L_1 distance between X and any permutation of Y. Of course, $|X - Y| \geq l$. It is a fact that $|sort(X) - sort(Y)| = l$. Therefore, the sensitivity of $\mathbf{Q}'$ is a lower bound for the sensitivity of $\mathbf{Q}$. □

We showed in Section 2.1 that $\mathbf{L}$ has sensitivity 1, so the sensitivity of $\mathbf{S}$ is also 1, and the following algorithm is ϵ-differentially private:

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

The accuracy of $\tilde{\mathbf{L}}$ and $\tilde{\mathbf{S}}$ is the same: $error(\tilde{\mathbf{L}}[i]) = error(\tilde{\mathbf{S}}[i]) = 2/\epsilon^2$, for each i. So it would appear as if we have failed in our task of improving the utility over $\tilde{\mathbf{L}}$. However, we note that $\mathbf{S}$ implies a powerful set of constraints because the elements of $\mathbf{S}$ must be increasing. That is, the constraint set $\gamma_{\mathbf{S}}$ contains a set of inequality constraints $\mathbf{S}[i] \leq \mathbf{S}[i+1]$ for each $i \in [1, n-1]$. We show next how to exploit these constraints.

3.1 Constrained Inference: computing $\overline{\mathbf{s}}$

The original private output $\tilde{s} = \tilde{\mathbf{S}}(I)$ will not, in general, satisfy the inequality constraints of $\gamma_{\mathbf{S}}$. But we know that prior to the addition of noise, the true sequence $\mathbf{S}(I)$ was in sorted order. So if for some i we observe that $\tilde{s}[i] > \tilde{s}[i+1]$, we must somehow reconcile this constraint violation. We could decrease $\tilde{s}[i]$ until it equals $\tilde{s}[i+1]$ or increase $\tilde{s}[i+1]$ until it equals $\tilde{s}[i]$ or set them both to a value in between $\tilde{s}[i]$ and $\tilde{s}[i+1]$. Suppose it also turns out that $\tilde{s}[i+1] = \tilde{s}[i+2]$, then this may be supporting evidence that the order violation is due to positive noise being added to $\tilde{s}[i]$ and we should decrease $\tilde{s}[i]$.

More generally, the constraints mean that the most accurate estimate for the i^{th} count should be a function not only of $\tilde{s}[i]$ but of the entire sequence. If the subsequence $\tilde{s}[1], \ldots, \tilde{s}[i-1]$ contains many counts that are larger than $\tilde{s}[i]$, then it is likely that $\tilde{s}[i]$ is too low. Conversely, if the subsequence $\tilde{s}[i+1], \ldots, \tilde{s}[n]$ contains many counts smaller than $\tilde{s}[i]$, this is evidence that $\tilde{s}[i]$ is too high. In this manner, the ordering constraints allows us to factor out some of the noise from $\tilde{s}$, and to infer a new sequence with significantly better utility.

Table 2: Examples showing the private output $\tilde{s} = \tilde{\mathbf{S}}(I)$, which may be unordered, and the corresponding closest ordered sequence $\overline{s} = \textsc{min}L_2(\tilde{s}, \gamma_{\mathbf{S}})$.

original output $\tilde{s}$	inferred output $\overline{s}$	distance $\lVert \tilde{s} - \overline{s} \rVert_2$
$\langle 10, 11, 13 \rangle$	$\langle 10, 11, 13 \rangle$	0
$\langle 10, 13, 11 \rangle$	$\langle 10, 12, 12 \rangle$	2
$\langle 14, 9, 10 \rangle$	$\langle 11, 11, 11 \rangle$	14
$\langle 14, 9, 10, 15 \rangle$	$\langle 11, 11, 11, 15 \rangle$	14

We factor out the noise by computing the ordered sequence closest to $\tilde{s}$, that is, $\overline{s} = \textsc{min}L_2(\tilde{s}, \gamma_{\mathbf{S}})$. We denote this new estimator for $\mathbf{S}(I)$ as $\overline{\mathbf{S}}$.

Example 1. *Table 2 shows examples of noisy outputs $\tilde{s}$ and the their minimum L_2 solutions, $\overline{s}$. Notice that in the first row, $\tilde{s} = (10, 11, 13)$ is already ordered, so the closest ordered sequence is just $\tilde{s}$ itself. For the out-of-order sequence $\tilde{s} = (10, 13, 11)$, the closest ordered sequence is $\overline{s} = (10, 12, 12)$.*

A method for computing $\textsc{min}L_2(\tilde{s}, \gamma_{\mathbf{S}})$ is not obvious, but the following theorem shows that there is in fact a closed-form solution which can be computed efficiently. 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 $M[i,j]$ record the mean of these elements, i.e. $M[i,j] = \sum_{k=i}^{j} \tilde{s}[k]/(j - i + 1)$.

Theorem 1. *Denote $L_k = \min_{j \in [k,n]} \max_{i \in [1,j]} M[i,j]$ and $U_k = \max_{i \in [1,k]} \min_{j \in [i,n]} M[i,j]$. The minimum L_2 solution $\tilde{s} = \textsc{min}L_2(\tilde{s}, \gamma_{\mathbf{S}})$, is unique and given by: $\tilde{s}[k] = L_k = U_k$.*

The proof appears in Appendix A.1. Below, we apply the theorem to the example.

Example 2. *Let us see how we compute $\overline{s}$ when $\tilde{s} = (14, 9, 10, 15)$. To compute $\overline{s}[1]$, by the above theorem, we can either compute L_1 or U_1 (as both are equal by our Theorem). Let us compute $U_1 = \max_{i \in [1,1]} \min_{j \in [i,4]} M[i,j]$, which is simply $\min_{j \in [1,4]} M[1,j]$, i.e. the smallest mean among the means for subsequences starting from 1. In this case, it is easy to see that this minimum is $M[1,3]$ which gives $\overline{s}[1] = U_1 = 11$. Similarly, we can compute $\overline{s}[2]$ and $\overline{s}[3]$. Now let us compute $\overline{s}[4]$. We shall compute $L_4 = \min_{j \in [4,4]} \max_{i \in [1,j]} M[i,j]$, which is simply $\max_{i \in [1,4]} M[i,4]$, i.e. the largest mean among the means for subsequences ending at 4. In this case, it is easy to see that this minimum is at $M[4,4]$ which gives $\overline{s}[4] = L_4 = 15$. Thus $\overline{s} = (11, 11, 11, 15)$.*

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

In this section, we analyze $\overline{\mathbf{S}}$ and show that it has much better utility than $\tilde{\mathbf{S}}$ (and therefore much better utility than $\tilde{\mathbf{L}}$ for unattributed histograms). Before presenting the theoretical statement of utility, we first give an example that illustrates under what conditions $\overline{\mathbf{S}}$ is likely to reduce error.

Example 3. *Figure 2 graphically depicts $\tilde{s}$ and $\overline{s}$ for an unattributed histogram of length 25. The true sequence $\mathbf{S}(I)$*

has three distinct counts: the first twenty elements of the sequence equal 10, followed by one count of 14, and then four counts equal to 18. The circles depict the original private output $\tilde{s}$ and squares depict the much more accurate inferred sequence, $\overline{s}$. Since $\overline{s}$ must obey the order constraints, it must find the monotone non-decreasing path closest to the noisy observations $\tilde{s}$. For the uniform subsequence from 1 to 20, this path is very close to the true answer. Essentially, the noise of neighboring positions in the sequence cancels out. This is not true for the twenty-first position, and the path passes directly through the noisy observation. In addition to showing a single $\tilde{s}$ and $\overline{s}$, the figure also depicts the expected error. The expected error of $\tilde{\mathbf{S}}$ (dotted lines) is ± 2 at all positions in the sequence. However, the expected error for $\overline{\mathbf{S}}$ (solid error bars) varies: it is smallest for positions in the middle of the uniform subsequence and largest for the endpoints.

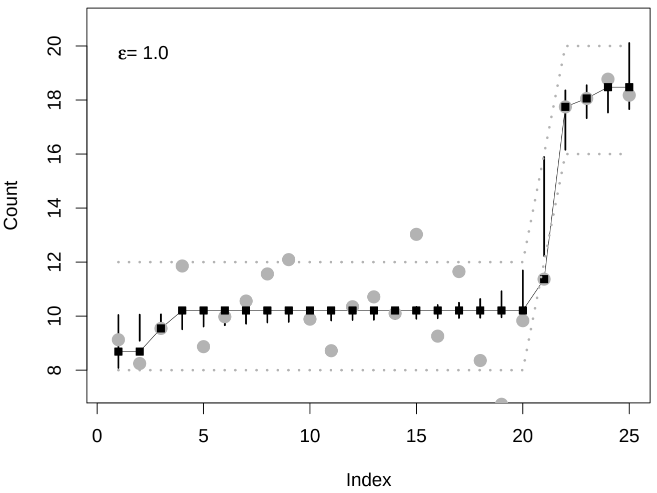


Figure 2: For a small sample data set, the original private estimates $\tilde{s} = \tilde{\mathbf{S}}(I)$ (grey circles) and inferred estimates $\overline{s} = \overline{\mathbf{S}}(I)$ (black squares).

The figure suggests that $error(\overline{\mathbf{S}})$ will be low for sequences that are concentrated, meaning that many counts are the same. We now quantify the utility of $\overline{\mathbf{S}}$ precisely.

Denote n and d as 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 exists 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 $error(\overline{\mathbf{S}}) = O(d \log^3 n/\epsilon^2)$. In comparison, $error(\tilde{\mathbf{S}}) = \Theta(n/\epsilon^2)$.

See Appendix A.2 for proof.

The above theorem shows that $error(\overline{\mathbf{S}})$ is better than $error(\tilde{\mathbf{S}})$. 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. That is an extreme case, but for most unattributed histograms found in practice, $d \ll n$, and thus $error(\tilde{\mathbf{S}})$ is significantly better than $error(\overline{\mathbf{S}})$. In Sec. 5, experiments on real data demonstrate that the error of $\overline{\mathbf{S}}$ can be several orders of magnitude lower than the error of $\tilde{\mathbf{S}}$.

4. UNIVERSAL HISTOGRAMS

In this section we describe our query strategy for supporting universal histograms. After defining the basic query, $\mathbf{H}$, and its differentially-private extension, $\tilde{\mathbf{H}}$, we describe the constrained inference process that transforms $\tilde{\mathbf{H}}$ into a consistent answer $\overline{\mathbf{H}}$. We conclude this section with an analysis of utility, comparing the expected error of $\overline{\mathbf{H}}$ against that of $\tilde{\mathbf{H}}$ as well as the natural baseline approach of $\tilde{\mathbf{L}}$.

The query $\mathbf{H}$ is defined with respect to a tree T which we define next. The tree T is formed by considering a recursive partitioning of dom. Each node v of the tree has an associated interval in dom, denoted as $range_v$. The root node r has range dom. The children of node v, denoted $succ(v)$, have ranges that partition $range_v$. That is, $range_v = \cup_{u \in succ(v)} range_u$ and $range_u$ is disjoint among the nodes $u \in succ(v)$. We denote $pred(v)$ as the parent of a node v. The height h of the tree is the number of nodes in the longest path from a leaf to the root. To simplify the discussion we will assume the following: $|dom| = n$ and T is a complete tree of degree k (i.e. each node has k children) and therefore $k^{h-1} = n$.

Before defining $\mathbf{H}$, we first show the tree T for the running example from Figure 1.

EXAMPLE 4. *Figure 3 shows a tree T over the domain of source IP addresses $\{000, 001, 010, 011\}$. Here $k = 2$ which results in a binary tree with height $h = 3$. The root r is associated with the range $range_r = [0 * *]$. This range is evenly subdivided among the children of r. Finally, the leaves of the tree consist of unit-length ranges,*

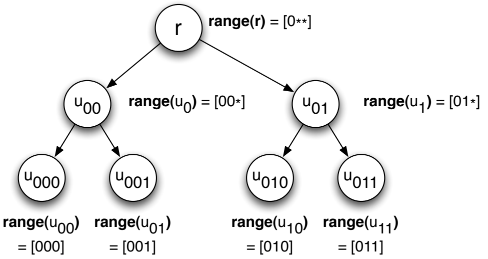


Figure 3: Continuation of the example in Figure 1. Shown is the tree T that defines a recursive partitioning of the domain of source IP addresses. Each node in the tree is associated with a range over $dom = \{000, 001, 010, 011\}$.

The query sequence $\mathbf{H}$ consists of a query for each node $v \in T$, counting the number of tuples in $range_v$. We index $\mathbf{H}$ using node identifiers, and write $\mathbf{H}[v] = c(range_v)$. Thus $\mathbf{H}[r]$ returns the total number of tuples in the relation R, and for a leaf u, $\mathbf{H}[u]$ returns a unit-length count. Observe that since $\mathbf{H}$ includes all unit-length counts, $\mathbf{H}$ contains the query sequence $\mathbf{L}$.

The counts associated with each internal node must equal the sum of the counts of their children. Thus, for each $v \in T$, the constraint set $\gamma_{\mathbf{H}}$ contains the constraint that $\mathbf{H}[v] = \sum_{u \in succ(v)} \mathbf{H}[u]$.

The sensitivity of $\mathbf{H}$ is equal to the height of the tree, h. If a tuple is added or removed from the relation, this affects the count for every range that includes this tuple. Since the tree is a recursive partition of the range, a single tuple is contained in the range of a single leaf and the ranges of the nodes along the path from the leaf to the root. Therefore, in the resulting vector for $\mathbf{H}$, exactly h counts will differ each by exactly 1. Thus, the following algorithm is ϵ-differentially private:

$$\tilde{\mathbf{H}} = \mathbf{H} + \langle \mathrm{Lap}(h/\epsilon) \rangle^{2n-1}$$

Since $\tilde{\mathbf{H}}$ is obtained by adding independent noise to each count, it may fail to satisfy the constraints in $\gamma_{\mathbf{H}}$. Next we show how to exploit these constraints to improve accuracy.

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

We now consider the problem of using the noisy answers $\tilde{\mathbf{H}}$ to derive a consistent estimate for each count in $\mathbf{H}$. For example, suppose that $\tilde{h} = \tilde{\mathbf{H}}(I)$ and consider deriving an estimate for the count at the root. One estimate is simply $\tilde{h}[r]$, the noisy count at the root. However, given the constraints, an alternative estimate for the root count is $\sum_{u \in succ(r)} \tilde{h}[u]$. In fact, one can derive h different estimates for the root count by summing the counts across each level of the tree. The estimates become increasingly noisy because each is a sum of an increasingly larger set of noisy counts. Nevertheless, since the noise is added independently to each count, the h estimates are independent observations of the same unknown quantity. The challenge is combining them in a natural way so that noisier estimates are given less "weight." A second challenge is to do this in such a way that the final answer is *consistent.* It turns out that minimum L_2 solution addresses both of these issues simultaneously. We describe it next.

Given the noisy output $\tilde{h} = \tilde{\mathbf{H}}(I)$, we now characterize the inferred query answer, $\overline{h} = MinL_2(\tilde{\mathbf{H}}(I), \gamma_{\mathbf{H}})$. Next we obtain a possibly inconsistent estimate, $z[v]$, for each node v as below. Let $h(v)$ denote the height of node v, and set $\alpha = k^{h(v)-1}$.

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

This follows the intuition that given an estimate $z[u]$ for the successors u of v, the quantity $\sum_{u \in succ(v)} z[u]$ is also an estimate for the count at v. The estimate $z[v]$ is a weighted sum of this estimate and $\tilde{h}[v]$, the noisy count at node v.

Once we have the estimate $z[v]$ for each node v, we can compute the required consistent estimate $\overline{h}$. This is done in a top-down fashion: we begin by assigning $\overline{h}[r] = z[r]$ for the root node r. 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 w by dividing the difference $\overline{h}[u] - \sum_{w \in succ(u)} z[w]$ equally among the k descendants.

The following proposition, proved in Appendix B.1, shows that this procedure in fact gives the minimum L_2 solution.

Proposition 2. *Given the noisy count sequence $\tilde{h} = \mathbf{H}(I)$, the unique minimum L_2 solution, $\overline{h} = \mathrm{MIN}L_2(\tilde{h}, \gamma_{\mathbf{H}})$, is given by the following recurrence relation (denoting $u = pred(v)$):*

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

We shall now compare $error(\overline{\mathbf{H}})$ with other possible estimates obtained using $\tilde{\mathbf{H}}$. We restrict ourselves to the class of linear unbiased estimators, which we define next. An estimator $E = \{E[v]\}_{v \in nodes(T)}$ is called linear, if for all nodes v, $E[v]$ can be expressed as a linear combination of the noisy counts in $\tilde{\mathbf{H}}$. Further, E is called unbiased, if for all nodes v, $\mathbb{E}(E[v]) = \mathbf{H}[v]$ where the expectation is computed over randomness in generating $E[v]$. Next we state a theorem that shows how $\overline{\mathbf{H}}$ compares with other estimators E. The proof uses the Gauss-Markov theorem [19] and is included in Appendix B.2.

Theorem 3. *If $\overline{\mathbf{H}}$ is defined as $\mathrm{MIN}L_2(\tilde{\mathbf{H}}(I), \gamma_{\mathbf{H}})$ then $\overline{\mathbf{H}}$ satisfies the following properties:* (*i*) *$\overline{\mathbf{H}}$ is a linear unbiased estimator,* (*ii*) *$error(\overline{\mathbf{H}})$ is the minimal among the $error(E)$ of all linear unbiased estimators E.*

In particular, since $\tilde{\mathbf{H}}$ is also linear and unbiased, we get $error(\overline{\mathbf{H}}) \leq error(\tilde{\mathbf{H}})$ showing that enforcing constraints only improves utility.

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

Universal histograms are designed to support the accurate computation of arbitrary range queries. Here we compare the utility of our inferred hierarchical estimates ($\overline{\mathbf{H}}$) with the hierarchical estimates $\tilde{\mathbf{H}}$ and with the conventional technique $\tilde{\mathbf{L}}$.

We use q to denote a single counting query $c([x, y])$ for some $x, y \in dom$. For any such q we use an indicator function χ_q that maps leaf nodes of the tree T to $\{0, 1\}$ such that the leaf nodes l that are mapped to 1 are exactly those in the range of q. The answer to q is simply $\sum_{l \in leaves(T)} \chi_q(l)\mathbf{H}[l]$.

Since $\overline{\mathbf{H}}$ is consistent, it is straightforward to use it to compute answers to range queries. Denote $\overline{\mathbf{H}}_q$ as the estimate of q obtained using $\overline{\mathbf{H}}$. Thus $\overline{\mathbf{H}}_q = \sum_{l \in leaves(T)} \chi_q(l)\overline{\mathbf{H}}[l]$.

An estimate for q can also be obtained from $\tilde{\mathbf{H}}$. In fact, since $\tilde{\mathbf{H}}$ is inconsistent, multple estimates are possible. For example, consider query $q_r = c(dom)$ which returns the total number of tuples. We can estimate q_r using both $\tilde{\mathbf{H}}[r]$ (the noisy count for the root r) and $\sum_{l \in leaves(T)} \tilde{\mathbf{H}}[l]$ (the sum of noisy counts at the leaves). We choose a unique estimate, defining $\tilde{\mathbf{H}}_q$ as the estimate obtained by summing over the least number of noisy counts in $\tilde{\mathbf{H}}$. More precisely, let $\mathcal{T}_1, \ldots, \mathcal{T}_k$ be the set of disjoint subtrees of $\tilde{\mathbf{H}}$ such that the union of their ranges equals $[x, y]$. The estimator returns $\sum_{i=1}^{k} \tilde{\mathbf{H}}[r_i]$ where r_i is the root of $\mathcal{T}_i$. Thus, for our example query q_r, $\tilde{\mathbf{H}}_{qr} = \tilde{\mathbf{H}}[r]$. Note that this problem of multiple possible estimates does not arise for consistent count estimators such as $\overline{\mathbf{H}}$, as all possible ways of estimating q using $\overline{\mathbf{H}}$ yield the same answer.

Finally, we can also use $\tilde{\mathbf{L}}$ to estimate range query answers. For query q, the estimate is $\tilde{\mathbf{L}}_q = \sum_l \chi_q(l)\tilde{\mathbf{L}}[l]$. Recall that $\tilde{\mathbf{L}}$ has lower sensitivity than $\tilde{\mathbf{H}}$ so the amount of noise added is less. The expected error for each unit-length count is $2/\epsilon^2$. However, for a range query, the expected error of $\tilde{\mathbf{L}}_q$ is linear in the size of the range.

The $error(\tilde{\mathbf{H}})$ is $2h^2/\epsilon^2$. Notice that $\tilde{\mathbf{H}}$ is less accurate than $\tilde{\mathbf{L}}$ for estimating unit-length counts: $2h^2/\epsilon^2$ is worse than $2/\epsilon^2$. However, $\tilde{\mathbf{H}}$ is much more accurate than $\tilde{\mathbf{L}}$ for estimating counts of large ranges. The estimate $\tilde{\mathbf{H}}[r]$ for the total count has *error* of $2h^2/\epsilon^2$, which is much better than $2n/\epsilon^2$ given by $\tilde{\mathbf{L}}$ (recall $n = 2^h$). The experiments in Sec 5 show clearly the accuracy of these two techniques as a function of range size.

Finally we show additional optimality results for our inferred estimates for range queries. The proof uses the Gauss-Markov theorem [19] and is included in Appendix B.3.

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

In particular, for a height 11 quadtree (i.e. $h = 11, k = 4$), the above theorem shows that for some range queries q, the error of $\overline{\mathbf{H}}[q]$ may be better than that of $\tilde{\mathbf{H}}[q]$ by as much as $2(h-1)(k-1)/3 = 20$ times.

Finally we note that a differentially private technique for answering range queries is also given in [7]. The technique uses a similar idea of recursive partitioning but creates leaf nodes that correpsond to equi-depth histograms, as opposed to the equi-width histogram obtained at the leaf nodes in the $\mathbf{H}$ queries. In terms of utility, this equi-width technique yields a bound on the *error* of $O(\frac{m^{4/3}}{\epsilon^2})$ where m is the total number of data values. Since the counts themselves belong in the range $\{1, \ldots, m\}$, this may result in a severe distortion in the returned estimates. On the other hand, our bound on error is $O(h^3/\epsilon^2)$, where h is typically small, say 10, and thus our technique can return quite accurate results.

5. EXPERIMENTS

In this section, we evaluate the proposed techniques on real data drawn from three different domains. The results for unattributed histograms and universal histograms are described separately. Below, we describe the datasets used in the experiments.

- `NetTrace` is derived from an IP-level network trace collected at a major university [1]. The trace monitors traffic at the gateway between IP addresses internal to the institution 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. In this context, the differential privacy guarantee ensures that individual connections remain private.

- `Social Network` is a graph derived from a the friendship relations on an online social network site [4]. The graph is limited to a population of roughly $11K$ 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. (This is not a critical assumption and, in fact, the number of students can be estimated privately within ±$1/\epsilon$ in expectation. Our techniques can be applied directly to either the true count or a noisy estimate.)

- `Search Query 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 [2]. We combined these summary statistics with a second dataset [3], which contains actual search 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. (See [6] for information on the privacy breaches that resulted from the publication of the "anonymized" query logs of [3].) Differential privacy can guarantee that the output will prevent the association of an individual user (or host machine) with a particular search term.

5.1 Unattributed histograms

The first set of experiments evaluates the accuracy of constrained inference on unattributed histograms. We will compare three estimators that use the differentially private output $\tilde{\mathbf{S}}(I)$ to derive an estimate of the true (hidden) input $\mathbf{S}(I)$. We compare the estimators described earlier: $\tilde{\mathbf{S}}$ and $\overline{\mathbf{S}}$. Recall that $\tilde{\mathbf{S}}(I)$ is likely to be inconsistent — out-of-order, non-integral, and possibly negative. So we consider a second baseline technique, denoted $\tilde{\mathbf{S}}_{\mathbf{r}}$, which enforces consistency as follows: if $\tilde{s} = \tilde{\mathbf{S}}(I)$, then $\tilde{s}_r$ is derived by sorting $\tilde{s}$ and rounding each count to the nearest non-negative integer.

To compare the estimators, we evaluate their performance on three queries from the three datasets:

1. On `NetTrace`: the connectivity between internal hosts and a large subnet of the external hosts ($\approx 65K$ hosts). The query returns the degree sequence of the external hosts.

2. On `Social Network`, the distribution of node degrees in sorted order (a person's degree is the number of "friends").

3. On `Search Query Logs`, the frequency distribution over a 3 month period of the top $20K$ queries. The output is a vector where position i is the number of times the i^{th} ranked query was searched.

The runtime of $\overline{\mathbf{S}}$ varied across datasets and ϵ, but the longest query took less than 3 minutes to compute (`NetTrace`, $\epsilon = 0.1$).

To evaluate the utility an of estimator, we measure its squared error. Results report the average squared error over 10 random samples from the differentially-private mechanism (each sample produces a new $\tilde{\mathbf{S}}(I)$). We also show results for three settings of $\epsilon = \{2.0, 1.0, 0.1\}$; smaller ϵ means more privacy and hence more random noise. To facilitate comparison across datasets and settings of ϵ, the error is normalized based on the expected error of $\tilde{\mathbf{S}}$. More precisely, we divide the observed squared error by $error(\tilde{\mathbf{S}})/n$ where n is the size of the unattributed histogram. Therefore, we expect the normalized error of the first baseline, $\tilde{\mathbf{S}}$, to be 1.

Figure 4 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

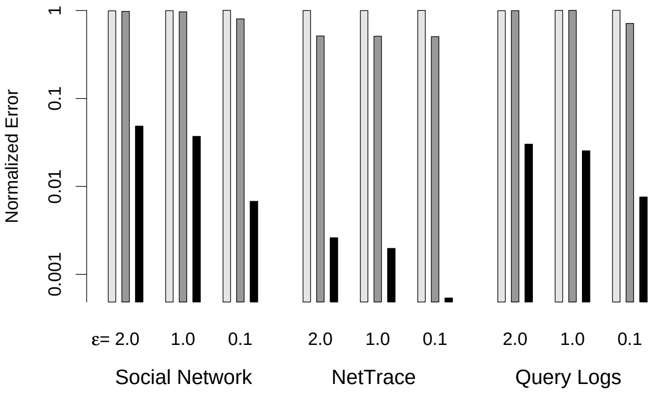


Figure 4: A comparison of normalized error of three estimators across varying datasets and ϵ. Each triplet of bars represents from left-to-right: $\tilde{\mathbf{S}}$ (light gray), $\tilde{\mathbf{S}}_\mathbf{r}$ (gray), and $\overline{\mathbf{S}}$ (black).

axis is normalized error on a log-scale; normalized error less than 1 indicates that the estimator has lower average error than $\tilde{\mathbf{S}}$. 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. Finally, the accuracy only improves with decreasing ϵ (more noise).

Figure 5 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 5 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.

5.2 Universal histograms

We now evaluate the effectiveness of constrained inference for the more general task of computing a universal histograms and arbitrary range queries.

We evaluate three techniques for supporting universal histograms. All three approaches ensure differential privacy, but they can be distinguished by their choice of query sequence and the use of constrained inference. We evaluate the estimators in terms of accuracy over an arbitrary workload of range queries.

The first technique uses the unit counts, $\tilde{\mathbf{L}}$, as the basis for its histogram. Since the answer contains only unit counts, there are no consistency constraints, and one can compute a range query by simply summing the noisy unit counts. For range $[x, y]$, if $\tilde{l} = \tilde{\mathbf{L}}(I)$, then the estimate for the count is $c(x, y) = \Sigma_{i \in [x,y]} \tilde{l}[i]$.

The second technique uses the hierarchy of counts, $\tilde{\mathbf{H}}$. In the experiments, the hierarchy forms a binary tree over *dom*. For range query $c(x, y)$, many estimates are possible because the levels of the hierarchy are not mutually consistent. In the experiments, we estimate the answer by summing over the least number of noisy counts in $\tilde{\mathbf{H}}$ (see Sec 4.2).

Finally, we compare these two approaches against $\overline{\mathbf{H}}$, which is the minimum L_2 solution given $\tilde{\mathbf{H}}(I)$ and $\gamma_\mathbf{H}$. Because it is consistent, the answer to a range query is simply the sum of the appropriate leaf counts.

We evaluate the accuracy over a set of range queries of varying size and location. The range sizes are 2^i for $i = 1, \ldots, h-1$ where h 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 samplings of $\tilde{\mathbf{L}}(I)$ and $\tilde{\mathbf{H}}(I)$, and for each sample, 1000 randomly chosen ranges.

Using the datasets described above, we evaluate the following histogram queries:

- `NetTrace`: the number of connections for each external host. This is similar to the query in Section 5.1 except that here the association between IP address and count is retained.

- `Search Query Logs`: the temporal frequency of the query term "Obama" from Jan. 1, 2004 to present. (The temporal resolution is in units of 1.5 hours, 16 units of time per day.)

Figure 6 shows the results for both datasets and varying ϵ. The top row corresponds to `NetTrace`, the bottom to `Search Query Logs`. Within a row, each plot shows a different setting of $\epsilon \in \{0.01, 0.1, 1.0, 2.0\}$. For all plots, the horizontal axis indicates the size of the range query, and the vertical axis is the squared error, averaged over sampled counts and intervals. Both axes are in log-scale.

The results reveal several interesting patterns.

- 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 has lower expected error. However, the error of $\tilde{\mathbf{L}}$ increases linearly with the size of the range. In contrast, the error of $\tilde{\mathbf{H}}$ increases linearly with the size of the minimal set of subtrees spanning the range. For ranges larger than

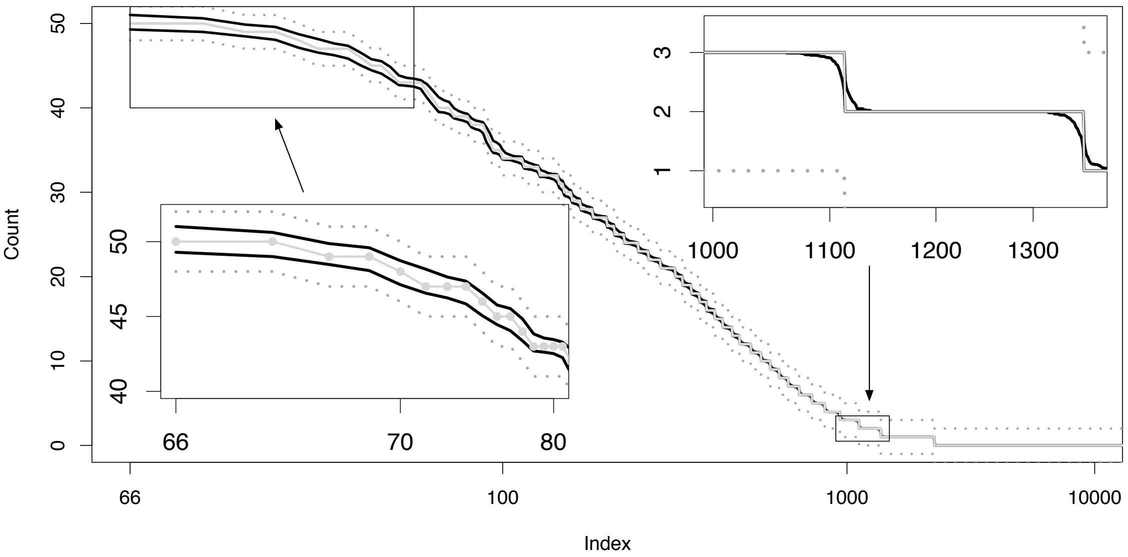


Figure 5: The unattributed histogram of `NetTrace` (solid gray), along with the average error of $\overline{\mathrm{S}}$ (solid black) and $\tilde{\mathrm{S}}$ (dotted gray), for $\epsilon = 1.0$.

about 2000 units, the error of $\tilde{\mathbf{L}}$ is higher than $\tilde{\mathbf{H}}$. For the root query, the error of $\tilde{\mathbf{L}}$ is larger than the error of $\tilde{\mathbf{H}}$ by roughly two orders of magnitude. This conforms with the theoretical expectation which is $\frac{n}{\log^2 n}$.

- Next, we compare the constrained estimator $\overline{\mathbf{H}}$ against its input $\tilde{\mathbf{H}}$. The error of $\overline{\mathbf{H}}$ is uniformly lower across all range sizes, settings of ϵ, and datasets. The performance gap is smallest for the root query. However, even at the root, the estimate of $\overline{\mathbf{H}}$ is more accurate because it combines h noisy observations (one per level of the tree) rather than just the root observation.

- 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 a smaller ϵ implies noisier observations. 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 Laplacian noise of $\overline{\mathbf{H}}$ is *larger* than that of $\tilde{\mathbf{L}}$ by a factor of $\log n$. The reduction in error is due to the fact the 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 half of the leaves in the range (in expectation).

Figure 7 shows a plot of searches for "Obama" over time for $\epsilon = 0.1$. The ground truth is shown in black along with the noisy estimates; and the estimates of the proposed approach are shown in red. The top figure shows the trends at the resolution of monthly intervals; the bottom figure is more fine-grained resolution at daily frequencies. (Because our approach outputs a single consistent estimate, the data can be analyzed at different resolutions and the counts will be consistent.) This graphic illustrates how the approach provides an extremely accurate reconstruction of the high-level (monthly) trends — the estimate is almost indistinguishable from ground truth. As a the finer-resolution of daily frequencies, the counts are small and the noise causes the estimator to misallocate some records to neighboring days. However, even at the daily resolution, some trends are quite apparent, such as the spike in July 28, 2004 (the day following Obama's speech at the Democratic convention) and November 5, 2008 (the day after winning the presidential election).

6. RELATED WORK

Since differential privacy was introduced [10], it has become an active area of data privacy research (for a compre-

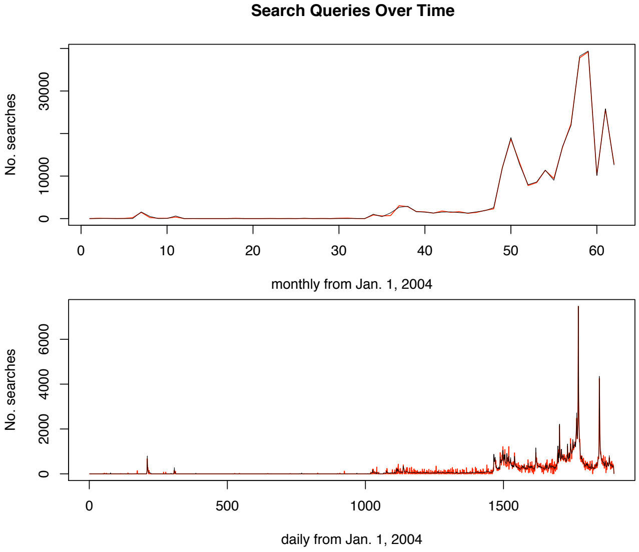


Figure 7: User searches for "Obama" over time, true in black and $\tilde{\mathbf{H}}$ in red, $\epsilon = 0.1$.

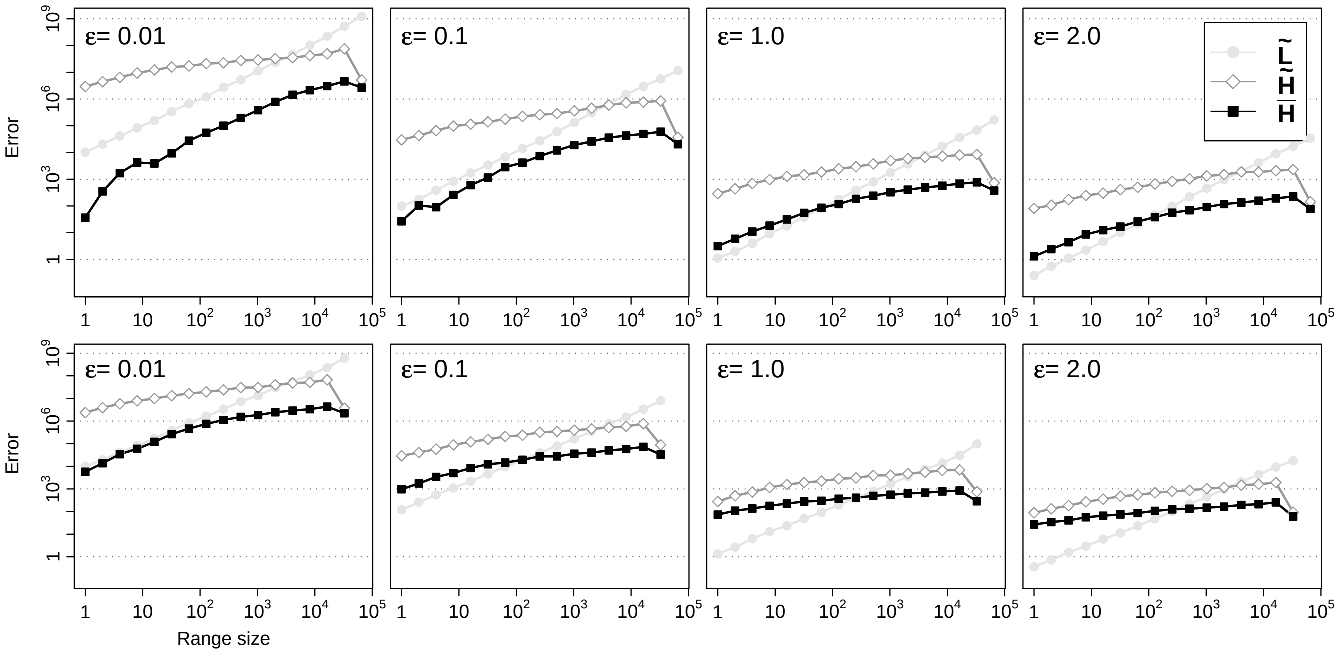


Figure 6: A comparison of universal histogram estimators $\tilde{\mathbf{L}}$ (diamonds), $\tilde{\mathbf{H}}$ (circles), and $\overline{\mathbf{H}}$ (squares) on two real-world datasets (top `NetTrace`, bottom `Search Query Logs`).

hensive review, see [9]). Below we highlight results closest to the present work.

The problem of producing query answers that are both private and consistent was explicitly considered for contingency tables [5]. The proposed algorithm outputs a set of consistent marginals. Similar to our approach, Laplace noise is added to guarantee differential privacy. A key difference from our work is that consistency is addressed by reformulating the query such that random perturbation does not produce inconsistency. More precisely, instead of returning noisy marginals, the data is projected onto a set of orthogonal basis vectors and then noise is added to the vector coefficients. The output defines a contingency table with possibly fractional/negative counts (which is then post-processed to achieve non-negativity and integrality). Another important difference is that the number of constraints is much lower $O(\sqrt{n})$ since there are at most $O(\sqrt{n})$ marginals in a table with n entries. As a consequence, enforcing the constraints does not increase utility. We consider more general constraints, and a larger number of constraints ($O(n)$), and show they improve utility.

Ghosh et al. [11] prove that, for a single counting query, the differentially-private mechanism of Laplace noise is, in fact, optimal in terms of utility. Every user—regardless of their prior information or utility goals—receives as much utility from this mechanism as from interacting with any other differentially-private mechanism. The results holds for a single counting query; the optimal mechanism for a sequence of counting queries remains an open problem. Our work contributes the insight that for a vector of queries the analogous result does not hold: given that constraints can reduce noise without violating privacy, this implies that independent Laplace noise is a sufficient, but not *necessary* condition for privacy.

We are aware of ongoing work that uses a differentially-private mechanism to generate synthetic data [14]. The focus is on domains that have hierarchical structure (similar to our universal histogram), and the techniques are based on an adaptive mechanism where the analyst recursively partitions the domain along paths that contain a sufficient quantity of data. In contrast, our approach descends down all paths (using a small, $\log n$, privacy budget) and combines the noisy observations (and associated constraints) to derive its estimate. Our results suggest that if any adaptive query goes to depth d, it is worth asking all queries at depth d since it will not increase sensitivity and the constrained inference process can add accuracy.

Blum et al. [7] propose an efficient algorithm to publish synthetic data that can be used to accurately answer half-space queries. They also present a solution for range queries; see Sec 4.2 for a direct comparison with the present work.

The analysis of social network data is an important application area for privacy research [12, 13, 20]. The focus has been primarily on publishing transformed networks which protect anonymity. A common measure of utility for these techniques has been the distortion of the degree distribution. Preserving this property of the graph is important since it has been shown to have profound implications on network structure and dynamics [16]. Our technique can be used to publish a degree distribution that is much more accurate, but the graph remains hidden. Perhaps our technique can be a component of a privacy-protecting algorithm for generating synthetic networks.

In our own prior work [18] we have considered techniques for privately answering queries involving joins, such as the number of triangles in a social network. Such queries have high sensitivity and cannot be accurately analyzed under differential privacy. That paper considered an alternative notion of privacy and proposed accurate perturbation techniques that did not apply constraints. The present paper uses constraints to show that the degree distribution is a graph property that can be accurately measured under differential privacy.

7. DISCUSSION & CONCLUSIONS

Here we recap our 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 inequalty 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. 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 exiting 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) [11]. 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 directly from the improved distribution.

8. REFERENCES

[1] `http://traces.cs.umass.edu/index.php/Network`.

[2] `http://www.google.com/trends`.

[3] `http://www.gregsadetsky.com/aol-data/`.

[4] `http://www.michaelkelly.org/projects/fb/`.

[5] 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.

[6] M. Barbaro and T. Z. Jr. A face is exposed for AOL searcher no. 4417749. *New York Times*, 2006.

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

[8] F. R. K. Chung and L. Lu. Survey: Concentration inequalities and martingale inequalities. *Internet Mathematics*, 2006.

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

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

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

[12] M. Hay, G. Miklau, D. Jensen, D. Towsley, and P. Weis. Resisting structural re-identification in anonymized social networks. In *VLDB*, 2008.

[13] K. Liu and E. Terzi. Towards identity anonymization on graphs. In *SIGMOD*, 2008.

[14] F. McSherry. Personal communication, February 2009.

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

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

[17] G. Owen. *Game Theory.* Academic Press Ltd, 1982.

[18] V. Rastogi, M. Hay, G. Miklau, and D. Suciu. Relationship privacy: Output perturbation for queries with joins. In *To appear, PODS*, 2009.

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

[20] B. Zhou and J. Pei. Preserving privacy in social networks against neighborhood attacks. In *ICDE*, 2008.

APPENDIX

A. UNATTRIBUTED HISTOGRAMS

A.1 Proof of Theorem 1

We first restate the theorem below. Recall that $\tilde{s}[i,j]$ denotes the subsequence of $j-i+1$ elements: $\langle\tilde{s}[i]$, $\tilde{s}[i+1]$, $\ldots,\tilde{s}[j]\rangle$. Let $\tilde{M}[i,j]$ record the mean of these elements, i.e. $\tilde{M}[i,j]=\sum_{k=i}^{j}\tilde{s}[k]/(j-i+1)$.

Theorem 1. *Denote $L_k=\min_{j\in[k,n]}\max_{i\in[1,j]}M[i,j]$ and $U_k=\max_{i\in[1,k]}\min_{j\in[i,n]}M[i,j]$. The minimum L_2 solution $\tilde{s}=\textsc{min}L_2(\tilde{s},\gamma_{\mathbf{S}})$, is unique and given by: $\tilde{s}[k]=L_k=U_k$.*

Proof. In the proof, we abbreviate the notation and implicitly assume that the range of i is $[1,n]$ or $[1,j]$ when j is specified. Similarly, the range of j is $[1,n]$ or $[i,n]$ when i is specified.

We start with the easy part, showing that $U_k\leq L_k$. Define an $n\times n$ matrix A^k as follows:

$$A^k_{ij} \;=\; \begin{cases}\tilde{M}[i,j] & \text{if i} \leq \text{j}\\ \infty & \text{if j} < \text{i} \leq \text{k}\\ -\infty & \text{otherwise}\end{cases}$$

Then $min_j max_i A^k_{ij}=L_k$ and $max_i min_j A^k_{ij}=U_k$. In any matrix A^k, $max_i min_j A^k_{ij}\leq min_j max_i A^k_{ij}$: this is a simple fact that can be checked directly, or see [17], hence $U_k\leq L_k$.

We show next that if $\bar{s}$ is a solution of $\textsc{min}L_2(\tilde{s},\gamma_{\mathbf{S}})$, then $L_k\leq\bar{s}[k]\leq U_k$. If we show this, then the proof of the theorem is completed, as then we will then have $\bar{s}[k]=L_k=U_k$. The proof relies on the following lemma.

Lemma 1. *Let $\bar{s}$ be a solution of $\textsc{min}L_2(\tilde{s},\gamma_{\mathbf{S}})$. Then (i) $\bar{s}[1]\leq U_1$, (ii) $\bar{s}[n]\geq L_n$, (iii) for all k, $\min(\bar{s}[k+1],\max_i\tilde{M}[i,k])\leq\bar{s}[k]\leq\max(\bar{s}[k-1],\min_j\tilde{M}[k,j])$.*

The proof of the lemma appears below, but now we use it to complete the proof of Theorem 1. First, we show that $\bar{s}[k]\leq U_k$ using induction on k. The base case is $k=1$ and it is stated in the lemma, part (i). For the inductive step, assume $\bar{s}[k-1]\leq U_{k-1}$. From (iii), we have that

$$\begin{aligned}\bar{s}[k] &\leq \max(\bar{s}[k-1],\min_j\tilde{M}[k,j])\\ &\leq \max(U_{k-1},\min_j\tilde{M}[k,j])\\ &= U_k\end{aligned}$$

The last step follows from the definition of U_k. A similar induction argument shows that $\bar{s}[k]\geq L_k$, except the order is reversed: the base case is $k=n$ and the inductive step assumes $\bar{s}[k+1]\geq L_{k+1}$. □

The only remaining step is to prove the lemma.

Proof of Lemma 1. For (i), it is sufficient to prove that $\bar{s}[1]\leq\tilde{M}[1,j]$ for all $j\in[1,n]$. Assume the contrary. Thus there exists a j such that for $\bar{s}[1]>\tilde{M}[1,j]$. Let $\delta=\bar{s}[1]-\tilde{M}[1,j]$. Thus $\delta>0$. Further, for all i, denote $\delta_i=\bar{s}[i]-\bar{s}[1]$. Consider the sequence $\bar{s}'$ defined as follows:

$$\bar{s}'[i] \;=\; \begin{cases}\bar{s}[i]-\delta & \text{if } i\leq j\\ \bar{s}[i] & \text{otherwise}\end{cases}$$

It is obvious to see that since $\bar{s}$ is a sorted sequence, so is $\bar{s}'$.

We now claim that $\|\,\bar{s}'-\tilde{s}\,\|_2<\|\,\bar{s}-\tilde{s}\,\|_2$. For this note that since the sequence $\bar{s}'[j+1,n]$ is identitcal to the sequence $\bar{s}[j+1,n]$, it is sufficient to prove $\|\,\bar{s}'[1,j]-\tilde{s}[1,j]\,\|_2<\|\,\bar{s}[1,j]-\tilde{s}[1,j]\,\|_2$. To prove that, note that $\|\,\bar{s}[1,j]-\tilde{s}[1,j]\,\|_2$ can be expanded as

$$\begin{aligned}\|\,\bar{s}[1,j]-\tilde{s}[1,j]\,\|_2 &= \sum_{i=1}^{j}(\bar{s}[i]-\tilde{s}[i])^2=\sum_{i=1}^{j}(\bar{s}[1]+\delta_i-\tilde{s}[i])^2\\ &= \sum_{i=1}^{j}(\tilde{M}[1,j]+\delta+\delta_i-\tilde{s}[i])^2\end{aligned}$$

Suppose for a moment that we fix $\tilde{M}[1,j]$ and δ_i's, and treat $\|\,\bar{s}[1,j]-\tilde{s}[1,j]\,\|_2$ as a function f over δ. The derivative of $f(\delta)$ is:

$$\begin{aligned}f'(\delta) &= 2\sum_{i=1}^{j}(\tilde{M}[1,j]+\delta+\delta_i-\tilde{s}[i])\\ &= 2\big(j\tilde{M}[1,j]-\sum_{i=1}^{j}\tilde{s}[i]\big)+2j\delta+2\sum_{i=1}^{j}\delta_i\\ &= 2j\delta+2\sum_{i=1}^{j}\delta_i\end{aligned}$$

Since $\delta_i\geq 0$ for all i, then the derivative is strictly greater than zero for any $\delta>0$, which implies that f is a strictly increasing function of δ and has a minimum at $\delta=0$. Therefore, $\|\,\bar{s}[1,j]-\tilde{s}[1,j]\,\|_2=f(\delta)>f(0)=\|\,\bar{s}'[1,j]-\tilde{s}[1,j]\,\|_2$. This is a contradiction since it was assumed that $\bar{s}$ was the minimum solution. This completes the proof for (i).

For (ii), the proof of $\bar{s}[n]\geq max_i\tilde{M}[i,n]$ follows from a similar argument: if $\bar{s}[n]<\tilde{M}[i,n]$ for some i, define $\delta=\tilde{M}[i,n]-\bar{s}[n]$ and the sequence $\bar{s}'$ with elements $\bar{s}'[j]=\bar{s}[j]+\delta$ for $j\geq i$. Then $\bar{s}'$ can be shown to be a strictly better solution than $\bar{s}$, proving (ii).

For the proof of (iii), we first show that $\bar{s}[k]\leq max(\bar{s}[k-1],min_j\tilde{M}[k,j])$. Assume the contrary, i.e. there exists a k such that $\bar{s}[k]>\bar{s}[k-1]$ and $\bar{s}[k]>min_j\tilde{M}[k,j]$. In other words, we assume there exists a k and j such that $\bar{s}[k]>\bar{s}[k-1]$ and $\bar{s}[k]>\tilde{M}[k,j]$. Denote $\delta=\bar{s}[k]-max(\bar{s}[k-1],\tilde{M}[k,j])$. By our assumption above, $\delta>0$. Define the sequence

$$\bar{s}'[i] \;=\; \begin{cases}\bar{s}[i]-\delta & \text{if } k\leq i\leq j\\ \bar{s}[i] & \text{otherwise}\end{cases}$$

Note that by construction, $\bar{s}'[k]=\bar{s}[k]-\delta=\bar{s}[k]-(\bar{s}[k]-max(\bar{s}[k-1],\tilde{M}[k,j]))=max(\bar{s}[k-1],\tilde{M}[k,j])$. It is easy to see that $\bar{s}'$ is sorted (indeed the only inversion in the sort order could have occured if $\bar{s}'[k-1]>\bar{s}'[k]$, but doesn't as $\bar{s}'[k-1]=\bar{s}[k-1]\leq max(\bar{s}[k-1],\tilde{M}[k,j])=\bar{s}'[k]$).

Now a similar argument as in the proof of (i) for the sequence $\tilde{s}[k,j]$, yields that the error $\|\,\bar{s}'[k,j]-\tilde{s}[k,j]\,\|_2<\|\,\bar{s}[k,j]-\tilde{s}[k,j]\,\|_2$. Thus $\|\,\bar{s}'-\tilde{s}\,\|_2<\|\,\bar{s}'-\tilde{s}\,\|_2$ and $\bar{s}'$ is a strictly better solution than $\bar{s}$. This yileds a contradiction as $\bar{s}$ is the minumum L_2 solution. Hence $\bar{s}[k]\leq max(\bar{s}[k-1],min_j\tilde{M}[k,j])$.

A similar argument in the the reverse direction shows that $\bar{s}[k]\geq min(\bar{s}_{k+1},max_i\tilde{M}[i,k])$ completing the proof of (iii). □

A.2 Proof of Theorem 2

We first restate the theorem below. Denote n and d as 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 exists 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 $error(\overline{\mathbf{S}}) = O(d \log^3 n/\epsilon^2)$. In comparison, $error(\tilde{\mathbf{S}}) = \Theta(n/\epsilon^2)$.

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.

Before showing the proof, we prove the following lemma.

LEMMA 2. *Let $s = \mathbf{S}(I)$ be the input sequence. Call a translation of s the operation of subtacting from each element of s a fixed amound δ. Then $error(\overline{\mathbf{S}}[i])$ is invariant under translation for all i.*

PROOF. Denote $Pr(\overline{s}|s)$ $(Pr(\tilde{s}|s))$ the probability that $\overline{s}$ $(\tilde{s})$ is output on the input sequence s. Denote s', $\overline{s}'$, and $\tilde{s}'$ the sequence obtained by translating s, $\overline{s}$, and $\tilde{s}$ by δ, respectively.

First observe that $Pr(\tilde{s}|s) = Pr(\tilde{s}'|s')$ as $\tilde{s}$ and $\tilde{s}'$ are obtained by adding the same Laplacian noise to s and s', respectively. Using Theorem 1 (since all U_k's and L_k's shift by δ on translating $\tilde{s}$ by $delta$), we get that if $\overline{s} = \text{MIN}L_2(\tilde{s}, \gamma_{\mathbf{S}})$, then $\overline{s}' = \text{MIN}L_2(\tilde{s}', \gamma_{\mathbf{S}})$. Thus, $Pr(\overline{s}|s) = Pr(\overline{s}'|s')$ for all sequences $\overline{s}$. Further, since $\overline{s}[i]$ and $\overline{s}'[i]$ yield the same L_2 error with $s[i]$ and $s'[i]$ respectively, we get that the expected $error(\overline{\mathbf{S}}[i])$ is same for both inputs s and s'. □

LEMMA 3. *Let X be any positive random variable that is bounded ($\lim_{x\to\infty} xPr(X > x)$ exists). Then*

$$\mathbb{E}(X) \leq \int_0^{\infty} Pr(X > x)dx$$

PROOF. The proof follows from the following chain of equalities.

$$\begin{aligned}
\mathbb{E}(X) &= \int_0^{\infty} x \frac{\partial}{\partial x} \left(Pr(X \leq x)\right) \\
&= -\int_0^{\infty} x \frac{\partial}{\partial x} \left(Pr(X > x)\right) \\
&= -[xPr(X > x)]_0^{\infty} + \int_0^{\infty} (Pr(X \leq x) - 1)dx \quad \text{(by parts)} \\
&= -\lim_{x \to \infty} xPr(X > x) + \int_0^{\infty} Pr(X > x)dx \\
&\leq \int_0^{\infty} Pr(X > x)dx
\end{aligned}$$

Here the last equality follows as X is bounded and therefore the limit exists and is positive.This completes the proof. □

We next state a theorem that was shown in [8]

THEOREM 5 (THEOREM 3.4 [8]). *Suppose that X_1, X_2, $\ldots$, X_n are independent random variables satisfying $X_i \leq \mathbb{E}(X_i) + M$, for $1 \leq i \leq n$. We consider the sum $X = \sum_{i=1}^{n} X_i$ with expectation $\mathbb{E}(X) = \sum_{i=1}^{n} E(X_i)$ and $Var(X) = \sum_{i=1}^{n} Var(X_i)$. Then, we have*

$$Pr(X \geq \mathbb{E}(X) + \lambda) \leq e^{\frac{-\lambda^2}{2(Var(X) + M\lambda/3)}}$$

For a random variabe X, denote $\mathbb{I}_X$ the indicator function that $X \geq 0$ (thus $\mathbb{I}_X = 1$ if $X \geq 0$ and 0 otherwise). Using Theorem 5, we prove the following lemma.

LEMMA 4. *Suppose i, j are indices such that for all $k \in [i, j]$, $s[k] \leq 0$. Then there exists a constant c such that for all $\tau \geq 1$ the following holds.*

$$Pr\left(\tilde{M}[i,j]^2 \mathbb{I}_{\tilde{M}[i,j]} \geq c(\frac{\log^2((j-i+1)\tau)}{(j-i+1)\epsilon^2})\right) \leq \frac{1}{(j-i+1)^2\tau^2}$$

PROOF. We apply Theorem 5 on $\tilde{s}[k]$ for $k \in [i, j]$. First note that $\mathbb{E}(\tilde{s}[k]) = s[k] \leq 0$. Further $Var(\tilde{s}[k]) = \frac{2}{\epsilon^2}$ as $\tilde{s}[k]$ is obtained by adding Laplace noise to $s[k]$ which has this variance. We also know that $\tilde{s}[k] \geq M + s[k]$ happens with probaility at most $e^{-\epsilon M}/2$.

For simplicity, call n to be $j - i + 1$. Denoting $X = \sum_{k \in [i,j]} \tilde{s}[k]$, we see that $\mathbb{E}(X) \leq 0$ and $Var(X) = \frac{2n}{\epsilon^2}$. Further, set $M = 3\log(n\tau)/\epsilon$. Denote B the event that for some k, $\tilde{s}[k] \geq M + s[k]$. Thus $Pr(B) \leq ne^{-\epsilon M}/2 \leq \frac{1}{2n^2\tau^3}$. If B does not happen, we know that $\tilde{s}[k] \leq M + s[k]$ for all $k \in [i, j]$. Thus we can then apply Theorem 5 to get:

$$\begin{aligned}
Pr\left(X \geq \mathbb{E}(X) + \lambda\right) &\leq e^{\frac{-\lambda^2}{2(2n/\epsilon^2 + \lambda\log(n\tau)/\epsilon)}} + Pr(B) \\
&= e^{\frac{-\lambda^2}{2(2n/\epsilon^2 + \lambda\log(n\tau)/\epsilon)}} + \frac{1}{2n^2\tau^3}
\end{aligned}$$

Setting $\lambda = \frac{8}{\epsilon}\sqrt{n}\log(n\tau)$ gives us that

$$Pr\left(X \geq \mathbb{E}(X) + \frac{8}{\epsilon}\sqrt{n}\log(n\tau)\right) \leq \frac{1}{n^2\tau^2}$$

Since $\mathbb{E}(X) \leq 0$, we get

$$Pr\left(X \geq \frac{8}{\epsilon}\sqrt{n}\log(n\tau)\right) \leq \frac{1}{n^2\tau^2}$$

Also we observe that $\tilde{M}[i,j] = X/n$, which yields

$$Pr\left(\tilde{M}[i,j] \geq \frac{8\log(n\tau)}{\sqrt{n}\epsilon}\right) \leq \frac{1}{n^2\tau^2}$$

Finally, observe that $\tilde{M}[i,j] \leq c$ implies that $\tilde{M}[i,j]^2 \mathbb{I}_{\tilde{M}[i,j]} \leq c^2$. Thus we get

$$Pr\left(\tilde{M}[i,j]^2 \mathbb{I}_{mm[i,j]} \geq \frac{64\log^2(n\tau)}{n\epsilon^2}\right) \leq \frac{1}{n^2\tau^2}$$

Putting $n = j - i + 1$ and using $c = 64$ gives us the required result. □

Now we can give the proof of Thereom 2.

PROOF OF THEOREM 2. The proof of $error(\tilde{\mathbf{S}}) = \Theta(n/\epsilon^2)$ is obvious since:

$$error(\tilde{\mathbf{S}}) = \sum_{k=1}^{n} error(\tilde{s}[i]) = n(\frac{2}{\epsilon^2})$$

In the rest of the proof, we shall show bound $error(\overline{\mathbf{S}})$. Let $s = S(I)$ be the input sequence. We know that s consists of d distinct elements. Denote s_r as the r^{th} distinct element of s. Also denote $[l_r, u_r]$ as the set of indices corresponding to s_r, i.e. $\forall_{i \in [l_r,u_r]} s[i] = s_r$ and $\forall_{i \notin [l_r,u_r]} s[i] \neq s_r$. Let $M[i,j]$ record the mean of elements in $s[i,j]$, i.e. $M[i,j] = \sum_{k=i}^{j} s[k]/(i-j+1)$.

To bound $error(\overline{\mathbf{S}})$, we shall bound $error(\overline{\mathbf{S}}[i])$ separately for each i. To bound $error(\overline{\mathbf{S}}[i])$, we can assume W.L.O.G that $s[i]$ is 0. This is because if $s[i] \neq 0$, then we can translate the sequence s by $s[i]$. As shown in Lemma 2 this preserves $error(\overline{\mathbf{S}}[i])$, while making $s[i] = 0$.

Let $k \in [l_r, u_r]$ be any index for the r^{th} distinct element of s. By definition, $error(\overline{\mathbf{S}}[k]) = \mathbb{E}(\overline{s}[k] - s[k])^2 = \mathbb{E}(\overline{s}[k]^2)$ (as we can assume W.L.O.G $s[k] = 0$). From Theorem 1, we know that $\overline{s}[k] = U_k$. Thus $error(\overline{\mathbf{S}}[k]) = \mathbb{E}(U_k^2)$. Here we treat $U_k = max_{i \leq k} min_j \tilde{M}[i,j]$ as a random variable. Now by definition of $\mathbb{E}$, we have

$$\mathbb{E}(U_k^2) = \mathbb{E}(U_k^2 \mathbb{I}_{U_k}) + \mathbb{E}(U_k^2(1 - \mathbb{I}_{U_k})) = A + B \text{ (say)}$$

We shall bound A and B separately. For bounding A, denote $\mathcal{U}_k = max_{i \leq k} \tilde{M}[i, u_r]$. It is apparent that $\mathcal{U}_k \geq U_k$ and thus $\mathcal{U}_k^2 \mathbb{I}_{\mathcal{U}_k} \geq U_k^2 \mathbb{I}_{U_k}$. To bound A, we observe that

$$A \;=\; \mathbb{E}(U_k^2 \mathbb{I}_{U_k}) \leq \mathbb{E}(\mathcal{U}_k^2 \mathbb{I}_{\mathcal{U}_k})$$

Further, since $\mathcal{U}_k = max_{i \leq k} \tilde{M}[i, u_r]$, we know that $\mathcal{U}_k^2 \mathbb{I}_{\mathcal{U}_k} = max_{i \leq k} \tilde{M}[i, u_r]^2 \mathbb{I}_{\tilde{M}[i,u_r]}$. Thus we can write:

$$A \;\leq\; \mathbb{E}(\mathcal{U}_k^2 \mathbb{I}_{\mathcal{U}_k}) = \mathbb{E}\left(max_{i \leq k} \tilde{M}[i, u_r]^2 \mathbb{I}_{\tilde{M}[i,u_r]} \right)$$

Let $\tau > 1$ be any number and c be the constant used in Lemma 4. Let us denote e_i the event that:

$$\tilde{M}[i, u_r]^2 \mathbb{I}_{\tilde{M}[i,u_r]} \geq c(\frac{\log^2((u_r - i + 1)\tau)}{(u_r - i + 1)\epsilon^2})$$

We can apply lemma 4 to compute the probability of e_i as $s[j] \leq 0$ for all $j \leq u_r$ (as we assumed W.L.O.G $s[k] = 0$). Thus we get $Pr(e_i) \leq \frac{1}{(u_r - i + 1)^2 \tau^2}$.

Define $e = \vee_{i=1}^{u_r} e_i$. Then $Pr(e) \leq \sum_{i=1}^{u_r} Pr(e_i) = 2/\tau^2$ (as $\sum_{i=1}^{u_r} 1/i^2 \leq 2$). If the event e does not happen, then it is easy to see that

$$\begin{aligned} \mathcal{U}_k^2 \mathbb{I}_{\mathcal{U}_k} &= max_{i \leq k} \tilde{M}[i, u_r]^2 \mathbb{I}_{\tilde{M}[i,u_r]} \\ &\leq c(\frac{\log^2((u_r - k + 1)\tau)}{(u_r - k + 1)\epsilon^2}) \end{aligned}$$

Thus with at least probability $1 - 2/\tau^2$ (which is $Pr(\neg e)$), we get $\mathcal{U}_k^2 \mathbb{I}_{\mathcal{U}_k}$ is bounded as above. This yields that there exist constants c_1 and c_2 such that $\mathbb{E}(\mathcal{U}_k^2 \mathbb{I}_{\mathcal{U}_k}) \leq \frac{c_1 \log^2(u_r - k + 1) + c_2}{(u_r - k + 1)\epsilon^2}$. The proof is by the application of Lemma 3 (as $\mathcal{U}_k$ is bounded) and a simple integration over τ ranging from 1 to ∞. Finally we get that $A \leq \mathbb{E}(\mathcal{U}_k^2 \mathbb{I}_{\mathcal{U}_k}) \leq \frac{c_1 \log^2(u_r - k + 1) + c_2}{(u_r - k + 1)\epsilon^2}$.

Recall that $B = \mathbb{E}(U_k^2(1 - \mathbb{I}_{U_k}))$. We can write B as $\mathbb{E}(L_k^2(1 - \mathbb{I}_{L_k}))$ as $L_k = U_k$. Using the exact same arguments as above for L_k but on sequence $-\mathbf{S}$ yields that $B \leq \frac{c_1 \log^2(k - l_r + 1) + c_2}{(k - l_r + 1)\epsilon^2}$.

Finally, we get that $\overline{\mathbf{S}}[k] = A + B$ which is less than $\frac{c_1 \log^2(u_r - k + 1) + c_2}{(u_r - k + 1)\epsilon^2} + \frac{c_1 \log^2(k - l_r + 1) + c_2}{(k - l_r + 1)\epsilon^2}$.

To obtain a bound on the total $error(\overline{\mathbf{S}})$.

$$\begin{aligned} error(\overline{\mathbf{S}}) &= \sum_{r=1}^{d} \sum_{k \in [l_r, u_r]} error(\overline{\mathbf{S}}[k]) \\ &\leq \sum_{r=1}^{d} \sum_{k \in [l_r, u_r]} \frac{c_1 \log^2(u_r - k + 1) + c_2}{(u_r - k + 1)\epsilon^2} + \\ & \sum_{r=1}^{d} \sum_{k \in [l_r, u_r]} \frac{c_1 \log^2(k - l_r + 1) + c_2}{(k - l_r + 1)\epsilon^2} \\ &\leq \sum_{r=1}^{d} \frac{c_1 \log^3(u_r - l_r + 1) + c_2}{\epsilon^2} \end{aligned}$$

Finally noting that $u_r - l_r + 1$ is just n_r, the number of occurences of s_r in s, we get $error(\overline{\mathbf{S}}) = \sum_r \frac{c_1 \log^3 n_r + c_2}{\epsilon^2} = O(d \log^3 n / \epsilon^2)$. This completes the proof of the theorem. □

B. HIERARCHICAL HISTOGRAMS

B.1 Proof of Proposition 2

We first restate the proposition below. Let $succZ[u] = \sum_{w \in succ(u)} z[w]$.

PROPOSITION 2. *Given the noisy count sequence $\tilde{h} = \mathbf{H}(I)$, the unique minimum L_2 solution, $\overline{h} = \text{MIN}L_2(\tilde{h}, \gamma_{\mathbf{H}})$, is given by the following recurrence relation (denoting $u = pred(v)$):*

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

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.

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

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.

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

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.

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

The above set of equations can be rewritten as (since $\sum_{l' \in leaves(v)} \overline{h}[l'] = \overline{h}[v]$):

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

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 $level(h-1)$ and leaves in $level(0)$. Abbreviating LHS (RHS) for the left (right) hand side of the above equation, we observe the following.

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

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.

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

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.

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

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

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

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.

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

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

$$\begin{aligned} &\text{minimize} \quad \sum_{w \in subtree(U)} \left((\sum_{l \in leaves(w)} \overline{h}[l]) - \tilde{h}[w] \right)^2 \\ &\text{subject to} \sum_{l \in leaves(u)} \overline{h}[u] = H \end{aligned}$$

We can now use the method of Lagrange mulitpliers 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.

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

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. □

B.2 Proof of Theorem 3

Theorem 3 is restated below.

THEOREM 3. *If $\overline{\mathbf{H}}$ is defined as $\textsc{min}L_2(\tilde{\mathbf{H}}(I), \gamma_{\mathbf{H}})$ then $\overline{\mathbf{H}}$ satisfies the following properties: (i) $\overline{\mathbf{H}}$ is a linear unbiased estimator, (ii) $error(\overline{\mathbf{H}})$ is the minimal among the $error(E)$ of all linear unbiased estimators E.*

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 estimaors. 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* proving (ii). □

B.3 Proof of Theorem 4

First, the theorem is restated.

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

Proof. For (i), we use the fact that $\overline{\mathbf{H}}$ is the OLS estimator. Hence it minimizes the *error* for estimating any linear combination of the original counts, which includes in particular the given range query q.

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

$$error(HC[q]) \leq kh(2h^2/\epsilon^2) = O(h^3/\epsilon^2)$$

For (iii), 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 $6h^2/\epsilon^2$. But, on the other hand, $\tilde{\mathbf{H}}$ will require summing $2(k-1)(h-1)$ noisy counts in total (2(k-1) at each level of the tree). Thus $error(\tilde{\mathbf{H}}_q) = 4(k-1)(h-1)h^2/\epsilon^2$. Thus

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

This completes the proof. □