A Multiplicative Weights Mechanism for Privacy-Preserving Data Analysis

Moritz Hardt
Center for Computational Intractability
Department of Computer Science
Princeton University
Email: mhardt@cs.princeton.edu

Guy N. Rothblum
Center for Computational Intractability
Department of Computer Science
Princeton University
Email: rothblum@alum.mit.edu

***Abstract*—We consider statistical data analysis in the interactive setting. In this setting a trusted curator maintains a database of sensitive information about individual participants, and releases privacy-preserving answers to queries as they arrive. Our primary contribution is a new differentially private multiplicative weights mechanism for answering a large number of interactive counting (or linear) queries that arrive online and may be adaptively chosen.**

This is the first mechanism with worst-case accuracy guarantees that can answer large numbers of interactive queries and is *efficient* (in terms of the runtime's dependence on the data universe size). The error is asymptotically *optimal* in its dependence on the number of participants, and depends only logarithmically on the number of queries being answered. The running time is nearly *linear* in the size of the data universe.

As a further contribution, when we relax the utility requirement and require accuracy only for databases drawn from a rich class of databases, we obtain exponential improvements in running time. Even in this relaxed setting we continue to guarantee privacy for *any* input database. Only the utility requirement is relaxed. Specifically, we show that when the input database is drawn from a *smooth* distribution — a distribution that does not place too much weight on any single data item — accuracy remains as above, and the running time becomes *poly-logarithmic* in the data universe size.

The main technical contributions are the application of multiplicative weights techniques to the differential privacy setting, a new privacy analysis for the interactive setting, and a technique for reducing data dimensionality for databases drawn from smooth distributions.

I. Introduction

Statistical analysis of sensitive information about individuals comes with important benefits. However, since the results of the analysis are often made public, these benefits might come at a serious cost to the privacy of individuals' sensitive data. A recent line of work, starting with the seminal works of Dinur and Nissim [3] and Dwork and Nissim [9] aims to provide a rigorous mathematical foundation for protecting the privacy of individuals in the setting of statistical analysis. This research has yielded the robust privacy guarantee of *differential privacy*, due to Dwork *et al.* [5], which guarantees that the outcome of the analysis on adjacent databases (databases that differ only in one participant's information) is "very similar" (in a strong sense). In particular, differential privacy guarantees that participation in the analysis does not incur significant additional risk for individuals (vs. non-participation). Throughout this paper and most of the prior work, the focus is on the setting where a trusted curator, holding a database of potentially sensitive information about n individuals, wishes to release statistics about the data while protecting individuals' privacy.

A central question in this line of research regards the tradeoff between utility and privacy. Namely, *what kinds of statistical queries can be answered, and with what accuracy, while protecting the privacy of individuals?* Early results seemed mixed: on one hand, moderate numbers of queries (smaller than the number of individuals in the database) could be answered with differential privacy and excellent accuracy by adding independent noise to the answers [3], [9], and a growing body of subsequent work. On the other hand, it was shown that there exist specific families of simple queries such that answering "too many" of the queries (more than the database size) with "overly accurate" responses (smaller error than the sampling error[1]) leads to blatant privacy violations [3], [6], [11]. Still, these negative results left open the possibility that very rich statistical analyses, say answering huge families of queries, could be run in a privacy-preserving way, as long as their accuracy was slightly worse than the sampling error.

Non-interactive mechanisms: A beautiful work of Blum, Ligett and Roth [2] showed that huge numbers of queries could in fact be answered in a privacy-preserving way. They considered *counting queries*: counting what fraction (between 0 and 1) of the participants in the analysis satisfy some property, where the "property" is taken to be a boolean predicate over the data universe U. They designed a privacy-preserving mechanism where for any set $\mathcal{C}$ of counting queries specified non-interactively (i.e. in advance), the error scales only *logarithmically* with the number of queries being answered. Specifically, the error's dependence on the database size n and query set size k was roughly $(1/n^{1/3}) \cdot \log k$ (we ignore for now, and throughout the

[1]The "sampling error" we refer to throughout the introduction is the error incurred by inferring statistical information about an underlying distribution of the population from n samples. This error is inherent to statistical data analysis. In fractional additive terms, the sampling error is asymptotically roughly $\tilde{O}(1/\sqrt{n})$ w.h.p.

introduction, the constants and the many other parameters in the error expression). Moreover, the mechanism's output was a *synthetic database*: a (privacy-preserving) database of entries from the data universe U, where for each counting query the fraction of participants who satisfy it in the input and output database is within the error bound. This is a useful output format, as it is compatible with existing tools for analyzing databases and guarantees consistency. While this result showed that (information theoretically at least) massive query sets could be answered in a privacy preserving way, it suffered from several disadvantages. First, this was a *non-interactive* mechanism: the query set had to be specified in advance, whereas previous mechanisms such as [9] were *interactive*, and allowed for answering arbitrary queries specified in an interactive and adaptive manner.[2] Moreover, the mechanism's error was significantly larger than the $1/\sqrt{n}$ sampling error. This meant that, for fixed fractional accuracy, the number of participants in a data analysis needed to be significantly higher. Finally, the running time was very high: super-polynomial in the size $N = |U|$ of the data universe, and in k, the query set size.

These last two concerns were considered in a work of Dwork *et al.* [8]. They gave a non-interactive mechanism, whose running time was *polynomial* in N and k, where the error's dependence on the query set size grew (roughly) as $(1/\sqrt{n}) \cdot k^{o(1)}$. This output was also a synthetic database. They showed that, under strong enough (exponential) cryptographic hardness assumptions, no general mechanism for answering counting queries that outputs synthetic data could have sublinear running-time in N or in k. In later work, Dwork, Rothblum and Vadhan [10] obtained a mechanism with similar running time whose error was $(1/\sqrt{n}) \cdot \mathrm{polylog} k$. Moreover, they showed how to obtain similar error bounds for *arbitrary* low-sensitivity queries (previous work was restricted to counting queries), for general queries the running time was no longer polynomial in N and k. Both of these mechanisms provide a slightly relaxed privacy guarantee known as (ε, δ)-differential privacy [5].

Interactive Mechanisms: For the *interactive* setting, where the queries are specified in an interactive and adaptive manner, it remained unclear whether large numbers of queries could be answered accurately in a privacy-preserving way. In a beautiful recent work, Roth and Roughgarden [13] presented a new mechanism for answering *interactive* counting queries, whose error scaled as $(1/n^{1/3}) \cdot \mathrm{polylog}(k)$. They gave a super-polynomial time (in N and k) mechanism, and a separate polynomial-time mechanism that guaranteed similar error bound w.h.p. over a database drawn from a random distribution.

[2]Note that in this setting, all other things being equal, an interactive mechanism is preferable to a non-interactive one: if we have an interactive mechanism, even if the queries are all specified in advance, we can still run the interactive mechanism on the queries, one by one, and obtain privacy-preserving answers to all of them.

Several important questions remained unanswered, even for the case of counting queries:

1) Is there a *polynomial-time* interactive mechanism (i.e., one that runs in time $\mathrm{poly}(N)$ on each of the k queries) with non-trivial error on all databases?
2) Could its error scale to the sampling error ${}^{1}/\sqrt{n}$ and grow only logarithmically with the number of queries k?
3) Given the negative results of [8], we cannot hope for sub-linear running time in N. Do there exist mechanisms that match or nearly-match this hardness result?
4) What are open avenues for side-stepping the negative results of [8]? Namely, are there meaningful relaxations that permit mechanisms whose running time is sub-linear or even poly-logarithmic in N?

A. This Work

Our main contribution is a new privacy-preserving interactive mechanism for answering counting queries, which we will refer to as the private multiplicative weights (PMW) mechanism. It allows us to give positive answers to the first three questions above, and to make partial progress on the last question. We proceed with a summary of our contributions. Throughout this section, when we refer to a mechanism's running time as being polynomial or linear, we are measuring the running time as a function of the data universe size N (which may be quite large for high-dimensional data).

Linear-Time Interactive Mechanism: The PMW mechanism runs in linear-time and provides a worst-case accuracy guarantees for *all* input databases. The mechanism is presented Figure 1, its performance stated in the theorem below. The proof is in Section IV. See Section III for the formal definitions of accuracy and differential privacy for interactive mechanisms.

Theorem I.1. *Let U be a data universe of size N. For any $k, \varepsilon, \delta, \beta > 0$, the Private Multiplicative Weights Mechanism of Figure 1, is an (ε, δ)-differentially private interactive mechanism. For any database of size n, the mechanism is (α, β, k)-accurate for (adaptive) counting queries over U, where $\alpha = O\left(\varepsilon^{-1} n^{-1/2} \cdot \log(1/\delta) \log^{1/4}(N) \cdot (\log k + \log(1/\beta))\right)$.*
The running time in answering each query is $N \cdot \mathrm{poly}(n) \cdot \mathrm{polylog}(1/\beta, 1/\varepsilon, 1/\delta)$.

The error (as a function of n and k) grows roughly as $(1/\sqrt{n}) \cdot \log k$. In particular, this shows that even in the interactive setting, differential privacy permits error (beyond the $1/\sqrt{n}$ lower bound of [3]) that grows only logarithmically with the number of queries being answered. Moreover, the running time is only *linear* in N (for each of the k queries), nearly tight with the cryptographic hardness results of [8]. Previous work (even in the non-interactive setting) had higher polynomial running time. Finally, we remark that this mechanism can also be used to generate a synthetic database with similar error and running time bounds (in the non-interactive setting), see below for this extensions.

Relaxed Notions of Utility: To answer Question 4 that was raised in the introduction, we begin with a discussion of the negative results of [8] and possible avenues for side-stepping them. The negative results for producing synthetic data can be side-stepped by a mechanism whose output has a different format. This is a promising avenue, but synthetic data is a useful output format. It is natural to try to side-step hardness while continuing to output synthetic data. One possibility is working for restricted query classes, but recent work of Ullman and Vadhan [14] shows hardness even for very simple and natural query classes such as conjunctions. In the known hardness results, however, the *databases* (or rather database distributions) that are hard to sanitize are (arguably) "unnatural", containing cryptographic data in [8] and PCP proofs for the validity of digital signatures in [14]. Thus, a natural approach to side-stepping hardness is relaxing the utility requirement, and not requiring accuracy for *every* input database.

A mechanism that works only for some input databases is only as interesting as the class of databases for which accuracy is guaranteed. For example, getting accuracy w.h.p. for *most* databases is simple, since (speaking loosely and informally) *most* databases behave like a uniformly random database. Thus, we can get privacy and accuracy by ignoring the input database (which gives perfect privacy) and answering according a new database drawn uniformly at random (which, for most input databases, will give fairly accurate answers).

Smooth databases and sublinear time: We consider accuracy guarantees for the class of *(pseudo)-smooth* databases. Intuitively, we think of these as databases sampled i.i.d. from *smooth* underlying distributions over the data universe U. I.e., underlying distributions that do not put too much weight on any particular data item (alternatively, they have high min-entropy). We say that a histogram or distribution y over U is *ξ-smooth*, if for every $u \in U$, the probability of u by y is at most ξ. We say that a histogram or database $x \in U^n$ is *(ξ, ϕ)-pseudo-smooth w.r.t a set $\mathcal{C}$ of queries* if there exists some ξ-smooth y that approximates it well w.r.t every query in $\mathcal{C}$. I.e., for every $f \in \mathcal{C}$, $|f(y) - f(x)| \leq \phi$ (where by $f(y)$ we mean the expectation of f over data items drawn from y). See Section V for formal definitions.

The PMW mechanism yields a mechanism with improved running time—sub-linear, or even polylogarithmic in N—for pseudo-smooth databases. The new mechanism (with smoothness parameter ξ) runs in time that depends linearly on ξN rather than N. It guarantees differential privacy for *any* input database. Its error is similar to that of the mechanism of Theorem I.1 (up to an additional ϕ error), but this accuracy guarantee is only: (i) for a set $\mathcal{C}$ of interactive counting queries that are fixed in advance (i.e. non-adaptively). We note that the mechanism is interactive in the sense that it need not know the queries in advance, but accuracy is not guaranteed for adversarially chosen queries (see the discussion in Section II for motivation for this relaxation), and (ii) for input databases that are (ξ, ϕ)-smooth with respect to the query class $\mathcal{C}$. The performance guarantees are in Theorem I.2 below. The proof is in Section V

Theorem I.2 (Smooth PMW). *Let U be a data universe of size N. For any $\varepsilon, \delta, \beta, \xi, \phi > 0$, the Private Multiplicative Weights Mechanism of Figure 1 is an (ε, δ)-differentially private interactive mechanism. For any sequence $\mathcal{C}$ of k interactive counting queries over U that are fixed in advance (non-adaptively), for any database of size n that is (ξ, ϕ)-pseudo-smooth w.r.t $\mathcal{C}$, the mechanism is (α, β, k)-non-adaptively accurate w.r.t. $\mathcal{C}$, where $\alpha = \tilde{O}\left(\phi + \varepsilon^{-1} n^{-1/2} \log(1/\delta) \log^{1/4}(\xi N) \cdot (\log k + \log(1/\beta))\right)$ The running time in answering each query is $(\xi N) \cdot \mathrm{poly}(n) \cdot \mathrm{polylog}(1/\beta, 1/\varepsilon, 1/\delta, 1/\xi, 1/\phi)$.*

In particular, for very good smoothness $\xi = \mathrm{polylog} N/N$, the running time will depend only poly-logarithmically on N. The main observation for achieving this improved running time is that for (pseudo)-smooth databases we can effectively reduce the data universe size by sub-sampling, and then apply our algorithm to the smaller data universe. The mechanism does not require knowledge of the histogram which certifies that the given input database is pseudosmooth.

The privacy guarantee is the standard notion of differential privacy. I.e., privacy holds always and for every database. The accuracy guarantee is only for pseudosmooth databases, and we interpret it as follows. The dataset is drawn i.i.d from an unknown underlying distribution D (the standard view in statistics). The mechanism guarantees accuracy and sublinear efficiency as long as the underlying data distribution is smooth. If the underlying distribution is ξ-smooth, then w.h.p. the database x (which we think of as being drawn i.i.d from D and of large enough size) is "close" to D on every query $f \in \mathcal{C}$, and so w.h.p. x is (ξ, ϕ)-smooth and the mechanism is accurate. An important aspect of this guarantee is that *there is no need to know what the underlying distribution is*, only that it is smooth. A promising approach in practice may be to run this mechanism as a very efficient heuristic. The heuristic *guarantees* privacy, and also has a rigorous accuracy guarantee under assumptions about the underlying distribution. We note that Dwork and Lei [4] also proposed mechanisms that always guarantee privacy, but guarantee accuracy only for a subset of databases (or underlying distributions).

We also note that [13] considered databases drawn from a distribution that was itself picked randomly from the set of all distributions. Such "random distributions" are indeed very smooth (w.h.p. $\xi \leq O(\log N/N)$) and therefore a special case of our model.

An interesting direction for future work is finding differentially private mechanisms for other and more useful or well motivated families of databases, or finding natural applications where pseudo-smooth databases are of particular interest. We note that (as one would expect given these positive results) the negative results for producing synthetic data are for databases that are neither smooth nor pseudo-smooth.

II. Overview of proof and techniques

Multiplicative Weights: We use a (privacy-preserving) multiplicative weights mechanism (see [12], [1]). The mechanism views databases as histograms or distributions (also known as "fractional" databases) over the data universe U (as was done in [8]). At a high level, the mechanism works as follows. The real database being analyzed is x (we view x as distribution or histogram over U, with positive weight on the data items in x). The mechanism also maintains an updated fractional database, denoted as x_t at the end of round t. In each round t, after the t-th counting query f_t has been specified, x_{t-1} is updated to obtain x_t. The initial database x_0 is simply the uniform distribution over the data universe. I.e., each coordinate $u \in U$ has weight $1/N$.

In the t-th round, after the t-th query f_t has been specified, we compute a noisy answer $\widehat{a}_t$ by adding (properly scaled) Laplace noise to $f_t(x)$—the "true" answer on the real database. We then compare this noisy answer with the answer given by the previous round's database $f_t(x_{t-1})$. If the answers are "close", then this is a "lazy" round, and we simply output $f_t(x_{t-1})$ and set $x_t \leftarrow x_{t-1}$. If the answers are "far", then this is an "update" round and we need to update or "improve" x_t using a multiplicative weights re-weighting. The intuition is that the re-weighting brings x_t "closer" to an accurate answer on f_t. In a nutshell, this is all the algorithm does. The only additional step required is bounding the number of "update" rounds: if the total number of update rounds grows to be larger than (roughly) n, then the mechanism fails and terminates. This will be a low probability event. See Figure 1 for the details. Given this overview of the algorithm, it remains to specify how to: (i) compute $f_t(x_{t-1})$, and (ii) re-weight or improve the database on update rounds. We proceed with an overview of the arguments for accuracy and privacy.

For this exposition, we think of the mechanism as explicitly maintaining the x_t databases, resulting in complexity that is roughly linear in $N = |U|$. Using standard techniques we can make the memory used by the mechanism logarithmic in N (computing each coordinate of x_t as it is needed). Either way, it is possible to compute $f_t(x_{t-1})$ in linear time.

The re-weighting (done only in update rounds), proceeds as follows. If in the comparison we made, the answer according to x_{t-1} was "too small", then we increase by a small multiplicative factor the weight of items $u \in U$ that satisfy the query f_t's predicate, and decrease the weight of those that do not satisfy it by the same factor. If the answer was "too large" then do the reverse in terms of increasing and decreasing the weights. We then normalize the resulting weights to obtain a new database whose entries sum to 1. The intuition, again, is that we are bringing x_t "closer" to an accurate answer on f_t. The computational work scales linearly with N.

To argue accuracy, observe that as long as the number of update rounds stays below the (roughly n) threshold, our algorithm ensures bounded error (assuming the Laplace noise we add is not too large). The question is whether the number of update rounds remains small enough. This is in fact the case, and the proof is via a multiplicative weights potential argument. Viewing databases as distributions over U, we take the *potential* of database y to be the relative entropy $\mathrm{RE}(x||y)$ between y and the real database x. We show that if the error of x_{t-1} on query f_t is large (roughly larger than $1/\sqrt{n}$), then the potential of the re-weighted x_t is smaller by at least (roughly) $1/n$ than the potential of x_{t-1}. Thus, in every "update" round, the potential drops, and the drop is significant. By bounding the potential of x_0, we get that the number of update rounds is at most (roughly) n.

"Pay as you go" privacy analysis: At first glance, privacy might seem problematic: we access the database and compute a noisy answer *in every round*. Since the number of queries we want to answer (number of rounds) might be huge, unless we add a huge amount of noise this collection of noisy answers *is not privacy preserving*. The point, however, is that in most rounds we don't release the noisy answer. All we do is check whether or not our current database x_{t-1} is accurate, and if so we use it to generate the mechanism's output. In all but the few update rounds, the perturbed true answer is not released, and we want to argue that privacy in all those lazy rounds comes (essentially) "for free". The argument builds on ideas from privacy analyses in previous works [8], [7], [13]).

A central concern is arguing that the "locations" of the update rounds be privacy-preserving (there is an additional, more standard, concern that the noisy answers in the few update rounds also preserve privacy). Speaking intuitively (and somewhat inaccurately), for any two adjacent databases, there are w.h.p. only roughly n "borderline" rounds, where the noise is such that on one database this round is update and on another this round is lazy. This is because, conditioning on a round being "borderline", with constant probability it is actually an "update" round. Since the number of update rounds is at most roughly n, with overwhelming probability the number of borderline rounds also is roughly n. For non-borderline rounds, those rounds' being an update or a lazy round is determined similarly for the two databases, and so privacy for these rounds come "for free". The borderline rounds are few, and so the total privacy hit incurred for them is small.

Given this intuition, we want to argue that the "privacy loss", or "confidence gain" of an adversary, is small. At a high level, if we bound the worst-case confidence gain in each update round by roughly $O(\varepsilon/\sqrt{n})$, then by an "evolution of confidence" argument due to [3], [9], [10], the total confidence gain of an adversary over the roughly n update rounds will be only ε w.h.p. To bound the confidence gain, we define "borderline" rounds as an event over the noise values on a database x, and show that: (1) Conditioned on a round being borderline on x, it will be an update round on x w.h.p. This means borderline rounds are few. (2) Conditioned on a round being borderline on x, the worst-case confidence gain of an adversary viewing the mechanism's behavior in this round on x vs. an adjacent x' is bounded by roughly $\varepsilon/\sqrt{n}$. This means the privacy hit in borderline rounds isn't too large, and we can "afford" roughly n of them. (3) Conditioned on a round *not* being borderline, there is no privacy loss in this round on x vs. any adjacent x'. I.e., non-borderline rounds come for free (in terms of privacy).

This analysis allows us to add less noise than previous works, while still maintaining (ε, δ) differential privacy. It may find other applications in interactive or adaptive privacy settings. Details are in Section IV-B.

Sublinear Time Mechanism for Smooth Databases: We observe that we can modify the PMW mechanism to work over a smaller data universe $V \subseteq U$, as long as there *exists* a database x^* whose support is only over V, and gives close answers to those of x on every query we will be asked. We modify the algorithm to maintain multiplicative weights only over the smaller set V, and increase slightly the inaccuracy threshold for declaring a round as "update". For the analysis, we modify the potential function: it measures relative entropy to x^* rather than x. In update rounds, the distance between x_{t-1} and this new x^* on the current query is large (since x^* is close to x, and x_{t-1} is far from x). This means that re-weighting will reduce $\mathrm{RE}(x^*||x_{t-1})$, and even though we maintain multiplicative weights only over a smaller set V, the number of update rounds will be small. Maintaining multiplicative weights over V rather than U reduces the complexity from linear in $|U|$ to linear in $|V|$.

To use the above observation, we argue that for any large set of counting queries $\mathcal{C}$ and any (ξ, ϕ)-pseudosmooth database x, if we choose a uniformly random small (but not too small) sub-universe $V \subseteq U$, then w.h.p there *exists* x^* whose support is in V that is close to x on all queries in $\mathcal{C}$. In fact, sampling a sub=universe of size roughly $\xi N \cdot n \cdot \log|\mathcal{C}|$ suffices. This means that indeed PMW can be run on the reduced data universe V with reduced computational complexity. See Section V-A for this argument.

Utility here is for a fixed non-adaptive set $\mathcal{C}$ of queries (that need not be known in advance). We find this utility guarantee to still be well motivated—note that, privacy aside, the input database itself, which is sampled i.i.d from an underlying distribution, isn't guaranteed to yield good answers for adaptively chosen queries. Finally, we remark that this technique for reducing the data universe size (the data dimensionality) may be more general than the application to PMW. In particular, previous mechanisms such as [8], [10] can also be modified to take advantage of this sampling and obtain improved running time for smooth databases (the running time will be polynomial, rather than linear as it is for the PMW mechanism).

Synthetic databases: We conclude by noting that the PMW mechanism can be used to generate synthetic data (in the non-interactive setting). To do this, iterate the mechanism over a set of queries $\mathcal{C}$, repeatedly processing all the queries in $\mathcal{C}$ and halting when either (i) we made roughly $n+1$ iterations, i.e. have processed every query in $\mathcal{C}$ n times, or (ii) we have made a complete pass over all the queries in $\mathcal{C}$ without any update rounds (whichever of these two conditions occurs first). If we make a complete pass over $\mathcal{C}$ without any update rounds, then we know that the x_t we have is accurate for all the queries in $\mathcal{C}$ and we can release it (or a subsample form it) as a privacy-preserving synthetic database. By the potential argument, there can be at most roughly n update rounds. Thus, after $n+1$ iterations we are guaranteed to have a pass without any update rounds. Previous mechanisms for generating synthetic databases involved linear programming and were more expensive computationally.

III. Preliminaries

Let $x, y \in \mathbb{R}^N$. We define the *relative entropy* between x and y as $\mathrm{RE}(x||y) = \sum_{i\in[N]} x_i \log\left(\frac{x_i}{y_i}\right) + y_i - x_i$. This reduces to the more familiar expression $\sum_i x_i \log(\frac{x_i}{y_i})$ when $\sum_i x_i = \sum_i y_i = 1$ (in particular this happens when x, y correspond to distributions over $[N]$). We let $Lap(\sigma)$ denote the one-dimensional Laplacian distribution centered at 0 with scaling σ and corresponding density $f(x) = \frac{1}{2\sigma}\exp\left(-\frac{|x|}{\sigma}\right)$. We denote by $\langle x, y\rangle = \sum_{i\in[N]} x_i y_i$ the real valued inner product between two vectors $x, y \in \mathbb{R}^N$. When $x \in \mathbb{R}^S$ is a vector supported on a subset of the coordinates $S \subseteq [N]$ and $y \in \mathbb{R}^N$, we still write $\langle x, y\rangle = \sum_{i\in S} x_i y_i$.

Histograms and linear queries: A *histogram* $x \in \mathbb{R}^N$ represents a database or data distribution over a universe U of size $|U| = N$. We will assume that x is normalized so that $\sum_{i\in U} x_i = 1$. We use histograms in the natural way to denote standard databases of size n (n-item multisets in U), and also to denote distributions over the data universe. The only difference is that databases have support size n, whereas distributions do not necessarily have small support.

In this work we focus on *linear queries* $f\colon \mathbb{R}^N \to [0,1]$. As usual we may view a linear query as a vector $f \in [0,1]^N$. We then use the equality $f(x) = \langle f, x\rangle$, where the histogram

x can be either an n-item database or a data distribution. By our normalization of x, the sensitivity of a linear query is $1/n$. While we assume that $f_t \in [0,1]^N$, our algorithm applies to any linear query $f_t \in [-c,c]^N$ by considering the query defined as $1/2 + f_t[i]/2c$ in coordinate i. In this case, the error of the algorithm scales linearly in c.

A special case of linear queries are *counting queries*. A counting query associated with a predicate from U to $\{0,1\}$, outputs what fraction of the items in its input database satisfy the predicate. We view a counting query f as a vector over $\{0,1\}^N$ specifying which data items satisfy the query's predicate.

Accuracy and privacy in the interactive setting: Formally, an *interactive mechanism* $M(x)$ is a stateful randomized algorithm which holds a histogram $x \in \mathbb{R}^N$. It receives successive linear queries $f_1, f_2, \ldots \in \mathcal{F}$ one by one, and in each round t, on query f_t, it outputs a (randomized) answer a_t (a function of the input histogram, the internal state, and the mechanism's coins). For privacy guarantees, we *always* assume that the queries are given to the mechanism in an adversarial and adaptive fashion by a randomized algorithm A called the *adversary*. For accuracy guarantees, while we usually consider adaptive adversarial, we will also consider non-adaptive adversarial queries chosen in advance—we still consider such a mechanism to be interactive, because it does not know in advance what these queries will be. The main query class we consider throughout this work is the class $\mathcal{F}$ of all linear queries, as well as sub-classes of it.

Definition III.1. We say that a mechanism M is (α,β,k)-*(adaptively) accurate* for a database x, if when it is run for k rounds, for any (adaptively chosen) linear queries, with all but β probability over the mechanism's coins $\forall t \in [k], |a_t - \langle f_t, x\rangle| \le \alpha$.

We say that a mechanism M is (α,β,k)-*non-adaptively accurate* for a query sequence $\mathcal{C}$ of size k and a database x, if when it is run for k rounds on the queries in $\mathcal{C}$, with all but β probability over the mechanism's coins $\forall t \in [k], |a_t - \langle f_t, x\rangle| \le \alpha$.

For privacy, the interaction of a mechanism $M(x)$ and an adversary A specifies a probability distribution $[M(x), A]$ over *transcripts*, i.e., sequences of queries and answers $f_1, a_1, f_2, a_2, \ldots, f_k, a_k$. Let $\mathrm{Trans}(\mathcal{F}, k)$ denote the set of all transcripts of any length k with queries from $\mathcal{F}$. We will assume that the parameter k is known to the mechanism ahead of time. Our privacy requirement asks that the entire transcript satisfies differential privacy.

Definition III.2. We say a mechanism M provides (ε,δ)-*differential privacy* for a class of queries $\mathcal{F}$, if for every adversary A and every two histograms $x, x' \in \mathbb{R}^N$ satisfying $\|x - x\|_1 \le 1/n$, the following is true: Let $P = [M(x), A]$ denote the transcript between $M(x)$ and A. Let $Q = [M(x'), A]$ denote the transcript between $M(x')$ and A. Then, for every $S \subseteq \mathrm{Trans}(\mathcal{F}, k)$, we have $P(S) \le e^{\varepsilon} Q(S) + \delta$.

We will find it useful to work with the following condition, which (by Lemma III.1 below) is no weaker than (ε,δ) privacy:

$$\mathbb{P}\mathrm{r}_{u\sim P}\left\{\left|\log\left(\frac{P(u)}{Q(u)}\right)\right| > \varepsilon\right\} \le \delta. \qquad (1)$$

(Note that here we are identifying the distribution P with its density function dP.)

Lemma III.1. *(1) implies (ε,δ)-differential privacy.*

IV. Private multiplicative weights mechanism

Parameters: A subset of the coordinates $V \subseteq U$ with $|V| = M$ (by default $V = U$), intended number of rounds $k \in \mathbb{N}$, privacy parameters $\varepsilon, \delta > 0$ and failure probability $\beta > 0$. Put $\sigma = \frac{10\cdot\log(1/\delta)\log^{1/4} M}{\sqrt{n}\cdot\varepsilon}$, $\eta = \frac{\log^{1/4} M}{\sqrt{n}}$, $T = 4\sigma\cdot(\log k + \log(1/\beta))$
Input: Database $D \in U^n$ corresponding to a histogram $x \in \mathbb{R}^N$
Algorithm: Set $y_0[i] = x_0[i] = 1/M$ for all $i \in V$
In each round $t \leftarrow 1, 2\ldots, k$ when receiving a linear query f_t do the following:

1) Sample $A_t \sim \mathit{Lap}(\sigma)$. Compute the noisy answer $\widehat{a}_t \leftarrow \langle f_t, x\rangle + A_t$.
2) Compute the difference $\widehat{d}_t \leftarrow \widehat{a}_t - \langle f_t, x_{t-1}\rangle$:
If $|\widehat{d}_t| \le T$, then set $w_t \leftarrow 0$, $x_t \leftarrow x_{t-1}$, output $\langle f_t, x_{t-1}\rangle$, and proceed to the next iteration.
If $|\widehat{d}_t| > T$, then set $w_t \leftarrow 1$ and (a) for all $i \in V$, update $y_t[i] \leftarrow x_{t-1}[i]\cdot\exp(-\eta\cdot r_t[i])$, where $r_t[i] = f_t[i]$ if $\widehat{d}_t > 0$ and $r_t[i] = 1 - f_t[i]$ otherwise. (b) Normalize, $x_t[i] \leftarrow \frac{y_t[i]}{\sum_{i\in V} y_t[i]}$. (c) Let $m = \sum_{j=1}^{t} w_j$. If $m > n\cdot\log^{1/2} M$, then abort and output "failure". Otherwise, output the noisy answer $\widehat{a}_t$ and proceed to the next iteration.

Figure 1. **Private Multiplicative Weights (PMW) Mechanism**

In the PMW mechanism of Figure 1, in each round t, we are given a linear query f_t over U and x_t denotes a fractional histogram (distribution over $V \subseteq U$) computed in round t. The domain of this histogram is V rather than U. Here, V could be much smaller than U and this allows for some flexibility later, in proving Theorem I.2, where we aim for improved efficiency. For this section, unless otherwise specified, we assume that $V = U$. In particular this is the case in the statement of Theorem I.1, the main theorem that we prove in this section.

We use a_t to denote the true answer on the database on query t, and $\widehat{a}_t$ denotes this same answer with noise added to it. We use d_t to denote the difference between the true answer a_t and the answer given by x_{t-1}, and $\widehat{d}_t$ to denote the difference between the *noisy* answer and the answer given by x_{t-1}. The boolean variable w_t denotes whether the noisy difference was large or small. If $\widehat{d}_t$ is smaller (in absolute value) than $\approx 1/\sqrt{n}$, then this round is *lazy* and we set $w_t = 0$. If $\widehat{d}_t$ is larger than threshold then this is an *update* round and we set $w_t = 1$.

We are now ready to prove Theorem I.1, i.e. the utility and privacy of the PMW mechanism. This follows directly from the next two lemmas that are proved in Section IV-A and Section IV-B, respectively.

A. Utility analysis

To argue utility, we need to show that even for very large total number of rounds k, the number of update rounds is at most roughly n with high probability. This is done using a potential argument. Intuitively, the potential of a database x_t is the relative entropy between the true histogram x and our estimate x_t.

Since in general $V \neq U$, we will actually define the potential with respect to a target histogram $x^* \in \mathbb{R}^N$ with support only over V. This x^* need not be equal to x, nor does it have to be known by the algorithm. This added bit of generality will be useful for us later in Section V when we modify the mechanism to run in sublinear time. For this section, however, unless we explicitly note otherwise the reader may think of x^* as being equal to x. The potential function is then defined as

$$\Phi_t = \mathrm{RE}(x^* || x_t) = \textstyle\sum_{i \in V} x^*[i] \log\left(\frac{x^*[i]}{x_t[i]}\right) . \qquad (2)$$

Note that x^* and x_t are both normalized so that we can think of them both as distributions or histograms over U. We start with two simple observations: First, note that $\Phi_0 \leq \log M$. Indeed, by the nonnegativity of entropy $\mathrm{H}(x^*)$ we get that $\Phi_0 = \log M - \mathrm{H}(x^*) \leq \log M$. Second, by the nonnegativity of relative entropy, we have $\Phi_t \geq 0$ for every t. Our goal is to show that if a round is an update round (and $w_t = 1$), then the potential drop in that round is at least $\log^{1/2} M/n$. In Lemma IV.3 we show that this is indeed the case in every round, except with β/k probability over the algorithm's coins. Taking a union bound, we conclude that with all but β probability over the algorithm's coins, there are at most $n \cdot \log^{1/2} M$ update rounds. The next lemma quantifies the potential drop in terms of the penalty vector r_t and the parameter η using a multiplicative weights argument.

Lemma IV.1. *In each update round t, we have $\Phi_{t-1} - \Phi_t \geq \eta\langle r_t, x_{t-1} - x^*\rangle - \eta^2$.*

Proof: A direct calculation shows that

$$\Phi_{t-1} - \Phi_t = -\eta\langle r_t, x^*\rangle - \log\left(\textstyle\sum_{i\in V} \exp(-\eta r_t[i]) x_{t-1}[i]\right)$$

Note that $\exp(-\eta r_t[i]) \leq 1 - \eta r_t[i] + \eta^2 r_t[i]^2 \leq 1 - \eta r_t[i] + \eta^2$. Using this and $\sum x_{t-1}[i] = 1$ we get

$$\log\left(\sum_{i\in V} \exp(-\eta r_t[i]) x_{t-1}[i]\right) \leq \log\left(1 - \eta\langle r_t x_{t-1}\rangle + \eta^2\right)$$
$$\leq -\eta\langle r_t, x_{t-1}\rangle + \eta^2 ,$$

where we used $\log(1+y) \leq y$ for $y > -1$. We conclude that $\Phi_{t-1} - \Phi_t \geq -\eta\langle r_t, x^*\rangle + \eta\langle r_t, x_{t-1}\rangle - \eta^2 = \eta\langle r_t, x_{t-1} - x^*\rangle - \eta^2$. ■

In the following lemmata, we condition on the event that $|A_t| \leq T/2$. Since A_t is a centered Laplacian with standard deviation σ and $T \geq 2\sigma(\log k + \log(1/\beta))$, this event occurs with all but β/k probability in every round t.

The next lemma connects the inner product $\langle r_t, x^* - x_{t-1}\rangle$ with the "error" of x_{t-1} on the query f_t. Here, error is measured with respect to the true histogram x. To relate x with x^*, we further denote $\mathrm{err}(x^*, f_t) = |\langle f_t, x^*\rangle - \langle f_t, x\rangle|$. When $x^* = x$ we get that $\mathrm{err}(x^*, f_t) = 0$ always, and in general we will be interested in x^* databases where $\mathrm{err}(x^*, f_t)$ is small for all $t \in [k]$.

Lemma IV.2. *In each round t where $|\widehat{d}_t| \geq T$ and $|A_t| \leq T/2$ we have $\langle r_t, x^* - x_{t-1}\rangle \geq |\langle f_t, x\rangle - \langle f_t, x_{t-1}\rangle| - \mathrm{err}(x^*, f_t)$.*

Proof: By assumption $|\widehat{d}_t| \geq T$ and $|d_t - \widehat{d}_t| \leq |A_t| \leq T/2$. Hence, $sign(d_t) = sign(\widehat{d}_t)$. Now suppose $sign(d_t) < 0$. In other words, $\langle f_t, x\rangle - \langle f_t, x_{t-1}\rangle < 0$ and therefore $r_t[i] = 1 - f_t[i]$. Hence,

$$\sum_{i\in V} r_t[i](x^*[i] - x_{t-1}[i]) = -\Big(\langle f_t, x^*\rangle - \langle f_t, x_{t-1}\rangle\Big)$$
$$\geq -\Big(\langle f_t, x\rangle - \langle f_t, x_{t-1}\rangle\Big) - \mathrm{err}(x^*, f_t)$$
$$= |\langle f_t, x\rangle - \langle f_t, x_{t-1}\rangle| - \mathrm{err}(x^*, f_t) .$$

The case where $sign(d_t) = sign(\widehat{d}_t) \geq 0$ is analogous. The claim follows. ■

Combining the previous two lemmas, we get the following claim (whose simple proof is omitted).

Lemma IV.3. *In each round t where $|\widehat{d}_t| \geq T$ and $A_t \leq T/2$ we have $\Phi_{t-1} - \Phi_t \geq \eta\left(\frac{T}{2} - \mathrm{err}(x^*, f_t)\right) - \eta^2$.*

We are now ready to prove our main lemma about utility.

Lemma IV.4 (Utility for $V = U$)**.** *When the PMW mechanism is run with $V = U$, it is an (α, β, k)-accurate interactive mechanism, where $\alpha = O\left(\varepsilon^{-1} n^{-1/2} \cdot \log(1/\delta) \log^{1/4} N \cdot (\log k + \log(1/\beta))\right)$*

Proof: For $V = U$, we may choose $x^* = x$ so that $\mathrm{err}(f_t) = 0$ for all $t \in [k]$. Furthermore, with all but β probability over the algorithm's coins, the event $A_t \leq T/2$ occurs for every round $t \in [k]$. Hence, by Lemma IV.3 and $T \geq 4\eta$, the potential drop in every update round is at least $\Phi_{t-1} - \Phi_t \geq \eta\frac{T}{2} - \eta^2 \geq \eta^2$.

Since $\eta = \frac{\log^{1/4} M}{\sqrt{n}}$, it follows that there are at most $n\sqrt{\log N}$ update rounds. In particular, the algorithm terminates. Furthermore, the error of the algorithm is never larger than $T + |A_t| \leq 2T$ which is what we claimed. ∎

We now give a utility analysis in the general case where we are working with a smaller universe $V \subseteq U$. This will be used (in Section V) to prove the utility guarantee of Theorem I.2. The proof is analogous that the previous one except for minor modifications.

Lemma IV.5. *Let $f_1, f_2, \ldots, f_k$ denote a sequence of k linear queries. Take $\gamma = \inf_{x^*} \sup_{t\in[k]} \mathrm{err}(x^*, f_t)$ where x^* ranges over all histograms supported on V. When the PMW mechanism is run with V on the query sequence above, and with threshold parameter $T' = T + \gamma$, it is an (α, β, k)-non-adaptively accurate interactive mechanism, where $\alpha = O\left(\gamma + \varepsilon^{-1} n^{-1/2} \log(1/\delta) \log^{1/4} M \cdot (\log k + \log(1/\beta))\right)$.*

B. Privacy analysis

Our goal in this section is to demonstrate that the interactive mechanism satisfies (ε, δ)-differential privacy (see Definition III.2). We assume that all parameters such as V, σ, η, and T are publicly known. They pose no privacy threat as they do not depend on the input database. For ease of notation we will assume that $V = U$ throughout this section. The proof is the same for $V \subseteq U$. (the sub-universe V is always public information).

Simplifying the transcript: Without loss of generality, we can simplify the output of our mechanism (and hence the transcript between adversary and mechanism). We claim that the output of the mechanism is determined by the following vector v. In particular, it is sufficient to argue that v is differentially private. For every round t, the t-th entry in v is defined as $v_t = \widehat{a}_t$ in case $w_t = 1$ and otherwise $v_t = \perp$. In other words, v_t is equal to $\perp$ if that round was a lazy round, or the noisy answer $\widehat{a}_t = \langle f_t, x\rangle + A_t$ if round t was an update round. This is sufficient information for reconstructing the algorithm's output: given the prefix $v_{<t} = (v_1, \ldots, v_{t-1})$, we can compute the current histogram x_{t-1} for the beginning of round t. For the lazy rounds, this is sufficient information for generating the algorithm's output. For the update rounds, $v_t = \widehat{a}_t$, which is the output for round t. It is also sufficient information for re-weighting and computing the new x_t.

Note that to argue differential privacy, we need to prove that the entire transcript, including the queries of the adversary, is differentially private. Without loss of generality, we may assume that the adversary is deterministic.[3] In this case f_t is determined by $v_{<t}$. Hence, there is no need to include f_t explicitly in our transcript. It suffices to show that the vector v is (ε, δ)-differentially private.

[3]We can think of a randomized adversary as a collection of deterministic adversaries one for each fixing of the adversary's randomness (which is independent of our algorithm's coin tosses).

Lemma IV.6 (Privacy). *The PMW mechanism satisfies (ε, δ)-differential privacy.*

Proof: Fix an adversary and histograms $x, x' \in \mathbb{R}^N$ so that $\|x - x'\|_1 \leq 1/n$. Take $m = n \cdot \log^{1/2} M$ and let $\varepsilon_0 = 1/\sigma m$ (where σ is the scaling parameter in our algorithm).

Let P denote the output distribution of our mechanism when run on the input database x and similarly let Q denote the output of our mechanism when run on x'. Both distributions are supported on $\mathcal{S} = (\{\perp\} \cup \mathbb{R})^k$. For $v \in \mathcal{S}$, we define the *loss* function $L(v) := \log\left(\frac{P(v)}{Q(v)}\right)$. We will then show that $\Pr_{v\sim P}\{L(v) \leq \varepsilon\} \geq 1 - \delta$. By Lemma III.1, this implies (ε, δ)-differential privacy and hence our claim.

Using the chain rule for conditional probabilities, let us rewrite $L(v)$ as

$$L(v) = \log\left(\frac{P(v)}{Q(v)}\right) = \sum_{t\in[k]} \log\left(\frac{P(v_t \mid v_{<t})}{Q(v_t \mid v_{<t})}\right), \qquad (3)$$

where $P(v_t \mid v_{<t})$ denotes the probability of outputting v_t on input histogram x, conditioned on $v_{<t} = (v_1, \ldots, v_{t-1})$, similarly defined for Q (on histogram x'). Note that conditioning on $v_{<t}$ is necessary, since the coordinates of v are not independent. Further, note that conditioned on $v_{<t}$, the estimate x_{t-1} is the same regardless of whether we started from x or x'.

Borderline event: Fix $v_{<t}$. We define an event $S_t = S(v_{<t}) \subseteq \mathbb{R}$ on the noise values as follows. Let $d_t = \langle f_t, x\rangle - \langle f_t, x_{t-1}\rangle$. Note that x_{t-1} depends on $v_{<t}$ and therefore S_t will depend on it as well.

We define S_t so it contains all of the noise values A_t where $|\widehat{d}_t| = |d_t + A_t|$ is within distance σ or more from the threshold T. Formally, we construct $S_t = S^+ \cup S^-$ to be made up of two intervals of noise values: one interval $S^+ = [T - d_t - \sigma, \infty]$, around $T - d_t$, and the second interval $S^- = [-\infty, -T - d_t + \sigma]$, around $-T - d_t$. Note that, since $T > 2\sigma$, these two intervals never intersect.

The next three claims collect the key properties of a borderline event. Claims IV.7 and IV.9 follow from basic properties of the Laplacian distribution. The formal proofs are omitted from this extended abstract.

Claim IV.7. $\Pr(|\widehat{d}_t| \geq T \mid A_t \in S_t, v_{<t}) \geq 1/6$.

Claim IV.8. *For every $a \in \mathbb{R} \cup \{\perp\}$:*

$$\log\left(\frac{P(v_t = a \mid A_t \notin S_t, v_{<t})}{Q(v_t = a \mid A_t \notin S_t, v_{<t})}\right) = 0.$$

Claim IV.9. *For every $a \in \mathbb{R} \cup \{\perp\}$:*

$$\log\left(\frac{P(v_t = a \mid A_t \in S_t, v_{<t})}{Q(v_t = a \mid A_t \in S_t, v_{<t})}\right) \leq 2\varepsilon_0.$$

Bounding the Expectation: It was shown in [10] that Claim IV.9 implies

$$\mathbb{E}\left[\log\left(\frac{P(v_t \mid A_t \in S_t, v_{<t})}{Q(v_t \mid A_t \in S_t, v_{<t})}\right)\right] \leq 8\varepsilon_0^2. \qquad (4)$$

Here the expectation is taken over v_t sampled according to the conditional distribution $P(v_t \mid A_t \in S_t, v_{<t})$. Directly by Claim IV.8 we have:

$$\log\left(\frac{P(v_t \mid A_t \notin S_t, v_{<t})}{Q(v_t \mid A_t \notin S_t, v_{<t})}\right) = 0\,, \tag{5}$$

We can express $P(v_t \mid v_{<t})$ as a convex combination in the form $P(v_t \mid v_{<t}) = \Pr(A_t \in S_t \mid v_{<t})P(v_t \mid A_t \in S_t, v_{<t}) + \Pr(A_t \notin S_t \mid v_{<t})P(v_t \mid A_t \notin S_t, v_{<t})$, and we can express $Q(v_t \mid v_{<t})$ similarly. These observations (with a convexity argument) imply that

$$\mathop{\mathbb{E}}_{v_t}\left[\log\left(\frac{P(v_t \mid v_{<t})}{Q(v_t \mid v_{<t})}\right)\right] \le 8\varepsilon_0^2 \Pr(A_t \in S_t \mid v_{<t})\,. \tag{6}$$

On the other hand, we have $\sum_{i=1}^{k} \Pr(A_t \in S_t \mid v_{<t}) \le 6m$ This is because each round where $A_t \in S_t$ is a update round with probability at least $1/6$ (by Claim IV.7.) Hence,

$$\mathbb{E}\, L(v) \le 48\varepsilon_0^2 m \le \varepsilon/2\,. \tag{7}$$

Number of Borderline Rounds: With overwhelming probability, the number m' of borderline rounds (rounds t where S_t occurs) is not much larger than m (the bound on the number of update rounds). This is because every borderline round is with probability at least $1/6$ a update round (Claim IV.7. This is made formal in the claim below.

Claim IV.10. $\Pr(m' > 32m\log^{1/2}(1/\delta)) \le \delta/2$

Proof: Recall conditioned on $A_t \in S_t$, we have that $A_t \in R_t$ with probability $1/6$. The latter can only happen m times. Moreover, the noise in each round is independent from previous rounds. Hence, by tail bounds for Bernoulli variables, the event $m' > 32\sqrt{\log(1/\delta)}m$ has probability less than $\exp(-\log(2/\delta))$. ■

Putting it Together: Condition on there being at most $m' = 32m\log^{1/2}(1/\delta)$ borderline rounds (this is the case with all but $\delta/2$ probability). We proceed by an "evolution of confidence argument" similar to [3], [9].

Specifically, we will apply Azuma's inequality to the set of m' borderline rounds. Formally, let $B \subseteq [k]$ denote the set of borderline rounds. For each $t \in B$, we view $X_t = \log\left(\frac{P(v_t|v_{<t})}{Q(v_t|v_{<t})}\right)$ as a random variable. Note that $L(v) = \sum_{t\in B} X_t$. Further $|X_t| \le 2\varepsilon_0$ by Claim IV.9. Hence, by Azuma's inequality, $\Pr\{L(v) > \varepsilon\} \le \Pr\{L(v) > \mathbb{E}[L(v)] + \varepsilon/2\} \le 2\exp\left(-\frac{\varepsilon^2}{8m'\cdot\varepsilon_0^2}\right)$, where $\frac{\varepsilon^2}{8m'\cdot\varepsilon_0^2} \ge \frac{\varepsilon^2\sigma^2 n^2}{m'} \ge \frac{100\log(1/\delta)^2 n \log^{1/2} N}{m'} = 100\log(1/\delta)\frac{m}{m'}$.

So, conditioning on having at most m' borderline rounds (occurs with all but $\delta/2$ probability), with all but $\delta/2$ probability the loss $L(v)$ deviates by at most $\varepsilon/2$ from its expectation. The expectation itself is at most $\varepsilon/2$ by (7). We conclude that with all but δ probability, the total loss $L(v)$ is bounded by ε. ■

V. Average-case complexity and smooth instances

In this section, we define a notion of average case complexity for interactive (and non-interactive) mechanisms that allows us to improve the running time of the PMW mechanism as a function of the data universe size. This is done using an argument for reducing the data universe size.

We start by defining the notion of a *smooth* histogram. We think of these histograms as distributions over the data universe that do not place too much weight on any given data item. In other words, we require the histogram to have high min-entropy.

Definition V.1 (Smooth)**.** A histogram $x \in \mathbb{R}^U$ s.t. $\sum_{u\in U} x_u = 1$ and $\forall u \in U : x_u \ge 0$ is ξ-smooth if $\forall u \in U : x_u \le \xi$.

In particular, a ξ-smooth histogram has *min-entropy* at least $\log(1/\xi)$. We typically think of ξ has a function of N, such as $polylogN/N$ or $1/\sqrt{N}$. Note that small databases (viewed as histograms) cannot be very smooth, since a ξ-smooth histogram has at least $1/\xi$ nonzero coordinates.

We therefore extend the notion of smoothness to the notion of pseudo-smoothness with respect to a set of queries $\mathcal{C}$. A histogram is *pseudo-smooth* w.r.t a query class $\mathcal{C}$ roughly speaking when there exists a smooth histogram x^* that is close on every query in $\mathcal{C}$. This notion allows even very sparse histograms (corresponding to small databases) to be very pseudo-smooth. The formal definition is as follows.

Definition V.2 (Pseudo-smooth)**.** A histogram $x \in \mathbb{R}^U$ s.t. $\sum_{u\in U} x_u = 1$ and $\forall u \in U : x_u \ge 0$ is (ξ, ϕ)-smooth w.r.t a class of linear queries $\mathcal{C}$ if there *exists* a ξ-smooth histogram x^* s.t. $\forall f \in \mathcal{C}$: $|\langle f, x\rangle - \langle f, x^*\rangle| \le \phi$.

A straightforward way of obtaining pseudo-smooth databases is by sampling from a smooth histogram.

Claim V.1. *Let U be a data universe, $\mathcal{C}$ a class of linear queries over U, and x^* a ξ-smooth histogram over U. For any $\alpha, \beta > 0$, sample a database x of $m = (\log(2/\beta) + \log|\mathcal{C}|)/\alpha^2$ items i.i.d from the distribution of x^* (i.e. in each sample we independently pick each $u \in U$ with probability x^*_u). Then with all but β probability over the samples taken, $\forall f \in \mathcal{C}$: $|\langle f, x\rangle - \langle f, x^*\rangle| \le \alpha$, and so the database x is (ξ, α)-pseudosmooth w.r.t $\mathcal{C}$.*

A. Domain reduction for pseudosmooth histograms

For a given smoothness parameter ξ, data universe U, and query class $\mathcal{C}$, let $V \subseteq U$ be a sub-universe sampled uniformly and at random from U. In this section we show that (as long as V is large enough) if x was a pseudosmooth histogram over U w.r.t a query class $\mathcal{C}$, then w.h.p. there will be a histogram x^* *with support only over (the smaller) V* that is "close" to x on $\mathcal{C}$. We emphasize that sampling the sub-universe V does not require knowing x nor knowing any x^* that certifies x being pseudosmooth, we only need to know ξ. In particular, this approach is privacy-preserving. This technique for reducing the universe size can be used to improve the efficiency of the PMW mechanism for pseudosmooth input databases.

Lemma V.2. *Let U be a data universe and $\mathcal{C}$ a collection of linear queries over U. Let x be (ξ, ϕ)-psuedo-smooth w.r.t $\mathcal{C}$. Take $\alpha, \beta > 0$, and sample uniformly at random (with replacement) $V \subseteq U$ so that $M = |V| = 4\max\{\xi N \cdot (\log(1/\beta) + \log|\mathcal{C}|)/\alpha^2, \log(1/\beta)\}$ Then, with all but β probability over the choice of V, there exists a histogram x^* with support only over V such that*

$$\forall f \in \mathcal{C} : |f(x) - f(x^*)| \leq \phi + \alpha\,. \tag{8}$$

Proof: Let y be the ξ-smooth histogram which shows that x is (ξ, ϕ)-pseudosmooth. If we sampled uniformly at random from x or from y then by Claim V.1, we could get a database over a very small sub-universe that is (as required) close to x on all the queries in $\mathcal{C}$. This is insufficient because we want the sub-universe that we find to be *independent* of the database x (and so also independent of y).

Still, let us re-examine the idea of sampling from y. One way of doing this is by *rejection sampling*. Namely, repeatedly sample $u \in U$ *uniformly at random* and then "keep" u with probability y_u/ξ. Otherwise reject. When we use this rejection sampling, since y is a ξ-smooth distribution, each sample that we keep is distributed by y (i.e. it is $u \in U$ w.p. y_u). Repeat this process until $m_1 = (\log(2/\beta) + \log|\mathcal{C}|)/\alpha^2$ samples have been accepted. There is now a set of coordinates $V_1 \subseteq U$, those that were kept (of size at most m_1), and a set of coordinates $V_2 \subseteq U$, those that were rejected. By Claim V.1 the sub-universe V_1 of samples that we keep (which are i.i.d samples from y) supports (except with probability $\beta/2$) a database x^* that is "close" to y (w.r.t $\mathcal{C}$), and so it will also be "close" to x. In particular, by triangle inequality, $\max_{f\in\mathcal{C}} |f(x) - f(x^*)| \leq \max_{f\in\mathcal{C}} |f(x) - f(y)| + \max_{f\in\mathcal{C}} |f(y) - f(x^*)| \leq \phi + \alpha\,.$

But now we may take $V = V_1 \cup V_2$. Note that V is simply a uniformly random subset of the coordinates of U. And by the previous argument, V supports a histogram that satisfies (8), namely x^*. To conclude the proof it remains to argue that V has the required size. Note that the probability of accepting sample i in the rejection procedure is given by $\sum_{i=1}^{N} \frac{1}{N} \cdot \frac{y_i}{\xi} = 1/\xi N$. Hence, the expected number of queries in total is $\mu = 2\xi N \cdot (\log(2/\beta) + \log|\mathcal{C}|)/\alpha^2$. Moreover, since every sample is independent, we have concentration around the expectation. A multiplicative Chernoff bound shows that the probability that V is larger than twice its expectation is bounded by $\exp(-\mu) \leq \beta/2$. ■

Finally, we use Lemma V.2 together with Lemma IV.5 (utility of PMW for general V), to sample a small sub-universe V and derive the accuracy guarantee of Theorem I.2 for the performance of the PMW mechanism on pseudosmooth databases. The details are omitted from this extended abstract.

VI. Acknowledgements

We thank Boaz Barak, Cynthia Dwork, Moni Naor, Aaron Roth, Rob Schapire and Salil Vadhan for their helpful and insightful comments. Thanks also to the anonymous FOCS 2010 reviewers for their helpful comments.

References

[1] S. Arora, E. Hazan, and S. Kale. The multiplicative weights update method: a meta algorithm and applications. Technical report, Princeton University, 2005.

[2] A. Blum, K. Ligett, and A. Roth. A learning theory approach to non-interactive database privacy. In *STOC '08: Proceedings of the 40th annual ACM symposium on Theory of computing*, pages 609–618, New York, NY, USA, 2008. ACM.

[3] I. Dinur and K. Nissim. Revealing information while preserving privacy. In *Proc.* 22*nd PODS*, pages 202–210. ACM, 2003.

[4] C. Dwork and J. Lei. Differential privacy and robust statistics. In *Proc.* 41*st STOC*, pages 371–380. ACM, 2009.

[5] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *Proc.* 3*rd TCC*, pages 265–284. Springer, 2006.

[6] C. Dwork, F. McSherry, and K. Talwar. The price of privacy and the limits of LP decoding. In *Proc.* 39*th STOC*, pages 85–94. ACM, 2007.

[7] C. Dwork, M. Naor, T. Pitassi, and G. N. Rothblum. Differential privacy under continual observation. In *Proc.* 42*nd STOC*. ACM, 2010.

[8] C. Dwork, M. Naor, O. Reingold, G. N. Rothblum, and S. P. Vadhan. On the complexity of differentially private data release: efficient algorithms and hardness results. In *Proc.* 41*st STOC*, pages 381–390. ACM, 2009.

[9] C. Dwork and K. Nissim. Privacy-preserving datamining on vertically partitioned databases. In *Proc.* 24*th CRYPTO*, pages 528–544. Springer, 2004.

[10] C. Dwork, G. N. Rothblum, and S. Vadhan. Boosting and differential privacy. Manuscript, 2010.

[11] C. Dwork and S. Yekhanin. New efficient attacks on statistical disclosure control mechanisms. In *Proc.* 28*th CRYPTO*, pages 469–480. Springer, 2008.

[12] N. Littlestone and M. K. Warmuth. The weighted majority algorithm. *Inf. Comput.*, 108(2):212–261, 1994.

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

[14] J. Ullman and S. Vadhan. PCPs and the hardness of generating synthetic data. *Electronic Colloquium on Computational Complexity (ECCC)*, 1(17), 2010.