Local, Private, Efficient Protocols for Succinct Histograms

Raef Bassily Adam Smith*
Department of Computer Science and Engineering
The Pennsylvania State University
{bassily, asmith}@psu.edu

ABSTRACT

We give efficient protocols and matching accuracy lower bounds for frequency estimation in the local model for differential privacy. In this model, individual users randomize their data themselves, sending differentially private reports to an untrusted server that aggregates them.

We study protocols that produce a succinct histogram representation of the data. A succinct histogram is a list of the most frequent items in the data (often called "heavy hitters") along with estimates of their frequencies; the frequency of all other items is implicitly estimated as 0.

If there are n users whose items come from a universe of size d, our protocols run in time polynomial in n and $\log(d)$. With high probability, they estimate the accuracy of every item up to error $O(\sqrt{\log(d)/(\epsilon^2 n)})$. Moreover, we show that this much error is necessary, regardless of computational efficiency, and even for the simple setting where only one item appears with significant frequency in the data set.

Previous protocols (Mishra and Sandler, 2006; Hsu, Khanna and Roth, 2012) for this task either ran in time $\Omega(d)$ or had much worse error (about $\sqrt[6]{\log(d)/(\epsilon^2 n)}$), and the only known lower bound on error was $\Omega(1/\sqrt{n})$.

We also adapt a result of McGregor et al (2010) to the local setting. In a model with public coins, we show that each user need only send 1 bit to the server. For all known local protocols (including ours), the transformation preserves computational efficiency.

1. INTRODUCTION

Consider a software producer that wishes to gather statistics on how people use its software. If the software handles sensitive information —for example, a browser for anonymous web surfing or a financial management software—users may not want to share their data with the producer. A producer may not want to collect the raw data either, lest they be subject to subpoena. How can the producer collect high-quality *aggregate* information about users while providing guarantees to its users (and itself!) that it isn't storing user-specific information?

In the *local model* for private data analysis (also called the *randomized response* model[1]), each individual user randomizes her data herself using a randomizer Q_i to obtain a report (or "signal") z_i which she sends to an *untrusted* server to be aggregated in to a summary s that can be used to answer queries about the data (Figure 1). The server may provide public coins visible to all parties, but privacy guarantees depend only on the randomness of the user's local coins. The local model has been studied extensively because control of private data remains in users' hands.

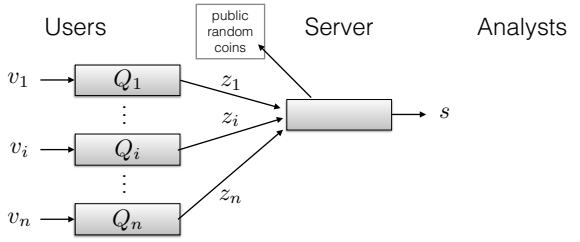

Figure 1: The local model for private data analysis.

We focus on protocols that provide *differential privacy* [6] (or, equivalently in the local model, γ-*amplification* [10] or *FRAPP* [1]).

DEFINITION 1.1. *We say that an algorithm $\mathcal{Q} : \mathcal{V} \to \mathcal{Z}$ is (ϵ,δ)-local differentially private (or (ϵ,δ)-***LDP***), if for any pair $v, v' \in \mathcal{V}$ and any (measurable) subset $\mathcal{S} \subseteq \mathcal{Z}$, we have*

$$\Pr\left[\mathcal{Q}(v) \in \mathcal{S}\right] \leq e^{\epsilon} \Pr\left[\mathcal{Q}(v') \in \mathcal{S}\right] + \delta.$$

The special case with $\delta = 0$ is called pure ϵ-***LDP***.

We describe new protocols and lower bounds for frequency estimation and finding heavy hitters in the local privacy model. Local differentially private protocols for frequency estimation are used in the Chrome web browser (Erlingsson et al. [9], Fanti et al. [11]), and can be used as the basis of other estimation tasks (see Mishra and Sandler [21], Dwork and Nissim [5]).

We also show a generic result for LDP protocols: in the public-coin setting, each user only needs to send 1 bit to the server.

Suppose that there are n users, and each user i holds a value v_i in a universe of size d (labeled by integers in $[d] = \{1, ..., d\}$). We wish to enable an analyst to estimate frequencies: $f(v) = \frac{1}{n}\#\{i : v_i = v\}$. Following Hsu et al. [14], we look at summaries that provide two types of functionality:

*Work done while A.S. was on sabbatical at Boston University and Harvard University.

STOC'15, June 14–17, 2015, Portland, Oregon, USA.

ACM 978-1-4503-3536-2/15/06 ...$15.00.
DOI: http://dx.doi.org/10.1145/2746539.2746632.

[1]The term "randomized response" may refer either to the model or a specific protocol; we use "local model" to avoid ambiguity.

- A **frequency oracle**, denoted FO, is a data structure together with an algorithm A that, for any $v \in \mathcal{V}$, allows computing an estimate $\hat{f}(v) = A(\text{FO}, v)$ of the frequency $f(v)$.

  The *error* of the oracle FO is the maximum over items v of $|\hat{f}(v) - f(v)|$. That is, we measure the ℓ_∞ error of the histogram estimate implicitly defined by $\hat{f}$. A protocol for generating frequency oracles has error (η,β) if for all data sets, it produces an oracle with error η with probability at least $1 - \beta$.

- A **succinct histogram**, denoted S-Hist, is a data structure that provides a (short) list of items $\hat{v}_1, ..., \hat{v}_k$, called the *heavy hitters*, together with estimated frequencies $(\hat{f}(\hat{v}_j) : j \in [k])$. The frequencies of the items not in the list are implicitly estimated as $\hat{f}(v) = 0$. As with the frequency oracle, we measure the error of S-Hist by the ℓ_∞ distance between the estimated and true frequencies, $\max_{v\in[d]} |\hat{f}(v) - f(v)|$.

  If a data structure aims to provide error η, the list need never contain more than $O(1/\eta)$ items (since items with estimated frequencies below η may be omitted from the list, at the price of at most doubling the error).

If we ignore computation, these two functionalities are equivalent since a succinct histogram defines a frequency oracle directly and an analyst with a frequency oracle FO can query the oracle on all possible items and retain only those with estimated frequencies above a threshold η (increasing the error by at most η). However, when the universe size d is large (for example, if a user's input is their browser's home page or a financial summary), succinct histograms are much more useful.

We say a protocol is *efficient* if it has computation time, communication and storage polynomial in n and $\log(d)$ (the users' input length). Prior to this work, efficient protocols for both tasks satisfied only (ϵ,δ)-LDP for $\delta > 0$. Efficient protocols for frequency oracles [21, 14] were known with worst-case expected error $O(\sqrt{\frac{\log(d)\log(1/\delta)}{\epsilon^2 n}})$, while the only protocols for succinct histograms [14] had much worse error — about $\sqrt[6]{\frac{\log(d)\log(1/\delta)}{\epsilon^2 n}}$. Very recently, Fanti et al. [11] proposed a heuristic construction for which worst-case bounds are not known. None of these protocols matched the best lower bound on accuracy, $\Omega(1/\sqrt{n})$ [14].

1.1 Our Results

Efficient Local Protocols for Succinct Histograms with Optimal Error. We provide the first polynomial time *local* $(\epsilon, 0)$-differentially private protocol for succinct histograms that has worst-case error $O(\sqrt{\frac{\log(d)}{\epsilon^2 n}})$. As we show, this error is optimal for local protocols (regardless of computation time). Furthermore, in the public coin model, each participant sends only 1 bit to the server.

Previous constructions were either inefficient [21, 14] (taking time polynomial in d rather than $\log d$), or had much worse error guarantees[2]—at least $\Omega\big(\frac{\log(d)}{\epsilon^2 n}\big)^{1/6}$. Furthermore, constructions with communication sublinear in d satisfied only (ϵ,δ) privacy for $\delta > 0$.

Our construction consists of two main pieces. Our first protocol efficiently recovers a heavy hitter from the input, given a promise that the heavy hitter is *unique*: that is, all players either have a particular value v (initially unknown to the server) or a default value $\perp$. The idea is to have each player send a highly noisy version of an error-correcting encoding of their input; the server can then recover (the codeword for) v by averaging all the received reports and decoding the resulting vector.

Our full protocol, which works for all inputs, uses ideas from the literature on low-space algorithms and compressive sensing, e.g., [12]. Specifically, using random hashing, we can partition the universe of possible items into bins in which there is likely to be only a single heavy hitter. Running many copies of the protocol for unique heavy hitters in parallel, we can recover the list of heavy hitters. A careful analysis shows that the cost to privacy is essentially the same as running only a single copy of the underlying protocol.

Along the way, we provide simpler and more private frequency-oracle protocols. Specifically, we show that the "JL" protocol of Hsu et al. [14] can be made $(\epsilon, 0)$-differentially private, and can be simplified to use computations in much smaller dimension (roughly, $O(n)$ instead of $\Omega(n^4 \log d)$).

Lower Bounds on Error. We show that, regardless of computation time and communication, every local (ϵ,δ)-DP protocol for frequency estimation has worst-case error $\Omega(\sqrt{\frac{\log(d)}{\epsilon^2 n}})$ as long as $\delta \ll 1/n$. This shows that our efficient protocols have optimal error.

The instances that give rise to this lower bound are simple: one particular item v (unknown to the algorithm) appears with frequency η, while the remaining inputs are chosen uniformly at random from $[d] \setminus \{v\}$. The structure of these instances has several implications. First, our lower bounds apply equally well to worst-case error (over data sets), and "minimax error" (worst-case error over distributions in estimating the underlying distribution on data).

Second, the accuracy of frequency estimation protocols must depend on the universe size d in the local model, even if one item appears much more frequently than all others. In contrast, in a centralized model, there are (ϵ,δ)-differentially private protocols that achieve error independent of the universe size, assuming only that there is a small gap (about $\frac{\log(1/\delta)}{\epsilon n}$) between the frequencies of the heaviest and second-heaviest hitters.

The proof of our lower bounds adapts (and simplifies) a framework developed by Duchi et al. [4] for translating lower bounds on statistical estimation to the local privacy model. We make their framework more modular, and show that it can be used to prove lower bounds for (ϵ,δ)-differentially private protocols for $0 < \delta < 1/n$ (in its original instantiation, it applied only for $\delta = 0$). One lemma, possibly of independent interest, states that the mutual information between the input and output of a local protocol is at most $O(\epsilon^2 + \frac{\delta}{\epsilon}\log(d\epsilon/\delta))$. In particular, the relaxation with $\delta > 0$ does not allow one to circumvent information-theoretic lower bounds unless δ is very large.

1-bit Protocols Suffice for Local Privacy. We show that a slight modification to the compression technique of McGregor et al. [20, Theorem 14] yields the following: in a public coin model (where the server and players have access to a common random string), every $(\epsilon, 0)$-DP local protocol can be transformed so that each user sends only a single bit to the server. Moreover, the transformation is efficient under the assumption that one can efficiently compute conditional probabilities $Q(y|x)$ for the randomizers in the protocol. To our knowledge, all the local protocols in the literature (in particular, our efficient protocol for heavy hitters) satisfy this extra computability condition.

The randomness of the public coins affects utility but not privacy in the transformed protocol; in particular, the coins may be gener-

[2]Mishra and Sandler [21] state error bounds for a single query to the frequency oracle, assuming the query is determined before the protocol is executed. Known frequency oracle constructions (both previous work and ours) achieve error $O(\sqrt{\log(1/\beta)/n})$ in that error model.

ated by the untrusted server, by applying a pseudorandom function to the user's ID (if it is available), or by expanding a short seed sent by the user using a pseudorandom generator.

The transformation, following [20], is based on rejection sampling: the public coins are used to select a random sample from a fixed distribution, and a player uses his input to decide whether or not the sample should be kept (and used by the server) or ignored. This decision is transmitted as 1 bit to the server. Local privacy ensures that the rejection sampling procedure accepts with sufficiently large probability (and leaks little information about the input).

1.2 Other Related Work

In addition to the works mentioned so far on frequency estimation [21, 14, 9, 11], many papers have studied the complexity of local private protocols for specific tasks.

Most relevant here are the results of [15, 4] on learning and statistical estimation in the local model. Kasiviswanathan et al. [15] showed that when data are drawn i.i.d. from a distribution, then every LDP learning algorithm can be simulated in the statistical queries model [16]. In particular, they showed that learning parity and related functions requires an exponential amount of data. Their simulation technique is the inspiration for our communication reduction result.

Recently, Duchi et al. [4] studied a class of convex statistical estimation problems, giving tight (minimax-optimal) error guarantees. One of the local randomizers developed in [4] was the basis for the "basic randomizer" which is a building block for our protocols. Moreover, our lower bounds are based on the information-theoretic framework they establish.

Finally, our efficient protocols are based on ideas from the large literature on streaming algorithms and compressive sensing (as were the efficient protocols of Hsu et al. [14]). For example, the use of hashing to isolate unique "heavy" items appears in the context of sparse approximations to a vector's Fourier representation [12] (and arguably that idea has roots in learning algorithms for Fourier coefficients such as [18]). This provides further evidence of the close relationship between low-space algorithms and differential privacy (see, e.g., [8, 7, 3, 17, 22]).

2. BUILDING BLOCKS

2.1 The Basic Randomizer

We describe a basic randomizer (Algorithm 1) that will be used in our constructions as a tool to ensure that each user generates an ϵ-differentially private report. This randomizer is a more concise version of one of the randomizers in Duchi et al. [4].

Our basic randomizer $\mathcal{R}$ takes as input either an m-bit string represented by one of the vertices of the hypercube $\{-\frac{1}{\sqrt{m}}, \frac{1}{\sqrt{m}}\}^m$, or a special symbol represented by the all-zero m-length vector $\mathbf{0}$. The randomizer $\mathcal{R}$ picks a bit x_j at random from the input string $\mathbf{x}$ (where j is the index of the chosen bit), then it randomizes and scales x_j to generate a bit $z_j \in \{-c_\epsilon\sqrt{m}, c_\epsilon\sqrt{m}\}$ (for some fixed $c_\epsilon = O(1/\epsilon)$). Finally, $\mathcal{R}$ outputs the pair (j, z_j). As will become clear later in our constructions, the m-bit input of $\mathcal{R}$ will be a unique encoding of one of the items in $\mathcal{V}$ whereas the special symbol $\mathbf{0}$ will serve notational purposes to describe a special situation in our constructions when a user sends no information about its item.

Algorithm 1 $\mathcal{R}$: ϵ-Basic Randomizer

Input: m-bit string $\mathbf{x} \in \{-\frac{1}{\sqrt{m}}, \frac{1}{\sqrt{m}}\}^m \cup \{\mathbf{0}\}$, and the privacy parameter ϵ.

1: Sample $j \leftarrow [m]$ uniformly at random.
2: **if** $\mathbf{x} \neq \mathbf{0}$ **then**
3: Randomize j-th bit x_j of the input $\mathbf{x} \in \{-\frac{1}{\sqrt{m}}, \frac{1}{\sqrt{m}}\}^m$ as follows:

$$z_j = \begin{cases} c_\epsilon m x_j & \text{w.p. } \frac{e^\epsilon}{e^\epsilon+1} \\ -c_\epsilon m x_j & \text{w.p. } \frac{1}{e^\epsilon+1} \end{cases}$$

where $c_\epsilon = \frac{e^\epsilon+1}{e^\epsilon-1} = O\left(\frac{1}{\epsilon}\right)$.
4: **else**
5: Generate a uniform bit: $z_j \leftarrow \{-c_\epsilon\sqrt{m}, c_\epsilon\sqrt{m}\}$.
6: **return** $\mathbf{z} = (0, \ldots, 0, z_j, 0, \ldots, 0) \in \{-c_\epsilon\sqrt{m}, c_\epsilon\sqrt{m}\}^m$ where z_j is in the j-th position of $\mathbf{z}$. (This output can be represented concisely by the pair (j, z_j) using $\lceil \log m \rceil + 1$ bits).

THEOREM 2.1. *$\mathcal{R}$ has the following properties:*

1. *$\mathcal{R}$ is ϵ-**LDP**, for every choice of the index j (that is, privacy depends only on the randomness in Step 3).*
2. *For every $\mathbf{x} \in \{-\frac{1}{\sqrt{m}}, \frac{1}{\sqrt{m}}\}^m \cup \{\mathbf{0}\}$, $\mathcal{R}(\mathbf{x})$ is an unbiased estimator of $\mathbf{x}$. That is, $\mathbb{E}[\mathcal{R}(\mathbf{x})] = \mathbf{x}$.*
3. *$\mathcal{R}$ is computationally efficient (i.e., $\mathcal{R}$ runs in $O(m)$ time).*

As noted in Step 6 of the algorithm, we view the output of $\mathcal{R}$ as a vector $\mathbf{z} \in \mathbb{R}^m$ of the same length as the input vector $\mathbf{x}$. However, the output can be represented concisely by only $\lceil \log m \rceil + 1$ bits (required to describe the index j and z_j).

In some settings, we may compress this output to just 1 bit. This comes from the fact that the privacy of $\mathcal{R}$ holds no matter how the index j is chosen in Step 1, so long as it is independent of the input. (The randomness of j *is* important for utility since it helps ensure that $\mathbb{E}[\mathcal{R}(\mathbf{x})] = \mathbf{x}$.) In particular, the randomness in the choice of j may come from outside the randomizer: it could be sent by the server, available as public coins, or generated pseudorandomly from other information. In such situations, the server receives j through other channels and we may represent the output using the single bit describing z_j.

2.2 A Private Frequency Oracle

We give an efficient private frequency oracle that follows almost the same lines of the construction of [14]. Our protocol differs from [14] in three respects. First, we use the construction only to provide a frequency oracle as opposed to identifying and estimating the frequency of heavy hitters. For that purpose, the construction is computationally efficient. The second difference is in the local randomization step at each user. Here, each user $i \in [n]$ uses an independent copy of the basic randomizer $\mathcal{R}_i$ given by Algorithm 1 (as opposed to adding noise as in [14]). This gives us pure ϵ-differential privacy guarantee (as opposed to (ϵ,δ) in [14]). The third difference is that computations are carried out in much smaller dimension, namely $O(n)$ as opposed to $\Omega(n^4 \log(d))$ in [14].

The description of our frequency oracle construction protocol is given in the full version [2]. We refer to such protocol as PROT-FO. This protocol outputs a frequency oracle FO which is composed of two objects; a *succinct* description of a binary matrix Φ whose columns represent encodings of each item in $\mathcal{V}$, and, an aggregate measurement of users reports $\bar{\mathbf{z}}$. Given our private frequency oracle, there is a simple efficient algorithm $\mathcal{A}_{\mathsf{FO}}$ (see the

full version [2] for a complete description) that, for any given input $v \in \mathcal{V}$, uses the frequency oracle to obtain a private estimate $\hat{f}(v)$ of the frequency $f(v)$ of the item v by simply computing the inner product between the encoding of v under Φ and an aggregate measurement $\bar{\mathbf{z}}$, which is the average of users' reports.

In protocol PROT-FO, the length of encoding of an item under Φ is $O(n)$, the report length of each user is $O(\log(n))$, and the total amount of randomness required to generate Φ is $O(n^2)$ random bits. Also, each basic randomizer is efficient, i.e., runs in $O(n)$ steps (Part 3 of Theorem 2.1). Hence, the construction is computationally efficient. Also, generating an estimate $\hat{f}(v)$ for a given item v using $\mathcal{A}_{\mathsf{FO}}$ takes only $O(n)$ time.

The privacy and utility guarantees of the frequency oracle constructed by PROT-FO are given in the following theorems.

THEOREM 2.2. *The construction of the frequency oracle FO given by Protocol* PROT-*FO is ϵ-differentially private.*

THEOREM 2.3. *Let $\epsilon > 0$. For any set of users items $\{v_1, ..., v_n\}$ and any $\beta > 0$, the error due to FO constructed by Protocol* PROT-*FO is bounded as*

$$\text{ERR}\,(\mathbf{f}; FO) \triangleq \max_{v \in \mathcal{V}} |\hat{f}(v) - f(v)| = O\left(\frac{1}{\epsilon}\sqrt{\frac{\log(d/\beta)}{n}}\right)$$

with probability at least $1-\beta$ over the randomness of the projection Φ and the basic randomizers $\mathcal{R}_i, i \in [n]$, where $\hat{f}(v)$ denote the output of Procedure $\mathcal{A}_{\mathsf{FO}}$ on an input v.

The above upper bound is asymptotically tight (Section 5). The theorem's proof relies on the concentration of the inner product between the aggregate $\bar{\mathbf{z}}$ and the encoding of any given item under the encoding matrix Φ. See the full version [2] for details.

3. EFFICIENT CONSTRUCTION WITH OPTIMAL ERROR

In this section, our goal is to construct an efficient private succinct histogram using the private frequency oracle given in the previous subsection together with other tools. In Section 3.1, we first give a construction for a simpler problem that we call the unique heavy hitter problem. Then, in Section 3.2, we give a reduction from this problem to the general problem.

3.1 The Unique Heavy Hitter Problem

In the unique heavy hitter problem, we are given the promise that at least an η fraction of the n users hold the same item v^* for some $v^* \in \mathcal{V}$ unknown to the server (here η is a parameter of the promise), and that all other users hold a special symbol $\perp$, representing "no item".

Our goal is to obtain an efficient construction of a private succinct histogram under this promise, for as small a value η as possible. We will take η to be at least $\frac{C}{\epsilon}\sqrt{\frac{\log(d)}{n}}$ for a universal constant $C > 0$. Our protocol is differentially private on all inputs. Under the promise, with high probability, it outputs the correct v^* together with an estimate $\hat{f}(v^*)$ of the frequency $f(v^*)$.

The main idea of the protocol is to first encode user's items with an error-correcting code and randomize the resulting codeword before sending it to the server. The redundancy in the code allows the server learn the unknown item v^* from the noisy reports.

We require an efficiently encodable and decodable binary (d, m, ζ)-code (of d codewords, block length m, and relative distance ζ) where $m = O(\log(d))$ with constant rate (so that $m = O(\log(d))$) and constant relative minimum distance $\zeta \in (0, 1/2)$, say $\zeta = 1/4$. (We do not require the rate $\frac{\log d}{m}$ or relative distance ζ to be optimal; these quantities will affect the constants in the error of our construction but not the asymptotic behavior.) There are several known constructions of such codes in the literature (see [13] for examples). Fix one such code, denoted $\mathsf{code}(d, m, \zeta)$, with associated encoder $\mathbf{c}$ and decoder Dec. The code is part of the protocol and so is known to all parties. For convenience, we represent codewords as points in the unit-radius hypercube $\{-\frac{1}{\sqrt{m}}, \frac{1}{\sqrt{m}}\}^m$.

Each user i first encodes its item v_i to obtain $\mathbf{x}_i = \mathbf{c}(v_i) \in \{-\frac{1}{\sqrt{m}}, \frac{1}{\sqrt{m}}\}^m$, then runs the basic randomizer $\mathcal{R}_i$ (given by Algorithm 1) on $\mathbf{x}_i$ to obtain the report $\mathbf{z}_i$. Users that have no item, i.e., users with input $\perp$, feed the zero vector $\mathbf{x}_i = \mathbf{0}$ to the basic randomizer.

The server aggregates the reports by computing $\bar{\mathbf{z}} = \frac{1}{n}\sum_{i=1}^n \mathbf{z}_i$, and then decodes $\bar{\mathbf{z}}$ to obtain the encoding $\mathbf{x}$ of v^*. One may not be able to feed $\bar{\mathbf{z}}$ directly to the decoding algorithm Dec of $\mathsf{code}(d, m, \zeta)$ since $\bar{\mathbf{z}}$ will not, in general, be a vertex of the hypercube $\{-\frac{1}{\sqrt{m}}, \frac{1}{\sqrt{m}}\}^m$. Instead, the server first rounds the aggregated signal $\bar{\mathbf{z}}$ to the nearest point $\mathbf{y}$ in the hypercube before running Dec. We argue that the combination of noise from randomization and the rounding step produces a vector $\mathbf{y}$ that is sufficiently close to $\mathbf{x}$ with high probability.

Algorithm 2 precisely describes our construction for the promise problem. The protocol is computationally efficient, i.e., the total computational cost is $\mathsf{poly}(\log(d), n)$ since $\mathsf{code}(d, m, \zeta) = (\mathbf{c}, \mathsf{Dec})$ runs in time $\mathsf{poly}(\log(d))$ and each basic randomizer $\mathcal{R}_i$ runs in time $O(\log(d))$. In fact, the computational cost at each user does not depend on n. Also, we note that the users' reports are succinct, namely, the report length is $O\left(\log\left(\log(d)\right)\right)$ bits.

Algorithm 2 PROT$^{\mathsf{PP}}$-S-Hist$_{\mathsf{PP}}$: ϵ-**LDP** Succinct Histogram Protocol under the Promise

Input: Users' inputs $\{v_i \in \mathcal{V} \cup \{\perp\}: i \in [n]\}$, the privacy parameter ϵ, and the confidence parameter $\beta > 0$.

1: **for** Users $i = 1$ to n **do**
2: If $v_i \neq \perp$, then user i encodes its item: $\mathbf{x}_i = \mathbf{c}(v_i)$. Else, user i sets $\mathbf{x}_i = \mathbf{0}$.
3: User i computes its private report: $\mathbf{z}_i = \mathcal{R}_i(\mathbf{x}_i, \epsilon)$.
4: User i sends $\mathbf{z}_i$ to the server.
5: Server computes $\bar{\mathbf{z}} = \frac{1}{n}\sum_{i=1}^n \mathbf{z}_i$.
6: Server computes $\mathbf{y}$ by rounding $\bar{\mathbf{z}}$ to $\{-\frac{1}{\sqrt{m}}, \frac{1}{\sqrt{m}}\}^m$. That is, for each $j = 1, ..., m$,
$y_j = \begin{cases} \frac{1}{\sqrt{m}} & \text{if } \bar{z}_j \geq 0\,, \text{ and} \\ -\frac{1}{\sqrt{m}} & \text{otherwise,} \end{cases}$ where $\bar{z}_j$ denotes the j-th entry of $\bar{\mathbf{z}}$.
7: Server decodes $\mathbf{y}$ into an estimate for the common item $\hat{v} = \mathsf{Dec}(\mathbf{y})$
and computes a frequency estimate $\hat{f}(\hat{v}) = \langle \mathbf{c}(\hat{v}), \bar{\mathbf{z}} \rangle$.
8: **return** $\left(\hat{v},\ \hat{f}(\hat{v})\right)$.

THEOREM 3.1. *The construction of the succinct histogram S-Hist$_{\mathsf{PP}}$ given by Algorithm 2 is ϵ-differentially private.*

PROOF. Privacy follows directly from the ϵ-differential privacy of the basic randomizers $\mathcal{R}_i, i \in [n]$ (Part 1 of Theorem 2.1). □

To analyze utility, we first isolate the guarantee provided by the rounding step. Let $\mathbb{S}_m = \{\mathbf{w} \in \mathbb{R}^m : \|\mathbf{w}\|_2 = 1\}$ denote the m-dimensional unit sphere.

LEMMA 3.2. *Let $\mathbf{z} \in \mathbb{S}_m$ be such that there is a codeword of $\mathsf{code}(d, m, \zeta)$, $\mathbf{x} \in \mathcal{C}$, with $\langle \mathbf{z}, \mathbf{x} \rangle > 1 - \zeta/4$. Let $\mathbf{y}$ be vector in*

the hypercube $\{-\frac{1}{\sqrt{m}}, \frac{1}{\sqrt{m}}\}^m$ *obtained by rounding each entry* z_j *of* $\mathbf{z}$ *to* $sign(z_j)/\sqrt{m}$*. Then the Hamming distance between* $\mathbf{y}$ *and* $\mathbf{x}$ *is less than* $m\zeta/2$*, i.e.,* $\sum_{j=1}^{m} \mathbf{1}(y_j \neq x_j) < m\zeta/2$.

PROOF. Since $\mathbf{z}$ and $\mathbf{x}$ are unit vectors, the distance $\|\mathbf{z}-\mathbf{x}\|_2$ satisfies

$$\|\mathbf{z}-\mathbf{x}\|_2^2 = \|\mathbf{z}\|_2^2 + \|\mathbf{x}\|_2^2 - 2\langle \mathbf{z}, \mathbf{x}\rangle \leq \zeta/2.$$

The vectors $\mathbf{x}$ and $\mathbf{y}$ disagree in coordinate j only if $|z_j - x_j| \geq \frac{1}{\sqrt{m}}$. There can be at most $m\zeta/2$ such coordinates, since each contributes at least $\frac{1}{m}$ to $\|\mathbf{z}-\mathbf{x}\|_2^2$. Thus, the Hamming distance between $\mathbf{y}$ and $\mathbf{x}$ is $\sum_{j=1}^{m} \mathbf{1}(y_j \neq x_j) < m\zeta/2$ completing the proof. □

THEOREM 3.3. *Let* $\epsilon > 0$*. Suppose that the conditions in the above promise are true for some common item* $v^* \in \mathcal{V}$*. For any* $\beta > 0$*, there is a setting of* $\eta = O\left(\frac{1}{\epsilon}\sqrt{\frac{\log(d)\log(1/\beta)}{n}}\right)$ *in the promise such that, with probability at least* $1-\beta$*, Protocol* $\mathsf{PROT}^{\mathsf{PP}}$*-S-Hist*$_{\mathsf{PP}}$ *publishes the right item* v^* *and the frequency estimation error is bounded by*

$$\max_{v\in\mathcal{V}} |\hat{f}(v) - f(v)| = O\left(\frac{1}{\epsilon}\sqrt{\frac{\log(1/\beta)}{n}}\right).$$

PROOF. Consider the conditions of the promise. Let $v^* \in \mathcal{V}$ be the unique heavy hitter (occurring with frequency at least η). Let $\beta > 0$. Given Lemma 3.2, to show that the protocol above recovers the correct item v^* with probability at least $1-\beta/2$, it suffices to show that, with probability at least $1-\beta/2$, we have

$$\langle \mathbf{c}(v^*), \frac{\bar{\mathbf{z}}}{\|\bar{\mathbf{z}}\|_2}\rangle > 1 - \zeta/4.$$

Note that the rounding step (Step 6 in Algorithm 2) would produce the same output whether it was run with $\bar{\mathbf{z}}$ or its normalized counterpart $\bar{\mathbf{z}}/\|\bar{\mathbf{z}}\|_2$.

By the promise, we have

$$\bar{\mathbf{z}} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{z}_i = \frac{1}{n}\sum_{i\in\mathcal{T}} \mathcal{R}_i\left(\mathbf{c}(v^*)\right) + \frac{1}{n}\sum_{i\in[n]\setminus\mathcal{T}} \mathcal{R}_i\left(0\right)$$

where $\mathcal{T}$ denotes the set of users having the item v^*. (Note that $\frac{|\mathcal{T}|}{n} = f(v^*) \geq \eta$).

First, we consider $\|\bar{\mathbf{z}}\|_2$. Since for every $i \in [n]$, $\mathcal{R}_i$ is unbiased (Part 2 of Theorem 2.1), we have $\|\mathbb{E}[\bar{\mathbf{z}}]\|_2 = f(v^*)$. Using the triangle inequality, we get $\|\bar{\mathbf{z}}\|_2 \leq f(v^*) + \|\bar{\mathbf{z}} - \mathbb{E}\left[\bar{\mathbf{z}}\right]\|_2$. Next, we obtain an upper bound on $\|\bar{\mathbf{z}} - \mathbb{E}\left[\bar{\mathbf{z}}\right]\|_2$. Note that $\mathbf{z}_i, i = 1, .., n$, are independent and that for every $i \in [n]$, $\|\mathbf{z}_i\|_2 = O\left(\frac{\sqrt{m}}{\epsilon}\right)$ with probability 1. Applying McDiarmid's inequality [19], with probability at least $1-\beta/4$, we have $\|\bar{\mathbf{z}} - \mathbb{E}\left[\bar{\mathbf{z}}\right]\|_2 \leq O\left(\frac{1}{\epsilon}\sqrt{\frac{m\log(1/\beta)}{n}}\right)$. Thus, with probability at least $1-\beta/4$, $\|\bar{\mathbf{z}}\|_2$ is bounded by

$$\|\bar{\mathbf{z}}\|_2 \leq f(v^*) + O\left(\frac{1}{\epsilon}\sqrt{\frac{m\log(1/\beta)}{n}}\right) \tag{1}$$

Next, we consider $\langle \mathbf{c}(v^*), \bar{\mathbf{z}}\rangle$. Observe that $\langle \mathbf{c}(v^*), \bar{\mathbf{z}}\rangle =$

$$f(v^*) + \frac{1}{n}\sum_{i\in\mathcal{T}} \langle \mathbf{c}(v^*), \mathcal{R}_i(\mathbf{c}(v^*)) - \mathbf{c}(v^*)\rangle + \frac{1}{n}\sum_{i\in[n]\setminus\mathcal{T}} \langle \mathbf{c}(v^*), \mathcal{R}_i(\mathbf{0})\rangle$$

By the tail properties of the distribution of the second and third terms above[3], we can show that with probability at least $1-\beta/4$, we have

$$\langle \mathbf{c}(v^*), \bar{\mathbf{z}}\rangle \geq f(v^*) - O\left(\frac{1}{\epsilon}\sqrt{\frac{\log(1/\beta)}{n}}\right) \tag{2}$$

Putting (1) and (2) together, then, with probability at least $1-\beta/2$, we have

$$\langle \mathbf{c}(v^*), \frac{\bar{\mathbf{z}}}{\|\bar{\mathbf{z}}\|_2}\rangle \geq \frac{\eta - O\left(\frac{1}{\epsilon}\sqrt{\frac{\log(1/\beta)}{n}}\right)}{\eta + O\left(\frac{1}{\epsilon}\sqrt{\frac{m\log(1/\beta)}{n}}\right)}$$

where we use the fact that $\eta \leq f(v^*)$ and assume that the numerator in the right-hand side is positive.

Since $m = O(\log(d))$, then there is a constant α_ζ that depends on ζ such that if we set $\eta = \alpha_\zeta \frac{1}{\epsilon}\sqrt{\frac{\log(d)\log(1/\beta)}{n}}$, then the above ratio is greater than $1-\zeta/4$. This proves that there is a setting of $\eta = O\left(\frac{1}{\epsilon}\sqrt{\frac{\log(d)\log(1/\beta)}{n}}\right)$ such that construction $\mathsf{PROT}^{\mathsf{PP}}$-S-Hist$_{\mathsf{PP}}$ outputs $\hat{v} = v$ with probability at least $1-\beta/2$.

Now, conditioned on correct decoding, for all $v \neq v^*$, the estimate $\hat{f}(v)$ is implicitly assumed to be zero (which is perfectly accurate in this case). Thus, it remains to inspect $\hat{f}(v^*)$. Observe that

$$|\hat{f}(v^*) - f(v^*)| = \left|\langle \mathbf{c}(v^*), \frac{\bar{\mathbf{z}}}{\|\bar{\mathbf{z}}\|_2}\rangle - f(v^*)\right|$$

$$\leq \left|\frac{1}{n}\sum_{i\in\mathcal{T}} \langle \mathbf{c}(v^*), \mathcal{R}_i(\mathbf{c}(v^*)) - \mathbf{c}(v^*)\rangle\right| + \left|\frac{1}{n}\sum_{i\in[n]\setminus\mathcal{T}} \langle \mathbf{c}(v^*), \mathcal{R}_i(\mathbf{0})\rangle\right|$$

Again, by the tail properties of the sums above, with probability at least $1-\frac{\beta}{2}$, we conclude that $|\hat{f}(v^*) - f(v^*)| \leq O\left(\frac{1}{\epsilon}\sqrt{\frac{\log(2/\beta)}{n}}\right)$.

Therefore, with probability at least $1-\beta$, protocol $\mathsf{PROT}^{\mathsf{PP}}$-S-Hist$_{\mathsf{PP}}$ recovers the correct common item v^* and the estimation error that is bounded by $O\left(\frac{1}{\epsilon}\sqrt{\frac{\log(1/\beta)}{n}}\right)$. □

3.2 Efficient Construction for the General Problem

In this section, we provide an efficient construction of private succinct histograms with optimal error for the general setting of the problem using the two protocols discussed in the previous sections as sub-protocols.

In the promise (unique heavy hitter) problem, the main advantage was the lack of interference from the users who do not hold the heavy hitter v^* in question. The main idea here is to obtain a reduction in which we create the conditions of the promise problem separately for each heavy hitter $v^* \in \mathcal{V}$ such that the extra computational cost is at most a small $\mathsf{poly}(n)$ factor. To do this, we *hash* each item $v \in \mathcal{V}$ into one of K separate *parallel* channels such that users holding the same item will transmit their reports in the same channel. Each user, in the remaining $K-1$ channels, will simulate an "idle" user with item $\perp$ as in the promise problem. By choosing K sufficiently large, and repeating the protocol in *parallel* for T times[4], we can guarantee that, with high probability, *every*

[3]Such tail bounds follow from the properties of the basic randomizers $\mathcal{R}_i$ and a straightforward application of Hoeffding's inequality.

[4]That is, the total number of parallel channels is KT. In each group of K channels, a fresh hash seed is used.

heavy hitter $v^* \in \mathcal{V}$ gets assigned to an interference-free channel. Hence, by using an error-optimal construction for the promise problem like S-Hist$_{\mathsf{PP}}$ in each one of these channels, we eventually obtain a list of at most KT items such that, with high probability, all the heavy hitters will be on that list. However, this list may also contain other erroneously decoded items due to hash collisions and we do not know which items on the list are the heavy hitters. To overcome this, in a separate parallel channel of the protocol, we run a frequency oracle protocol (like PROT-FO) and use the resulting frequency oracle to estimate the frequencies of all the items on that list, then output all the items whose estimated frequencies are above η together with their estimated frequencies.

For the purpose of this construction, it suffices to use a pairwise independent hash function $\mathsf{Hash} : \{0,1\}^\ell \times \mathcal{V} \to [K]$ whose first input is a random seed $s \in \{0,1\}^\ell$. Choices of ℓ and K are given in the protocol description (Algorithm 3). All users and the server are assumed to have access to Hash. We use a source of *public* randomness RndGen that, on an input integer $\ell > 0$, generates a *seed* for Hash, which is a random uniform string from $\{0,1\}^\ell$ that is seen by everyone[5].

Algorithm 3 PROT-S-Hist: ϵ-**LDP** Efficient Protocol for Succinct Histograms

Input: Users' inputs $\{v_i \in \mathcal{V} \cup \{\perp\} : i \in [n]\}$, the privacy parameter ϵ, and the confidence parameter $\beta > 0$.

1: LIST $\leftarrow \emptyset$ (initialize list of heavy hitters to the empty set.)
2: $\ell \leftarrow 2\max(\log(d), \log(n))$; $K \leftarrow n^{3/2}$.
3: $T \leftarrow \lceil \log(3/\beta) \rceil$
4: **for** $t = 1$ to T **do**
5: $s_t \leftarrow \mathsf{RndGen}(\ell)$.
6: **for** Channels $k = 1$ to K **do**
7: **for** Users $i = 1$ to n **do**
8: If $\mathsf{Hash}(\mathsf{s_t}, \mathsf{v_i}) \neq \mathsf{k}$, set $v_i' \leftarrow \perp$. Else, set $v_i' \leftarrow v_i$
9: $\hat{v} \leftarrow \mathsf{PROT}^{\mathsf{PP}}\text{-S-Hist}_{\mathsf{PP}}\left(\{v_1', ..., v_n'\}; \frac{\epsilon}{2T+1}; \frac{\beta}{3d}\right)$ {i.e., run PROT$^{\mathsf{PP}}$-S-Hist$_{\mathsf{PP}}$ on the modified set of users' items to obtain an estimate $\hat{v}$ of the possibly unique item transmitted in the k-th channel.}
10: If $\hat{v} \notin$ LIST, then add $\hat{v}$ to LIST.
11: FO $\leftarrow$ PROT-FO$\left(\{v_1, ..., v_n\}; \frac{\epsilon}{2T+1}; \beta/3\right)$ {i.e., run PROT-FO on the original set of users' items to obtain the frequency oracle FO. }
12: **for** $\hat{v} \in$ LIST **do**
13: $\hat{f}(\hat{v}) \leftarrow \mathcal{A}_{\mathsf{FO}}(\mathrm{FO}, \hat{v})$. {$\mathcal{A}_{\mathsf{FO}}$ is the frequency estimator given in Section 2.2.}
14: If $\hat{f}(\hat{v}) < \frac{2T+1}{\epsilon}\sqrt{\frac{\log(d)\log(1/\beta)}{n}}$, remove $\hat{v}$ from LIST.
15: **return** $\left\{\left(v, \hat{f}(v)\right) : v \in \text{LIST}\right\}$.

It is not hard to see that the *total* computational cost of this construction is

$$O\left(n^{3/2}\log(1/\beta)\mathsf{cost}_{\mathsf{PP}} + \mathsf{cost}_{\mathsf{FO}} + n\mathsf{cost}_{\mathcal{A}_{\mathsf{FO}}}\right)$$

where $\mathsf{cost}_{\mathsf{PP}}$, $\mathsf{cost}_{\mathsf{FO}}$, and $\mathsf{cost}_{\mathcal{A}_{\mathsf{FO}}}$ are the computational costs of the promise problem sub-protocol, the frequency oracle sub-protocol, and the algorithm that computes a given frequency estimate, respectively. Hence, for our choice of the sub-protocols above, one can easily verify the overall worst case cost of our construction $O(n^{5/2}\mathsf{poly}(\log(d))\log(1/\beta))$.

[5]We may also think of RndGen as being run at the server which then announces the resulting random string to all the users.

The report length of each user is now scaled by KT compared to that of the promise problem, that is, $O\left(n^{3/2}\log(1/\beta)\log(\log(d))\right)$. In the next section, we will discuss an approach that gets it down to 1 bit at the expense of increasing the public coins.

THEOREM 3.4. *Protocol* PROT-*S-Hist given by Algorithm 3 is ϵ-differentially private.*

PROOF. First, observe that Protocol PROT-S-Hist runs Protocol PROT$^{\mathsf{PP}}$-S-Hist$_{\mathsf{PP}}$ over KT channels and runs Protocol PROT-FO once over a separate channel. In the first KT channels, for any fixed sequence of the values of the seed of the hash function, the reports of each user over these channels are independent. Moreover, each user gets assigned to exactly T channels. Fix any user i and any two items $v_i, v_i' \in \mathcal{V}$. Using these observations, one can see that, for any fixed sequence of values of the seed of the hash over these KT channels, the distribution of the report of user i when its item is v_i differs from the distribution when the user's item is v_i' in *at most* $2T$ channels, and in each of these channels, the ratio between the two distributions is at most $e^{\frac{\epsilon}{2T+1}}$ by the differential privacy of PROT$^{\mathsf{PP}}$-S-Hist$_{\mathsf{PP}}$ (note that the input privacy parameter to in Step 9 is $e^{\frac{\epsilon}{2T+1}}$). Hence, by independence of the user's reports over separate channels, the corresponding ratio over all the KT channels is at most $e^{\frac{2T\epsilon}{2T+1}}$. In the separate channel for the frequency oracle protocol, again by the differential privacy of PROT-FO, this ratio is bounded by $e^{\frac{\epsilon}{2T+1}}$. Putting this together with the argument in the previous paragraph completes the proof. □

THEOREM 3.5. *For any set of users' items $\{v_1, ..., v_n\}$ and any $\beta > 0$, there is a number $\eta = O\left(\frac{\log^{\frac{3}{2}}(1/\beta)}{\epsilon}\sqrt{\frac{\log(d)}{n}}\right)$ such that, with probability at least $1 - \beta$, Protocol* PROT-*S-Hist outputs $\left\{\left(v, \hat{f}(v)\right) : v \in \text{LIST}\right\}$ where $\text{LIST} = \{v^* \in \mathcal{V} : f(v^*) \geq \eta\}$ (i.e., a list of all items whose frequencies are greater than η), and the error in the frequency estimates satisfies*

$$\max_{v \in \mathcal{V}} |\hat{f}(v) - f(v)| = O\left(\frac{\log^{\frac{3}{2}}(1/\beta)}{\epsilon}\sqrt{\frac{\log(d)}{n}}\right).$$

(As mentioned before, the frequency estimates of items $v \notin$ LIST are implicitly zero.)

PROOF. Let $\mathcal{U}$ denote the set of the users' items $\{v_1, ..., v_n\}$. We first show that for the setting of K and T in Algorithm 3, running PROT$^{\mathsf{PP}}$-S-Hist$_{\mathsf{PP}}$ over KT channels will isolate every heavy hitter (i.e., every item occurring with frequency at least η) in at least one channel without interference from other items. Let $\mathsf{Heavhit} = \{v^* \in \mathcal{V} : f(v^*) \geq \eta\}$ denote the set of the heavy hitters. Note that $|\mathsf{Heavhit}| \leq \frac{1}{\eta}$.

CLAIM 3.6. *If $\eta \geq \frac{2T+1}{\epsilon}\sqrt{\frac{\log(d)\log(1/\beta)}{n}}$, then with probability at least $1 - \beta/3$ (over the sequence of the seed values $s_1, ..., s_T$ of* Hash*), for every heavy hitter $v^* \in$ Heavhit there is $t \in [T]$ such that $\mathsf{Hash}(\mathsf{s_t}, \mathsf{v}^*) \neq \mathsf{Hash}(\mathsf{s_t}, \mathsf{v})$ for all $v \in \mathcal{U} \setminus \{v^*\}$.*

First, we prove this claim. Fix $v^* \in$ Heavhit. Let $t \in [T]$. Let $\mathsf{Coll}_{s_t}(v^*) \triangleq |\{i \in [n] : \mathsf{Hash}(\mathsf{s}, \mathsf{v}^*) = \mathsf{Hash}(\mathsf{s}, \mathsf{v_i}), \mathsf{v_i} \neq \mathsf{v}^*\}|$ denote the number of collisions between v^* and users' items that are different from v^* when the hash seed is s_t. First, we bound the expected number of such collisions:

$$\mathbb{E}[\mathsf{Coll}_{s_t}(v^*)] \leq \sum_{i: v_i \neq v^*} \frac{1}{K} \leq \frac{n}{K} = \frac{1}{\sqrt{n}}$$

Hence, by Markov's inequality, with probability at least $1 - \frac{1}{\sqrt{n}}$, $\mathsf{Coll}_{s_t}(v^*) = 0$. Hence, with probability at least $1 - \frac{1}{\eta}\left(\frac{1}{\sqrt{n}}\right)^T \geq 1 - \beta/3$, for each $v^* \in \mathsf{Heavhit}$, there exists $t \in [T]$ such that $\mathsf{Coll}_{s_t}(v^*) = 0$, which proves the claim.

This implies that with probability at least $1 - \beta/3$, there is a set $\mathcal{W} \subset [KT]$ of "good" channels whose size is the same as the number of heavy hitters such that each heavy hitter $v^* \in \mathsf{Heavhit}$ is hashed into one of these channels without collisions. Conditioned on this event, let $w \in \mathcal{W}$ and let v^*_w denote the heavy hitter in channel w. By Theorem 3.3, running Protocol $\mathsf{PROT}^{\mathsf{PP}}$-S-Hist$_{\mathsf{PP}}$ over channel w yields the correct estimate of v^*_w with probability at least $1 - \frac{\beta}{3d}$ (Step 9 of Algorithm 3). Hence, with probability at least $1 - \beta/3$, all estimates of $\mathsf{PROT}^{\mathsf{PP}}$-S-Hist$_{\mathsf{PP}}$ in all channels in $\mathcal{W}$ are correct. Hence, at this point, with probability at least $1 - \frac{2\beta}{3}$, LIST contains all the heavy hitters in $\mathsf{Heavhit}$ among other possibly unreliable estimates of $\mathsf{PROT}^{\mathsf{PP}}$-S-Hist$_{\mathsf{PP}}$ for the channels in $[KT] \setminus \mathcal{W}$.

Now, conditioned on the event above, by the error guarantee of FO given by Theorem 2.3, with probability at least $1 - \beta/3$, the maximum error in the frequency estimates of all the items in LIST (Step 13 of Algorithm 3), denoted by ERR (LIST), is

$$O\left(\frac{2T+1}{\epsilon}\sqrt{\frac{\log(d/\beta)}{n}}\right) = O\left(\frac{\log(1/\beta)}{\epsilon}\sqrt{\frac{\log(d/\beta)}{n}}\right).$$

Hence, all those items in LIST with actual frequencies greater than

$$\begin{aligned}\eta &\triangleq \frac{2T+1}{\epsilon}\sqrt{\frac{\log(d)\log(1/\beta)}{n}} + \textsc{Err}\,(\textsc{List}) \\ &= O\left(\frac{\log^{\frac{3}{2}}(1/\beta)}{\epsilon}\sqrt{\frac{\log(d)}{n}}\right)\end{aligned}$$

will be kept in LIST whereas all those items with frequency less than η will be removed. Note that the frequency estimates that are implicitly assumed to be zero of those items that are not on the list cannot have error greater than η since their actual frequencies are less than η. This completes the proof. □

4. THE FULL PROTOCOL

4.1 Generic Protocol with 1-Bit Reports

In this section, we give a generic approach that transforms *any* private protocol in the distributed setting (not necessarily for frequency estimation or succinct histograms) to a private distributed protocol where the report of each user is a single bit at the expense of adding to the overall original public randomness a number of bits that is $O(n\tau)$ where τ is the length of each user's report in the original protocol. As mentioned in the introduction, the transformation is a modification of the general compression technique of McGregor et al. [20].

Let $\mathsf{GenPROT}$ be any private distributed protocol in which n users are communicating with an untrusted server. $\mathsf{GenPROT}$ follows the following general steps. Each user $i \in [v]$ has a data point $v_i \in \mathcal{V} = [d]$. Let $\mathcal{Q}_i : \mathcal{V} \cup \{\perp\} \to \mathcal{Z}$ be *any* ϵ-local randomizer used by user $i \in [n]$. We assume, w.l.o.g., that $\mathcal{Q}_i$ may also take a special symbol $\perp$ as an input. Each user runs its ϵ-local randomizer $\mathcal{Q}_i$ on its data v_i (and any public randomness in the protocol, if any) and outputs a report $\mathbf{z}_i$. For simplicity, each report $\mathbf{z}_i$ is assumed to be a binary string of length τ. Let $\mathsf{stat} : \mathcal{V}^n \to \mathcal{C}$ be some statistic about the data that the server wishes to estimate where $\mathcal{C}$ is some bounded subset of $\mathbb{R}^k$ for some integer $k > 0$. The server collects the reports $\{\mathbf{z}_i : i \in [n]\}$ and runs some algorithm $\mathcal{A}_{\mathsf{stat}}$ on the reports (and the public randomness) and outputs an estimate $\widehat{\mathsf{stat}} \in \mathcal{C}$ of $\mathsf{stat}\,(v_1, ..., v_n)$.

We now give a generic construction 1-Bit-$\mathsf{GenPROT}$ (Algorithm 4) for $\mathsf{GenPROT}$ where each user's report is one bit.

Algorithm 4 1-Bit-PROT: ϵ-**LDP** Generic 1-Bit Protocol

Input: Users' inputs $\{v_i \in \mathcal{V} : i \in [n]\}$ and a privacy parameter $\epsilon \leq \ln(2)$.

1: Generate n independent public strings $y_1 \leftarrow \mathcal{Q}_1(\perp), ..., y_n \leftarrow \mathcal{Q}_n(\perp)$.
2: **for** Users $i = 1$ to n **do**
3: Compute $p_i = \frac{1}{2}\frac{\Pr[\mathcal{Q}_i(v_i)=y_i]}{\Pr[\mathcal{Q}_i(\perp)=y_i]}$.
4: Sample a bit b_i from Bernoulli(p_i) and sends it to the server.
5: Reports $\leftarrow \emptyset$. {Server initialize the set of collected reports.}
6: **for** $i = 1$ to n **do**
7: Server checks if $b_i = 1$, add y_i to Reports.
8: $\widehat{\mathsf{stat}} \leftarrow \mathcal{A}_{\mathsf{stat}}$ (Reports) . {Run algorithm $\mathcal{A}_{\mathsf{stat}}$ on the collected reports to obtain an estimate of the desired statistic as described in the original protocol $\mathsf{GenPROT}$.}
9: **return** $\widehat{\mathsf{stat}}$.

Note that the only additional computational cost in this generic transformation is in Step 3. If computing these probabilities can be done efficiently, then this transformation preserves the computational efficiency of the original protocol.

THEOREM 4.1. *Protocol* 1-Bit-PROT *given by Algorithm 4 is* ϵ-**LDP**.

PROOF. Consider the output bit b_i of any user $i \in [n]$. First, note that p_i (in Step 3) is a valid probability since for any item $v_i \in \mathcal{V}$, the right-hand side of Step 3 is at most $\frac{e^\epsilon}{2}$ by ϵ-differential privacy of $\mathcal{Q}_i$, and since $\epsilon \leq \ln(2)$, $p_i \leq 1$. For any $v \in \mathcal{V}$ and any public string y_i, let $p_i(v, y_i)$ denote the conditional probability that $b_i = 1$ given that $\mathcal{Q}_i(\perp) = y_i$ when the item of user i is v. Let $v, v' \in \mathcal{V}$ be any two items. It is easy to see that $\frac{p_i(v,y_i)}{p_i(v',y_i)} = \frac{\Pr[\mathcal{Q}_i(v)=y_i]}{\Pr[\mathcal{Q}_i(v')=y_i]}$ which lies in $[e^{-\epsilon}, e^\epsilon]$ by ϵ-differential privacy of $\mathcal{Q}_i$. One can also verify that $\frac{1-p_i(v,y_i)}{1-p_i(v',y_i)}$ also lies in $[e^{-\epsilon}, e^\epsilon]$. □

One important feature in the construction above is that the conditional distribution of the public string y_i given that $b_i = 1$ is exactly the same as the distribution of $\mathcal{Q}_i(v_i)$, and hence, upon receiving a bit $b_i = 1$ from user i, the server's view of y_i is the same as its view of an actual report $\mathbf{z}_i \leftarrow \mathcal{Q}_i(v_i)$.

Note also that the probability that a user $i \in [n]$ accepts (sets $b_i = 1$) taken over the randomness of y_i is

$$\frac{1}{2}\sum_y \frac{\Pr\left[\mathcal{Q}_i\left(v_i\right) = y\right]}{\Pr\left[\mathcal{Q}_i(\perp) = y\right]} \cdot \Pr\left[\mathcal{Q}_i(\perp) = y\right] = \frac{1}{2}.$$

Key statement: *The two facts above show that our protocol is functionally equivalent to: first, sampling a subset of the users where each user is sampled independently with probability* $1/2$*, then, running the original protocol* $\mathsf{GenPROT}$ *on the sample. Thus, if the original protocol is resilient to sampling, i.e., its error performance (with respect to some notion of error) is not essentially affected by this sampling step, then the generic transformation given by Algorithm 4 will have essentially the same error performance.*

4.2 Efficient Construction with 1-Bit Reports

We now apply the generic transformation discussed above to our efficient protocol PROT-S-Hist (Algorithm 3) to obtain an *efficient* private protocol for succinct histograms with 1-bit reports

and optimal error. The fact that such protocol has the same error as PROT-S-Hist follows from the key statement above and the fact that PROT-S-Hist is resilient to sampling.

THEOREM 4.2. *The* 1*-Bit Protocol for succinct histograms is* ϵ*-differentially private.*

THEOREM 4.3. *The* 1*-Bit Protocol for succinct histograms provides the same guarantees of Protocol* PROT*-S-Hist given in Theorem 3.2.*

Computational efficiency: To show that the protocol remains efficient after this transformation, we argue that the probabilities in Step 3 of Algorithm 4 can be computed efficiently in our case. The overall ϵ-local randomizer $\mathcal{Q}_i^{\mathsf{Full}}$ at each user i over all the $KT+1$ parallel channels in PROT-S-Hist is described in Algorithm 5. Note that given the user's item v_i and the seed of the hash, the $KT+1$ components of $\mathcal{Q}_i^{\mathsf{Full}}(v_i)$ are independent. Moreover, note that $(K-1)T$ of these components have the same (uniform) distribution since each user gets assigned by the hash function to only $T+1$ channels and in the remainder channels the user's report is uniformly random. Hence, to execute Step 3 of Algorithm 4, each user in our case only needs to compute $T+1$ probabilities out of the total $KT+1$ components. This is easy because of the way the basic randomizer $\mathcal{R}$ works. To see this, first note that each y_i (referring to the public string y_i in Algorithm 4) is now a sequence of $(\mathsf{index}, \mathsf{bit})$ pairs: $(j_1, b_{j_1}), \ldots, \left(j_{KT+1}, b_{j_{KT+1}}\right)$. To compute the probability corresponding to one of the $T+1$ item-dependent components of $\mathcal{Q}_i^{\mathsf{Full}}(v_i)$, each user first locates in the public string y_i the pair (j,b) corresponding to this component. Then, it compares the sign of the j-th bit of the encoding[6] of its item v_i with the sign of b. If signs are equal, then the desired probability is $\frac{e^\epsilon}{1+e^\epsilon}$, otherwise it is $\frac{1}{1+e^\epsilon}$. Hence, the computational cost of this step (per user) is $O\left(T\log\left(m_{\mathsf{PP}}\right)+\log\left(m_{\mathsf{FO}}\right)\right) = O(\log(\log(d))\log(1/\beta)+\log(n))$ where m_{PP} is the length of the encoding $\mathbf{c}(v_i)$ used in the promise problem protocol $\mathsf{PROT}^{\mathsf{PP}}$-S-Hist$_{\mathsf{PP}}$ and m_{FO} is the length of the encoding ϕ_{v_i} used in the frequency oracle protocol PROT-FO. Thus, at worst the overall computational cost of the 1-Bit protocol is the same as that of protocol PROT-S-Hist.

Algorithm 5 $\mathcal{Q}_i^{\mathsf{Full}}$: ϵ-Local Randomizer of User i in PROT-S-Hist (Algorithm3)

Input: item $v_i \in \mathcal{V}$, privacy parameter ϵ, seeds of Hash $s_1, ..., s_T$.

1: **for** $t = 1$ to T **do**
2: **for** Channels $k = 1$ to K **do**
3: If $\mathsf{Hash}(\mathsf{s_t}, \mathsf{v_i}) \neq \mathsf{k}$, set $\mathbf{z}_i^{(t,k)} = \mathcal{R}_i(\mathbf{0}, \epsilon)$. Else, set $\mathbf{z}_i^{(t,k)} = \mathcal{R}_i(\mathbf{c}(v_i), \epsilon)$. { $\mathbf{z}_i^{(t,k)}$ denotes the report of user i in the k-th channel in the t-th group.}
4: Set $\mathbf{z}_i^{(fo)} = \mathcal{R}_i(\phi_{v_i}, \epsilon)$. {$\phi_{v_i}$ is the v_i-th column of Φ the encoding matrix in the construction of the frequency oracle FO.}
5: **return** $\mathbf{z}_i = \left(\mathbf{z}_i^{(t,k)}, \mathbf{z}_i^{(fo)} : t = 1, ..., T; k = 1, ..., K\right)$.

5. TIGHT LOWER BOUND ON ERROR

We derive a matching lower bound on the error of (ϵ,δ)-**LDP** frequency oracles and succinct histograms for all $\delta = o\left(\frac{1}{n\log(n)}\right)$. Our lower bound implies that there is no advantage of (ϵ,δ) algorithms over pure ϵ algorithms in terms of asymptotic error for all the meaningful settings of δ. Our approach is inspired by some of the techniques in [4] used for lower bounds on multinomial estimation error in the pure ϵ local model. We make their framework more modular, and show that it can be used to prove lower bounds for (ϵ,δ)-local differentially private protocols. Our lower bound is formally stated in the following theorem.

THEOREM 5.1. *For any* $\epsilon = O(1)$ *and* $0 \leq \delta \leq o\left(\frac{1}{n\log(n)}\right)$. *For any sequence* $\mathcal{Q}_i : \mathcal{V} \to \mathcal{Z}, i \in [n]$ *of* (ϵ,δ)-**LDP** *algorithms, and for any algorithm* $\mathcal{B} : \mathcal{Z}^n \to [0,1]^d$, *there exists a distribution* $\mathcal{P}$ *over* $\mathcal{V}$ *(from which* $v_i, i \in [n]$ *are sampled in i.i.d. fashion) such that the expected error with respect to such* $\mathcal{P}$ *satisfies*

$$\mathbb{E}\left[\max_{v\in[d]}|\hat{f}(v) - f(v)|\right] = \Omega\left(\min\left(\frac{1}{\epsilon}\sqrt{\frac{\log(d)}{n}}, 1\right)\right).$$

We discuss the main steps of our technique next. For more details on the proof, we refer the reader to the full version [2].

5.1 Our Technique

In a scenario where the item of each user is drawn independently from an unknown distribution on $\mathcal{V}$, we first derive a lower bound on the expected worst-case error (the minimax rate) in estimating the right distribution. We then show using standard concentration bounds that this implies a lower bound on the maximum error in estimating the actual frequencies of all the items in $\mathcal{V}$.

To obtain a lower bound on the minimax error, we first define the notion of an η-degrading channel which is a noise operator that, given a user's item as input, outputs the same item with probability η, and outputs a uniform random item from $\mathcal{V}$ otherwise. Formally, for any $\eta \in [0,1]$, an η-degrading channel $\mathbf{W}^{(\eta)} : \mathcal{V} \to \mathcal{V}$ is a randomized mapping that is defined as follows: for every $v \in \mathcal{V}$,

$$\mathbf{W}^{(\eta)}(v) = \begin{cases} v & \text{with probability } \eta \\ U, & \text{with probability } 1-\eta \end{cases} \tag{3}$$

where U is a uniform random variable over $\mathcal{V}$.

We compare two scenarios: in the first scenario, each user feeds its item first to an η-degrading channel, then feeds its output into its (ϵ,δ) local randomizer to generate a report, whereas the second scenario is the normal scenario where the user feeds its item directly into its local randomizer. We then show that a lower bound of $\Omega(1)$ on the minimax error in the first scenario implies a lower bound of $\Omega(\eta)$ in the second scenario.

Thus, to reach our result, it would suffice to show that a lower bound of $\Omega(1)$ is true in the first scenario with an η-degrading channel when $\eta = \Omega\left(\frac{1}{\epsilon}\sqrt{\frac{\log(d)}{n}}\right)$. To derive the $\Omega(1)$ lower bound in the first scenario, we proceed as follows.

First, we derive the following upper bound on the mutual information between a uniform random item V from $\mathcal{V}$ and the output of an (ϵ,δ)-local randomizer with input V.

CLAIM 5.2. *Let* V *be uniformly distributed over* $\mathcal{V}$. *Let* $\mathcal{Q} : \mathcal{V} \to \mathcal{Z}$ *be an* (ϵ,δ)-**LDP** *algorithm and let* Z *denote* $\mathcal{Q}(V)$. *Then, we have*

$$I(V;Z) = O\left(\epsilon^2 + \frac{\delta}{\epsilon}\log(d) + \frac{\delta}{\epsilon}\log(\epsilon/\delta)\right).$$

Then, we prove that the application of an η-degrading channel on a user's item *amplifies privacy*, namely, scales down both ϵ and δ by η.

[6]This encoding is either $\mathbf{c}(v_i)$ or ϕ_{v_i} depending on whether we are at Step 3 or Step 4 of Algorithm 5.

CLAIM 5.3. *The composition* $\mathcal{Q}\left(\mathbf{W}^{(\eta)}(\cdot)\right)$ *of an* η*-degrading channel* $\mathbf{W}^{(\eta)}$ *(defined in (3)) with an* (ϵ,δ)-$\mathbf{LDP}$ *algorithm* $\mathcal{Q}$ *yields a* $(O(\eta\epsilon), O(\eta\delta))$-$\mathbf{LDP}$ *algorithm.*

This implies that in the first scenario above with an η-degrading channel, the mutual information between the users' items and the corresponding outputs of their (ϵ,δ)-local randomizers is

$$I(v_1,...,v_n;\mathbf{z}_1,...\mathbf{z}_n) \leq \sum_{i=1}^{n} I(v_i;\mathbf{z}_i) = O\left(n\eta^2\epsilon^2 + n\frac{\delta}{\epsilon}\log(d\epsilon/\delta)\right)$$

Fano's inequality implies that the minimax error in this scenario is bounded from below as

$$\mathsf{MinMaxError}_\eta \geq 1 - \frac{I(v_1,...,v_n;\mathbf{z}_1,...\mathbf{z}_n)+1}{\log(d)}$$

Hence, by an appropriate setting of $\eta = \Omega\left(\min\left(\sqrt{\frac{\log(d)}{\epsilon^2 n}}, 1\right)\right)$ and for $\delta = o\left(\frac{1}{n\log(n)}\right)$, our mutual information bound above together with Fano's inequality implies that the error in the first scenario is $\Omega(1)$. This concludes the proof of our lower bound.

Acknowledgments

R.B. and A.S. were funded by NSF awards #0747294 and #0941553. Some of this work was done while A.S. was on sabbatical at Boston University's Hariri Center for Computation and at Harvard University's CRCS. We are grateful to helpful conversations with Úlfar Erlingsson, Aleksandra Korolova, Frank McSherry, Ilya Mironov, Kobbi Nissim, and Vasyl Pihur.

References

[1] S. Agrawal and J. R. Haritsa. A framework for high-accuracy privacy-preserving mining. In *ICDE*, pages 193–204. IEEE Computer Society, 2005.

[2] R. Bassily and A. Smith. Local, private, efficient protocols for succinct histograms. *arXiv*, April 2015.

[3] J. Blocki, A. Blum, A. Datta, and O. Sheffet. The johnson-lindenstrauss transform itself preserves differential privacy. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 410–419, 2012.

[4] J. C. Duchi, M. I. Jordan, and M. J. Wainwright. Local privacy and statistical minimax rates. In *IEEE Symp. on Foundations of Computer Science (FOCS)*, 2013.

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

[6] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography Conference*, pages 265–284. Springer, 2006.

[7] C. Dwork, M. Naor, T. Pitassi, and G. N. Rothblum. Differential privacy under continual observation. STOC '10, pages 715–724. ACM, 2010.

[8] C. Dwork, M. Noar, T. Pitassi, G. Rothblum, and S. Yekhanin. Pan-private streaming algorithms. In *Symposium on Innovations in Computer Science (ICS)*, 2010.

[9] Ú. Erlingsson, A. Korolova, and V. Pihur. RAPPOR: randomized aggregatable privacy-preserving ordinal response. *ACM Symposium on Computer and Communications Security (CCS)*, 2014.

[10] A. Evfimievski, J. Gehrke, and R. Srikant. Limiting privacy breaches in privacy preserving data mining. In *PODS*, pages 211–222. ACM, 2003.

[11] G. Fanti, V. Pihur, and Ú. Erlingsson. Building a rappor with the unknown: Privacy-preserving learning of associations and data dictionaries. In *arXiv:1503.01214 [cs.CR]*, 2015.

[12] A. C. Gilbert, S. Guha, P. Indyk, S. Muthukrishnan, and M. Strauss. Near-optimal sparse fourier representations via sampling. STOC 2002, pages 152–161. ACM, 2002.

[13] V. Guruswami. *List Decoding of Error-Correcting Codes*. PhD thesis, 2002.

[14] J. Hsu, S. Khanna, and A. Roth. Distributed private heavy hitters. In *ICALP (1)*, pages 461–472, 2012.

[15] S. P. Kasiviswanathan, H. K. Lee, K. Nissim, S. Raskhodnikova, and A. Smith. What can we learn privately? In *FOCS*, 2008.

[16] M. Kearns. Efficient noise-tolerant learning from statistical queries. *Journal of the ACM*, 45(6):983–1006, 1998. ISSN 0004-5411. Preliminary version in *proceedings of STOC'93*.

[17] K. Kenthapadi, A. Korolova, I. Mironov, and N. Mishra. Privacy via the johnson-lindenstrauss transform. *CoRR*, abs/1204.2606, 2012.

[18] E. Kushilevitz and Y. Mansour. Learning decision trees using the fourier spectrum. STOC'91, pages 455–464. ACM, 1991.

[19] C. McDiarmid. On the method of bounded differences. In *In Surveys in Combinatorics*, pages 148–188. Cambridge University Press, 1989.

[20] A. McGregor, I. Mironov, T. Pitassi, O. Reingold, K. Talwar, and S. P. Vadhan. The limits of two-party differential privacy. In *FOCS*, pages 81–90, 2010.

[21] N. Mishra and M. Sandler. Privacy via pseudorandom sketches. In *PODS*, pages 143–152. ACM, 2006.

[22] J. Upadhyay. Random projections, graph sparsification, and differential privacy. In *Advances in Cryptology - ASIACRYPT*, pages 276–295, 2013.