Locally Differentially Private Frequency Estimation with Consistency

Tianhao Wang[1], Milan Lopuhaä-Zwakenberg[2], Zitao Li[1], Boris Skoric[2], Ninghui Li[1]
[1]Purdue University, [2]Eindhoven University of Technology
{tianhaowang, li2490, ninghui}@purdue.edu, {m.a.lopuhaa, b.skoric}@tue.nl

***Abstract*—Local Differential Privacy (LDP) protects user privacy from the data collector. LDP protocols have been increasingly deployed in the industry. A basic building block is frequency oracle (FO) protocols, which estimate frequencies of values. While several FO protocols have been proposed, the design goal does not lead to optimal results for answering many queries. In this paper, we show that adding post-processing steps to FO protocols by exploiting the knowledge that all individual frequencies should be non-negative and they sum up to one can lead to significantly better accuracy for a wide range of tasks, including frequencies of individual values, frequencies of the most frequent values, and frequencies of subsets of values. We consider 10 different methods that exploit this knowledge differently. We establish theoretical relationships between some of them and conducted extensive experimental evaluations to understand which methods should be used for different query tasks.**

I. Introduction

Differential privacy (DP) [12] has been accepted as the *de facto* standard for data privacy. Recently, techniques for satisfying DP in the local setting, which we call LDP, have been studied and deployed. In this setting, there are many users and one aggregator. The aggregator does not see the actual private data of each individual. Instead, each user sends randomized information to the aggregator, who attempts to infer the data distribution based on that. LDP techniques have been deployed by companies like Apple [1], Google [14], Microsoft [9], and Alibaba [32]. Examples of use cases include collecting users' default browser homepage and search engine, in order to understand the unwanted or malicious hijacking of user settings; or frequently typed emoji's and words, to help with keyboard typing recommendation.

The fundamental tools in LDP are mechanisms to estimate frequencies of values. Existing research [14], [5], [31], [2], [36] has developed frequency oracle (FO) protocols, where the aggregator can estimate the frequency of any chosen value in the specified domain (fraction of users reporting that value). While these protocols were designed to provide unbiased estimations of individual frequencies while minimizing the estimation variance [31], they can perform poorly for some tasks. In [17], it is shown that when one wants to query the frequency of all values in the domain, one can obtain significant accuracy improvement by exploiting the belief that the distribution likely follows power law. Also, some applications naturally require querying the sums of frequencies for values in a subset. For example, with the estimation of each emoji's frequency, one may be interested in understanding what categories of emoji's are more popular and need to issue subset frequency queries. For another example, in [38], multiple attributes are encoded together and reported using LDP, and recovering the distribution for each attribute separately requires computing the frequencies of sets of encoded values. For frequencies of a subset of values, simply summing up the estimations of all values is far from optimal, especially when the input domain is large.

We note that the problem of answering queries using information obtained from the frequency oracle protocols is an estimation problem. Existing methods such as those in [31] do not utilize any prior knowledge of the distribution to be estimated. Due to the significant amount of noise needed to satisfy LDP, the estimations for many values may be negative. Also, some LDP protocols may result in the total sum of frequencies to be different from one. In this paper, we show that one can develop better estimation methods by exploiting the universal fact that all frequencies are non-negative and they sum up to 1.

Interestingly, when taking advantage of such prior knowledge, one introduces biases in the estimations. For example, when we impose the non-negativity constraint, we are introducing positive biases in the estimation as a side effect. Essentially, when we exploit prior beliefs, the estimations will be biased towards the prior beliefs. These biases can cause some queries to be much more inaccurate. For example, changing all negative estimations to zero improves accuracy for frequency estimations of individual values. However, the introduced positive biases accumulate for range queries. Different methods to utilize the prior knowledge introduces different forms of biases, and thus have different impacts for different kinds of queries.

In this paper, we consider 10 different methods, which utilizes prior knowledge differently. Some methods enforce only non-negativity; some other methods enforce only that all estimations sum to 1; and other methods enforce both. These methods can also be combined with the "Power" method in [17] that exploits power law assumption.

We evaluate these methods on three tasks, frequencies of

Network and Distributed Systems Security (NDSS) Symposium 2020
23-26 February 2020, San Diego, CA, USA
ISBN 1-891562-61-4
https://dx.doi.org/10.14722/ndss.2020.24157
www.ndss-symposium.org

individual values, frequencies of the most frequent values, and frequencies of subsets of values. We find that there is no single method that out-performs other methods for all tasks. A method that exploits only non-negativity performs the best for individual values; a method that exploits only the summing-to-one constraint performs the best for frequent values; and a method that enforces both can be applied in conjunction with Power to perform the best for subsets of values.

To summarize, the main contributions of this paper are threefold:

- We introduced the consistency properties as a way to improve accuracy for FO protocols under LDP, and summarized 10 different post-processing methods that exploit the consistency properties differently.
- We established theoretical relationships between Constrained Least Squares and Maximum Likelihood Estimation, and analyze which (if any) estimation biases are introduced by these methods.
- We conducted extensive experiments on both synthetic and real-world datasets, the results improved the understanding on the strengths and weaknesses of different approaches.

Roadmap. In Section II, we give the problem definition, followed by the background information on FO in Section III. We present the post-processing methods in Section IV. Experimental results are presented in V. Finally we discuss related work in Section VI and provide concluding remarks in Section VII.

II. Problem Setting

We consider the setting where there are many *users* and one *aggregator*. Each user possesses a value v from a finite domain D, and the aggregator wants to learn the distribution of values among all users, in a way that protects the privacy of individual users. More specifically, the aggregator wants to estimate, for each value $v \in D$, the fraction of users having v (the number of users having v divided by the population size). Such protocols are called *frequency oracle (*FO*)* protocols under Local Differential Privacy (LDP), and they are the key building blocks of other LDP tasks.

Privacy Requirement. An FO protocol is specified by a pair of algorithms: Ψ is used by each user to perturb her input value, and Φ is used by the aggregator. Each user sends $\Psi(v)$ to the aggregator. The formal privacy requirement is that the algorithm $\Psi(\cdot)$ satisfies the following property:

Definition 1 (ϵ-Local Differential Privacy)**.** *An algorithm $\Psi(\cdot)$ satisfies ϵ-local differential privacy (ϵ-LDP), where $\epsilon \geq 0$, if and only if for any input $v, v' \in D$, we have*

$$\forall y \in \Psi(D):\ \mathsf{Pr}\left[\Psi(v) = y\right] \leq e^{\epsilon}\, \mathsf{Pr}\left[\Psi(v') = y\right],$$

where $\Psi(D)$ is discrete and denotes the set of all possible outputs of Ψ.

Since a user never reveals v to the aggregator and reports only $\Psi(v)$, the user's privacy is still protected even if the aggregator is malicious.

Utility Goals. The aggregator uses Φ, which takes the vector of all reports from users as the input, and produces $\tilde{\mathbf{f}} = \langle \tilde{f}_v \rangle_{v \in D}$, the estimated frequencies of the $v \in D$ (i.e., the fraction of users who have input value v). As Ψ is a randomized function, the resulting $\tilde{\mathbf{f}}$ becomes inaccurate.

In existing work, the design goal for Ψ and Φ is that the estimated frequency for each v is unbiased, and the variance of the estimation is minimized. As we will show in this paper, these may not result in the most accurate answers to different queries.

In this paper, we consider three different query scenarios 1) query the frequency of every value in the domain, 2) query the aggregate frequencies of subsets of values, and 3) query the frequencies of the most frequent values. For each value or set of values, we compute its estimate and the ground truth, and calculate their difference, measured by Mean of Squared Error (MSE).

Consistency. We will show that the utility of existing mechanisms can be improved by enforcing the following consistency requirement.

Definition 2 (Consistency)**.** *The estimated frequencies are consistent if and only if the following two conditions are satisfied:*

1) The estimated frequency of each value is non-negative.
2) The sum of the estimated frequencies is 1.

III. Frequency Oracle Protocols

We review the state-of-the-art frequency oracle protocols. We utilize the generalized view from [31] to present the protocols, so that our post-processing procedure can be applied to all of them.

A. Generalized Random Response (GRR)

This FO protocol generalizes the *randomized response* technique [35]. Here each user with private value $v \in D$ sends the true value v with probability p, and with probability $1-p$ sends a randomly chosen $v' \in D \setminus \{v\}$. Suppose the domain D contains $d = |D|$ values, the perturbation function is formally defined as

$$\forall_{y \in D}\ \mathsf{Pr}\left[\Psi_{\mathsf{GRR}(\epsilon,d)}(v) = y\right] = \begin{cases} p = \frac{e^{\epsilon}}{e^{\epsilon}+d-1}, & \text{if } y = v \\ q = \frac{1}{e^{\epsilon}+d-1}, & \text{if } y \neq v \end{cases} \quad (1)$$

This satisfies ϵ-LDP since $\frac{p}{q} = e^{\epsilon}$.

From a population of n users, the aggregator receives a length-n vector $\mathbf{y} = \langle y_1, y_2, \cdots, y_n \rangle$, where $y_i \in D$ is the reported value of the i-th user. The aggregator counts the number of times each value v appears in $\mathbf{y}$ and produces a length-d vector $\mathbf{c}$ of natural numbers. Observe that the components of $\mathbf{c}$ sum up to n, i.e., $\sum_{v \in D} c_v = n$. The

aggregator then obtains the estimated frequency vector $\tilde{\mathbf{f}}$ by scaling each component of $\mathbf{c}$ as follows:

$$\tilde{f}_v = \frac{\frac{c_v}{n} - q}{p - q} = \frac{\frac{c_v}{n} - \frac{1}{e^\epsilon + d - 1}}{\frac{e^\epsilon - 1}{e^\epsilon + d - 1}}$$

As shown in [31], the estimation variance of GRR grows linearly in d; hence the accuracy deteriorates fast when the domain size d increases. This motivated the development of other FO protocols.

B. Optimized Local Hashing (OLH)

This FO deals with a large domain size d by first using a random hash function to map an input value into a smaller domain of size g, and then applying randomized response to the hash value in the smaller domain. In OLH, the reporting protocol is

$$\Psi_{\mathsf{OLH}(\epsilon)}(v) \coloneqq \langle H,\ \Psi_{\mathsf{GRR}(\epsilon,g)}(H(v))\rangle,$$

where H is randomly chosen from a family of hash functions that hash each value in D to $\{1 \dots g\}$, and $\Psi_{\mathsf{GRR}(\epsilon,g)}$ is given in (1), while operating on the domain $\{1 \dots g\}$. The hash family should have the property that the distribution of each v's hashed result is uniform over $\{1 \dots g\}$ and independent from the distributions of other input values in D. Since H is chosen independently of the user's input v, H by itself carries no meaningful information. Such a report $\langle H, r\rangle$ can be represented by the set $Y = \{y \in D \mid H(y) = r\}$. The use of a hash function can be viewed as a compression technique, which results in constant size encoding of a set. For a user with value v, the probability that v is in the set Y represented by the randomized report $\langle H, r\rangle$ is $p = \frac{e^\epsilon - 1}{e^\epsilon + g - 1}$ and the probability that a user with value $\neq v$ is in Y is $q = \frac{1}{g}$.

For each value $x \in D$, the aggregator first computes the vector $\mathbf{c}$ of how many times each value is in the reported set. More precisely, let Y_i denote the set defined by the user i, then $c_v = |\{i \mid H(v) \in Y_i\}|$. The aggregator then scales it:

$$\tilde{f}_v = \frac{\frac{c_v}{n} - 1/g}{p - 1/g} \tag{2}$$

In OLH, both the hashing step and the randomization step result in information loss. The choice of the parameter g is a tradeoff between losing information during the hashing step and losing information during the randomization step. It is found that the estimation variance when viewed as a continuous function of g is minimized when $g = e^\epsilon + 1$ (or the closest integer to $e^\epsilon + 1$ in practice) [31].

C. Other FO Protocols

Several other FO protocols have been proposed. While they take different forms when originally proposed, in essence, they all have the user report some encoding of a subset $Y \subseteq D$, so that the user's true value has a probability p to be included in Y and any other value has a probability $q < p$ to be included in Y. The estimation method used in GRR and OLH (namely, $\tilde{f}_v = \frac{c_v/n - q}{p - q}$) equally applies.

Optimized Unary Encoding [31] encodes a value in a size-d domain using a length-d binary vector, and then perturbs each bit independently. The resulting bit vector encodes a set of values. It is found in [31] that when d is large, one should flip the 1 bit with probability $1/2$, and flip a 0 bit with probability $1/e^\epsilon$. This results in the same values of p, q as OLH, and has the same estimation variance, but has higher communication cost (linear in domain size d).

Subset Selection [36], [30] method reports a randomly selected subset of a fixed size k. The sensitive value v is included in the set with probability $p = 1/2$. For any other value, it is included with probability $q = p \cdot \frac{k-1}{d-1} + (1-p) \cdot \frac{k}{d-1}$. To minimize estimation variance, k should be an integer equal or close to $d/(e^\epsilon + 1)$. Ignoring the integer constraint, we have $q = \frac{1}{2} \cdot \frac{2k-1}{d-1} = \frac{1}{2} \cdot \frac{2\frac{d}{e^\epsilon+1} - 1}{d-1} = \frac{1}{e^\epsilon+1} \cdot \frac{d-(e^\epsilon+1)/2}{d-1} < \frac{1}{e^\epsilon+1}$. Its variance is smaller than that of OLH. However, as d increases, the term $\frac{d-(e^\epsilon+1)/2}{d-1}$ gets closer and closer to 1. For a larger domain, this offers essentially the same accuracy as OLH, with higher communication cost (linear in domain size d).

Hadamard Response [4], [2] is similar to Subset Selection with $k = d/2$, where the Hadamard transform is used to compress the subset. The benefit of adopting this protocol is to reduce the communication bandwidth (each user's report is of constant size). While it is similar to OLH with $g = 2$, its aggregation part Φ faster, because evaluating a Hadamard entry is practically faster than evaluating hash functions. However, this FO is sub-optimal when $g = 2$ is sub-optimal.

D. Accuracy of Frequency Oracles

In [31], it is proved that $\tilde{f}_v = \frac{c_v/n - q}{p - q}$ produces unbiased estimates. That is, $\forall v \in D,\ \mathbb{E}\left[\tilde{f}_v\right] = f_v$. Moreover, $\tilde{f}_v$ has variance

$$\sigma_v^2 = \frac{q(1-q) + f_v(p-q)(1-p-q)}{n(p-q)^2} \tag{3}$$

As c_v follows Binomial distribution, by the central limit theorem, the estimate $\tilde{f}_v$ can be viewed as the true value f_v plus a Normally distributed noise:

$$\tilde{f}_v \approx f_v + \mathcal{N}(0, \sigma_v). \tag{4}$$

When d is large and ϵ is not too large, $f_v(p-q)(1-p-q)$ is dominated by $q(1-q)$. Thus, one can approximate Equation (3) and (4) by ignoring the f_v. Specifically,

$$\sigma^2 \approx \frac{q(1-q)}{n(p-q)^2}, \tag{5}$$

$$\tilde{f}_v \approx f_v + \mathcal{N}(0, \sigma). \tag{6}$$

As the probability each user's report support each value is independent, we focus on post-processing $\tilde{\mathbf{f}}$ instead of $\mathbf{Y}$.

IV. Towards Consistent Frequency Oracles

While existing state-of-the-art frequency oracles are designed to provide unbiased estimations while minimizing the variance, it is possible to further reduce the variance by performing post-processing steps that use prior knowledge to adjust the estimations. For example, exploiting the property that all frequency counts are non-negative can reduce the variance; however, simply turning all negative estimations to 0 introduces a systematic positive bias in all estimations. By also ensuring the property that the sum of all estimations must add up to 1, one ensures that the sum of the biases for all estimations is 0. However, even though the biases cancel out when summing over the whole domain, they still exist. There are different post-processing methods that were explicitly proposed or implicitly used. They will result in different combinations of variance reduction and bias distribution. Selecting a post-processing method is similar to considering the bias-variance tradeoff in selecting a machine learning algorithm.

We study the property of several post-processing methods, aiming to understand how they compare under different settings, and how they relate to each other. Our goal is to identify efficient post-processing methods that can give accurate estimations for a wide variety of queries. We first present the baseline method that does not do any post-processing.

- **Base**: *We use the standard* FO *as presented in Section III to obtain estimations of each value.*

Base has no bias, and its variance can be analytically computed (e.g., using [31]).

A. Baseline Methods

When the domain is large, there will be many values in the domain that have a zero or very low true frequency; the estimation of them may be negative. To overcome negativity, we describe three methods: Base-Pos, Post-Pos, and Base-Cut.

- **Base-Pos**: *After applying the standard* FO*, we convert all negative estimations to* 0*.*

This satisfies non-negativity, but the sum of all estimations is likely to be above 1. This reduces variance, as it turns erroneous negative estimations to 0, closer to the true value. As a result, for each individual value, Base-Pos results in an estimation that is at least as accurate as the Base method. However, this introduces systematic positive bias, because some negative noise are removed or reduced by the process, but the positive noise are never removed. This positive bias will be reflected when answering subset queries, for which Base-Pos results in biased estimations. For larger-range queries, the bias can be significant.

Lemma 1. *Base-Pos will introduce positive bias to all values.*

Proof. The outputs of standard FO are unbiased estimation, which means for any v,

$$f_v = \mathbb{E}\left[\tilde{f}_v\right] = \mathbb{E}\left[\tilde{f}_v \cdot \mathbf{1}[\tilde{f}_v \geq 0]\right] + \mathbb{E}\left[\tilde{f}_v \cdot \mathbf{1}[\tilde{f}_v < 0]\right]$$

As Base-Pos changes all negative estimated frequencies to 0, we have

$$\mathbb{E}\left[f'_v\right] = \mathbb{E}\left[\tilde{f}_v \cdot \mathbf{1}[\tilde{f}_v \geq 0]\right]$$

After enforcing non-negativity constraints, the bias will be $\mathbb{E}\left[f'_v\right] - f_v > 0$. □

- **Post-Pos**: *For each query result, if it is negative, we convert it to* 0*.*

This method does not post-process the estimated distribution. Rather, it post-processes each query result individually. For subset queries, as the results are typically positive, Post-Pos is similar to Base. On the other hand, when the query is on a single item, Post-Pos is equivalent to Base-Pos.

Post-Pos still introduces a positive bias, but the bias would be smaller for subset queries. However, Post-Pos may give inconsistent answers in the sense that the query result on $A \cup B$, where A and B are disjoint, may not equal the addition of the query results for A and B separately.

- **Base-Cut**: *After standard* FO*, convert everything below some sensitivity threshold to 0.*

The original design goal for frequency oracles is to recover frequencies for *frequent* values, and oftentimes there is a sensitivity threshold so that only estimations above the threshold are considered. Specifically, for each value, we compare its estimation with a threshold

$$T = F^{-1}\left(1 - \frac{\alpha}{d}\right)\sigma, \tag{7}$$

where d is the domain size, F^{-1} is the inverse of cummulative distribution function of the standard normal distribution, and σ is the standard deviation of the LDP mechanism (i.e., as in Equation (5)). By Base-Cut, estimations below the threshold are considered to be noise. When using such a threshold, for any value $v \in D$ whose original count is 0, the probability that it will have an estimated frequency above T (or the probability a zero-mean Gaussian variable with standard deviation δ is above T) is at most $\frac{\alpha}{d}$. Thus when we observe an estimated frequency above T, the probability that the true frequency of the value is 0 is (by union bound) at most $d \times \frac{\alpha}{d} = \alpha$. In [14], it is recommended to set $\alpha = 5\%$, following conventions in the statistical community.

Empirically we observe that $\alpha = 5\%$ performs poorly, because such a threshold can be too high when the population size is not very large and/or the ϵ is not large. A large threshold results in all except for a few estimations to be below the threshold and set to 0. We note that the choice of α is trading off false positives with false negatives. Given a large domain, there are likely between several and a few dozen values that have quite high frequencies, with most of the remaining values having low true counts. We want to keep an estimation if it is a lot more likely to be from a frequent value than from a very low frequency one. In this paper, we choose to set $\alpha = 2$, which ensures that the expected number of *false positives*, i.e., values with very low true frequencies but estimated frequencies above T, to be around 2. If there are

around 20 values that are truly frequent and have estimated frequencies above T, then ratio of true positives to false positives when using this threshold is 10:1.

This method ensures that all estimations are non-negative. It does not ensure that the sum of estimations is 1. The resulting estimations are either high (above the chosen threshold) or zero. The estimation for each item with non-zero frequency is subject to two bias effects. The negative bias effect is caused by the situation when the estimations are cut to zero. The positive effect is when large positive noise causes the estimation to be above the threshold, the resulting estimation is higher than true frequency.

B. Normalization Method

We now explore several methods that normalize the estimated frequencies of the whole domain to ensure that the sum of the estimates equals 1. When the estimations are normalized to sum to 1, the sum of the biases over the whole domain has to be 0.

Lemma 2. *If a normalization method adjusts the unbiased estimates so that they add up to* 1*, the sum of biases it introduces over the whole domain is* 0*.*

Proof. Denote f'_v as the estimated frequency of value v after post-processing. By linearity of expectations, we have

$$\sum_{v\in D}\left(\mathbb{E}\left[\,f'_v\,\right]-f_v\right)=\mathbb{E}\left[\sum_{v\in D}f'_v\right]-\sum_{v\in D}f_v=\mathbb{E}\left[\,1\,\right]-1=0$$

□

One standard way to do such normalization is through additive normalization:

- **Norm**: *After standard* FO*, add δ to each estimation so that the overall sum is 1.*

The method is formally proposed for the centralized setting [16] of DP and is used in the local setting, e.g., [28], [22]. Note the method does not enforce non-negativity. For GRR, Hadamard Response, and Subset Selection, this method actually does nothing, since each user reports a single value, and the estimations already sum to 1. For OLH, however, each user reports a randomly selected subset whose size is a random variable, and Norm would change the estimations. It can be proved that Norm is unbiased:

Lemma 3. *Norm provides unbiased estimation for each value.*

Proof. By the definition of Norm, we have $\sum_{v\in D}f'_v = \sum_{v\in D}(\tilde{f}_v+\delta)=1$. As the frequency oracle outputs unbiased estimation, i.e., $\mathbb{E}\left[\,\tilde{f}_v\,\right]=f_v$, we have

$$\begin{aligned}\mathbb{E}\left[\sum_{v\in D}f'_v\right]&=1=\mathbb{E}\left[\sum_{v\in D}(\tilde{f}_v+\delta)\right]\\&=\sum_{v\in D}\mathbb{E}\left[\,\tilde{f}_v\,\right]+d\cdot\mathbb{E}\left[\,\delta\,\right]=1+d\cdot\mathbb{E}\left[\,\delta\,\right]\\\Longrightarrow\ &\mathbb{E}\left[\,\delta\,\right]=0\end{aligned}$$

Thus $\mathbb{E}\left[\,f'_v\,\right]=\mathbb{E}\left[\,\tilde{f}_v+\delta\,\right]=\mathbb{E}\left[\,\tilde{f}_v\,\right]+0=f_v$. □

Besides sum-to-one, if a method also ensures non-negativity, we first state that it introduces positive bias to values whose frequencies are close to 0.

Lemma 4. *If a normalization method adjusts the unbiased estimates so that they add up to* 1 *and are non-negative, then it introduces positive biases to values that are sufficiently close to* 0*.*

Proof. As the estimates are non-negative and sum up to 1, some of the estimates must be positive. For a value close to 0, there exists some possibility that its estimation is positive; but the possibility its estimation is negative is 0. Thus the expectation of its estimation is positive, leading to a positive bias. □

Lemma 4 shows the biases for any method that ensures both constraints cannot be all zeros. Thus different methods are essentially different ways of distributing the biases. Next we present three such normalization methods.

- **Norm-Mul**: *After standard* FO*, convert negative value to 0. Then multiply each value by a multiplicative factor so that the sum is 1.*

More precisely, given estimation vector $\tilde{\mathbf{f}}$, we find γ such that

$$\sum_{v\in D}\max(\gamma\times\tilde{f}_v,0)=1,$$

and assign $f'_v=\max(\gamma\times\tilde{f}_v,0)$ as the estimations. This results in a consistent FO. Kairouz et al. [19] evaluated this method and it performs well when the underlying dataset distribution is smooth. This method results in positive biases for low-frequency items, but negative biases for high-frequency items. Moreover, the higher an item's true frequency, the larger the magnitude of the negative bias. The intuition is that here γ is typically in the range of $[0,1]$; and multiplying by a factor may result in the estimation of high frequency values to be significantly lower than their true values. When the distribution is skewed, which is more interesting in the LDP case, the method performs poorly.

- **Norm-Sub**: *After standard* FO*, convert negative values to 0, while maintaining overall sum of 1 by adding δ to each remaining value.*

More precisely, given estimation vector $\tilde{\mathbf{f}}$, we want to find δ such that

$$\sum_{v\in D}\max(\tilde{f}_v+\delta,0)=1$$

Then the estimation for each value v is $f'_v=\max(\tilde{f}_v+\delta,0)$. This extends the method Norm and results in consistency. Norm-Sub was used by Kairouz et al. [19] and Bassily [3] to process results for some FO's. Under Norm-Sub, low-frequency values have positive biases, and high-frequency items have negative biases. The distribution of biases, however, is more even when compared to Norm-Mul.

- **Norm-Cut**: *After standard* FO*, convert negative and small positive values to 0 so that the total sums up to 1.*

We note that under Norm-Sub, higher frequency items have higher negative biases. One natural idea to address this is to turn the low estimations to 0 to ensure consistency, without changing the estimations of high-frequency values. This is the idea of Norm-Cut. More precisely, given the estimation vector $\tilde{\mathbf{f}}$, there are two cases. When $\sum_{v\in D}\max(\tilde{f}_v,0)\le 1$, we simply change each negative estimations to 0. When $\sum_{v\in D}\max(\tilde{f}_v,0)>1$, we want to find the smallest θ such that

$$\sum_{v\in D|\tilde{f}_v\ge\theta}\tilde{f}_v\le 1$$

Then the estimation for each value v is 0 if $\tilde{f}_v<\theta$ and $\tilde{f}_v$ if $\tilde{f}_v\ge\theta$. This is similar to Base-cut in that both methods change all estimated values below some thresholds to 0. The differences lie in how the threshold is chosen. This results in non-negative estimations, and typically results in estimations that sum up to 1, but might result in a sum <1.

C. Constrained Least Squares

From a more principled point of view, we note that what we are doing here is essentially solving a Constraint Inference (CI) problem, for which CLS (Constrained Least Squares) is a natural solution. This approach was proposed in [16] but without the constraint that the estimates are non-negative (and it leads to Norm). Here we revisit this approach with the consistency constraint (i.e., both requirements in Definition 2).

- **CLS**: *After standard* FO*, use least squares with constraints (summing-to-one and non-negativity) to recover the values.*

Specifically, given the estimates $\tilde{\mathbf{f}}$ by FO, the method outputs $\mathbf{f}'$ that is a solution of the following problem:

$$\begin{aligned}\text{minimize: } &||\mathbf{f}'-\tilde{\mathbf{f}}||_2\\ \text{subject to: } &\forall_v f'_v\ge 0\\ &\sum_v f'_v=1\end{aligned}$$

We can use the KKT condition [21], [20] to solve the problem. The process is presented in Appendix A. In the solution, we partition the domain D into D_0 and D_1, where $D_0\cap D_1=\emptyset$ and $D_0\cup D_1=D$. For $v\in D_0$, assign $f'_v=0$. For $v\in D_1$,

$$f'_v=\tilde{f}_v-\frac{1}{|D_1|}\left(\sum_{v\in D_1}\tilde{f}_v-1\right)$$

Norm-Sub is the solution to the Constraint Least Square (CLS) formulation to the problem, and $\delta=-\frac{1}{|D_1|}\left(\sum_{v\in D_1}\tilde{f}_v-1\right)$ is the δ we want to find in Norm-Sub.

D. Maximum Likelihood Estimation

Another more principled way of looking into this problem is to view it as recovering distributions given some LDP reports. For this problem, one standard solution is Bayesian inference. In particular, we want to find the $\mathbf{f}'$ such that

$$\mathsf{Pr}\left[\mathbf{f}'|\tilde{\mathbf{f}}\right]=\frac{\mathsf{Pr}\left[\tilde{\mathbf{f}}|\mathbf{f}'\right]\cdot\mathsf{Pr}\left[\mathbf{f}'\right]}{\mathsf{Pr}\left[\tilde{\mathbf{f}}\right]}\tag{8}$$

is maximized. Note that we require $\mathbf{f}'$ satisfies $\forall_v f'_v\ge 0$ and $\sum_v f'_v=1$. In (8), $\mathsf{Pr}[\mathbf{f}']$ is the prior, and the prior distribution influence the result. In our setting, as we assume there is no such prior, $\mathsf{Pr}[\mathbf{f}']$ is uniform. That is, $\mathsf{Pr}[\mathbf{f}']$ is a constant. The denominator $\mathsf{Pr}\left[\tilde{\mathbf{f}}\right]$ is also a constant that does not influence the result. As a result, we are seeking for $\mathbf{f}'$ which is the maximal likelihood estimator (MLE), i.e., $\mathsf{Pr}\left[\tilde{\mathbf{f}}|\mathbf{f}'\right]$ is maximized.

For this method, Peter et al. [19] derived the exact MLE solution for GRR and RAPPOR [14]. We compute $\mathsf{Pr}\left[\tilde{\mathbf{f}}|\mathbf{f}'\right]$ using the general form of Equation (4), which states that, given the original distribution $\mathbf{f}'$, the vector $\tilde{\mathbf{f}}$ is a set of independent random variables, where each component $\tilde{f}_v$ follows Gaussian distribution with mean f'_v and variance σ'^2_v. The likelihood of $\tilde{\mathbf{f}}$ given $\mathbf{f}'$ is thus

$$\begin{aligned}&\mathsf{Pr}\left[\tilde{\mathbf{f}}|\mathbf{f}'\right]=\prod_v\mathsf{Pr}\left[\tilde{f}_v|f'_v\right]\\ \approx&\prod_v\frac{1}{\sqrt{2\pi\sigma'^2_v}}\cdot e^{-\frac{(f'_v-\tilde{f}_v)^2}{2\sigma'^2_v}}=\frac{1}{\sqrt{2\pi\prod_v\sigma'^2_v}}\cdot e^{-\sum_v\frac{(f'_v-\tilde{f}_v)^2}{2\sigma'^2_v}}.\end{aligned}\tag{9}$$

To differentiate from [19], we call it MLE-Apx.

- **MLE-Apx**: *First use standard* FO*, then compute the MLE with constraints (summing-to-one and non-negativity) to recover the values.*

In Appendix B, we use the KKT condition [21], [20] to obtain an efficient solution. In particular, we partition the domain D into D_0 and D_1, where $D_0\cap D_1=\emptyset$ and $D_0\cup D_1=D$. For $v\in D_0$, $f'_v=0$; for $v\in D_1$,

$$f'_v=\frac{q(1-q)x_v+\tilde{f}_v(p-q)}{p-q-(p-q)(1-p-q)x_v}\tag{10}$$

where

$$x_v=\frac{\sum_{x\in D_1}\tilde{f}_v(p-q)-(p-q)}{(p-q)(1-p-q)-|D_1|q(1-q)}$$

We can rewrite Equation (10) as

$$f'_v=\tilde{f}_v\cdot\gamma+\delta,$$

where

$$\begin{aligned}\gamma&=\frac{p-q}{p-q+(p-q)(1-p-q)x_v}\\ \delta&=\frac{q(1-q)x_v}{p-q+(p-q)(1-p-q)x_v}\end{aligned}$$

Hence MLE-Apx appears to represent some hybrid of Norm-Sub and Norm-Mul. In evaluation, we observe that Norm-Sub and MLE-Apx give very close results, as $\gamma\sim 1$. Furthermore,

Method	Description	Non-neg	Sum to 1	Complexity
Base-Pos	Convert negative est. to 0	Yes	No	$O(d)$
Post-Pos	Convert negative query result to 0	Yes	No	N/A
Base-Cut	Convert est. below threshold T to 0	Yes	No	$O(d)$
Norm	Add δ to est.	No	Yes	$O(d)$
Norm-Mul	Convert negative est. to 0, then multiply γ to positive est.	Yes	Yes	$O(d)$
Norm-Cut	Convert negative and small positive est. below θ to 0.	Yes	Almost	$O(d)$
Norm-Sub	Convert negative est. to 0 while adding δ to positive est.	Yes	Yes	$O(d)$
MLE-Apx	Convert negative est. to 0, then add δ to positive est.	Yes	Yes	$O(d)$
Power	Fit Power-Law dist., then minimize expected squared error	Yes	No	$O(\sqrt{n} \cdot d)$
PowerNS	Apply Norm-Sub after Power	Yes	Yes	$O(\sqrt{n} \cdot d)$

TABLE I
SUMMARY OF METHODS.

when the f_v component in variance is dominated by the other component (as in Equation (5)), the CLS formulation is equivalent to our MLE formulation.

E. Least Expected Square Error

Jia et al. [17] proposed a method in which one first assumes that the data follows some type of distribution (but the parameters are unknown), then uses the estimates to fit the parameters of the distribution, and finally updates the estimates that achieve expected least square.

- **Power**: *Fit a distribution, and then minimize the expected squared error.*

Formally, for each value v, the estimate $\tilde{f}_v$ given by FO is regarded as the addition of two parts: the true frequency f_v and noise following the normal distribution (as shown in Equation (6)). The method then finds f'_v that minimizes $\mathbb{E}\left[(f_v - f'_v)^2 | \tilde{f}_v\right]$. To solve this problem, the authors estimate the true distribution f_v from the estimates $\tilde{\mathbf{f}}$ (where $\tilde{\mathbf{f}}$ is the vector of the $\tilde{f}_v$'s).

In particular, it is assume in [17] that the distribution follows Power-Law or Gaussian. The distributions can be determined by one or two parameters, which can be fitted from the estimation $\tilde{\mathbf{f}}$. Given $\mathsf{Pr}[x]$ as the probability $f_v = x$ from the fitted distribution, and $\mathsf{Pr}[x \sim \mathcal{N}(0, \sigma)]$ as the pdf of x drawn from the Normal distribution with 0 mean and standard deviation σ (as in Equation (6)), one can then minimize the objective. Specifically, for each value $v \in D$, output

$$f'_v = \int_0^1 \frac{\mathsf{Pr}\left[(\tilde{f}_v - x) \sim \mathcal{N}(0, \sigma)\right] \cdot \mathsf{Pr}[x] \cdot x}{\int_0^1 \mathsf{Pr}\left[(\tilde{f}_v - y) \sim \mathcal{N}(0, \sigma)\right] \cdot \mathsf{Pr}[y]\, dy} dx. \quad (11)$$

We fit $\mathsf{Pr}[x]$ with the Power-Law distribution and call the method Power. Using this method requires knowledge and/or assumption of the distribution to be estimated. If there are too much noise, or the underlying distribution is different, forcing the observations to fit a distribution could lead to poor accuracy. Moreover, this method does not ensure the frequencies sum up to 1, as Equation (11) only considers the frequency of each value v independently. To make the result consistent, we use Norm-Sub to post-process results of Power, since Power is close to CLS, and Norm-Sub is the solution to CLS. We call it PowerNS.

- **PowerNS**: *First use standard* FO*, then use Power to recover the values, finally use Norm-Sub to further process the results.*

F. Summary of Methods

In summary, Norm-Sub is the solution to the Constraint Least Square (CLS) formulation to the problem. Furthermore, when the f_v component in variance is dominated by the other component (as in Equation (5)), the CLS formulation is equivalent to our MLE formulation. In that case, Norm-Sub is equivalent to MLE-Apx.

Table I gives a summary of the methods. First of all, all of the methods preserve the frequency order of the value, i.e., $f'_{v_1} \leq f'_{v_2}$ iff $\tilde{f}_{v_1} \leq \tilde{f}_{v_2}$. The methods can be classifies into three classes: First, enforcing non-negativity only. Base-Pos, Post-Pos, Base-Cut, and Power fall in this category. Second, enforcing summing-to-one only. Only Norm is in this class. Third, enforcing the two requirement simultaneously. Norm-Mul, Norm-Cut, Norm-Sub, and PowerNS satisfy both requirements.

V. EVALUATION

As we are optimizing multiple utility metrics together, it is hard to theoretically compare different methods. In this section, we run experiments to empirically evaluate these methods.

At the high level, our evaluations show that different methods perform differently in different settings, and to achieve the best utility, it may or may not be necessary to exploit all the consistency constraints. As a result, we conclude that for full-domain query, Base-Cut performs the best; for set-value query, PowerNS performs the best; and for high-frequency-value query, Norm performs the best.

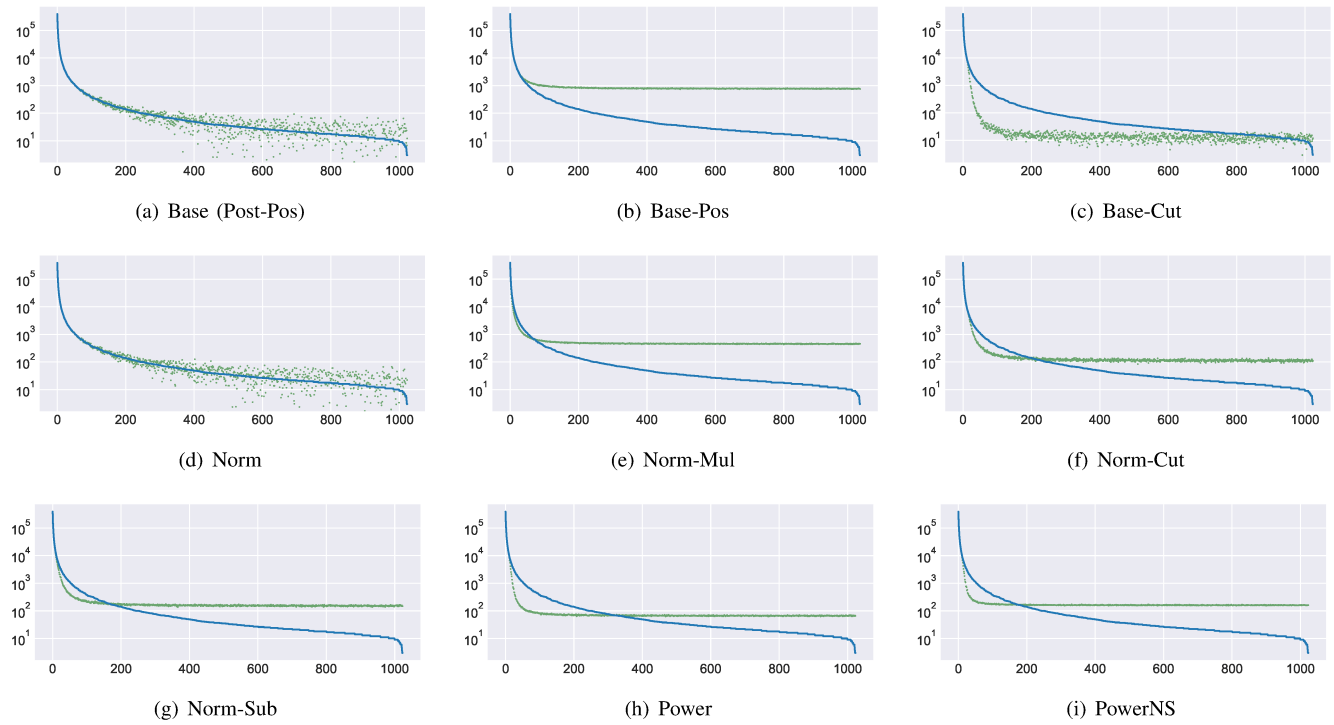
A. Experimental Setup

Datasets. We run experiments on two datasets (one synthetic and one real).

- Synthetic Zipf's distribution with 1024 values and 1 million reports. We use $s = 1.5$ in this distribution.
- Emoji: The daily emoji usage data. We use the average emoji usage of an emoji keyboard [1], which gives the total count of $n = 884427$ with $d = 1573$ different emojis.

Setup. The FO protocols and post-processing algorithms are implemented in Python 3.6.6 using Numpy 1.15; and all the experiments are conducted on a PC with Intel Core i7-4790 3.60GHz and 16GB memory. Although the post-processing methods can be applied to any FO protocol, we

[1] http://www.emojistats.org/, accessed 12/15/2019 10pm ET



(a) Base (Post-Pos) (b) Base-Pos (c) Base-Cut

(d) Norm (e) Norm-Mul (f) Norm-Cut

(g) Norm-Sub (h) Power (i) PowerNS

Fig. 1. Log-scale distribution of the Zipf's dataset fixing $\epsilon = 1$, the x-axes indicates the sorted value index and the y-axes is its count. The blue line is the ground truth; the green dots are estimations by different methods.

focus on simulating OLH as it provides near-optimal utility with reasonable communication bandwidth.

Metrics. We evaluate three scenarios 1) estimate the frequency of every value in the domain (full-domain), 2) estimate the aggregate frequencies of a subset of values (set-value), and 3) estimate the frequencies of the most frequent values (frequent-value).

We use the metrics of *Mean of Squared Error* (MSE). MSE measures the mean of squared difference between the estimate and the ground truth for each (set of) value. For full-domain, we compute

$$\text{MSE} = \frac{1}{d}\sum_{v\in D}(f_v - f'_v)^2.$$

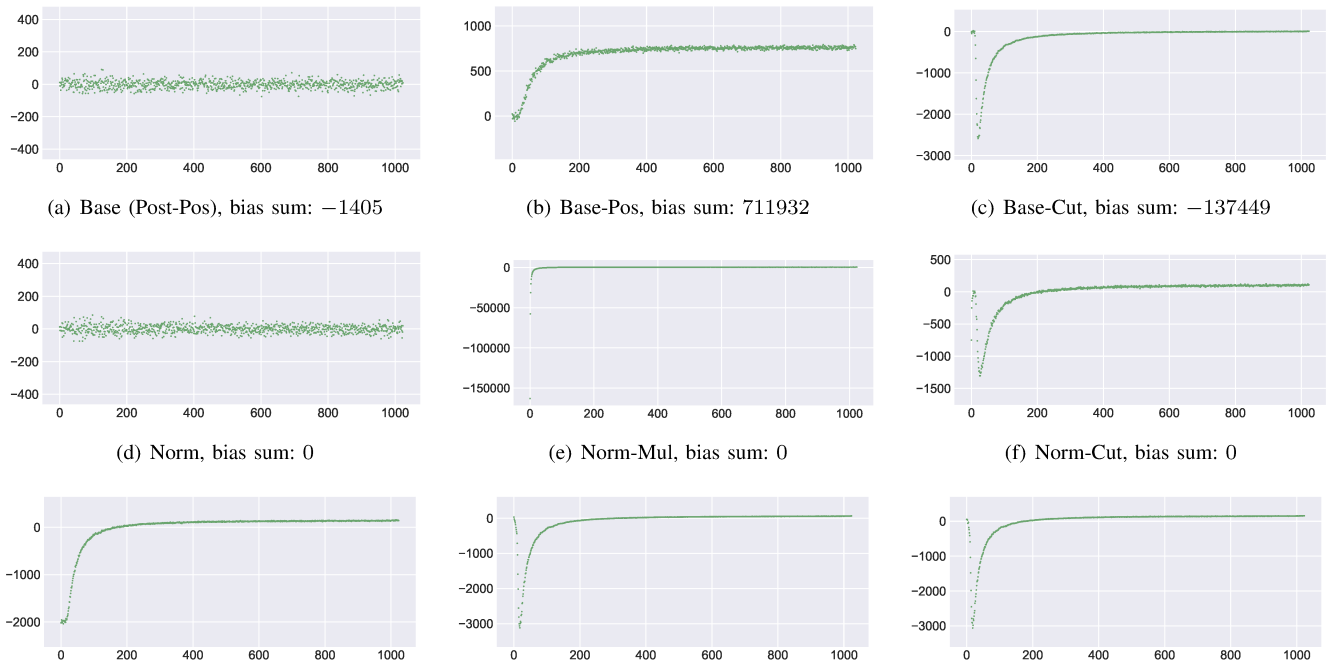
For frequent-value, we consider the top k values with highest f_v instead of the whole domain D; and for set-value, instead of measuring errors for singletons, we measure errors for sets, that is, we first sum the frequencies for a set of values, and then measure the difference.

Plotting Convention. Unless otherwise specified, for each dataset and each method, we repeat the experiment 30 times, with result mean and standard deviation reported. The standard deviation is typically very small, and barely noticeable in the figures.

Because there are 11 algorithms (10 post-processing methods plus Base), and for any single metric there are often multiple methods that perform very similarly, resulting their lines overlapping. To make Figures 4–8 readable, we plot results on two separate figures on the same row. On the left, we plot 6 methods, Base, Base-Pos, Post-Pos, Norm, Norm-Mul, and Norm-Sub. On the right, we plot Norm-Sub with the remaining 5 methods, MLE-Apx, Base-Cut, Norm-Cut, Power and PowerNS. We mainly want to compare the methods in the right column.

B. Bias-variance Evaluation

Figure 1 shows the true distribution of the synthetic Zipf's dataset and the mean of the estimations. As we plot the count estimations (instead of frequency estimations), the variance is larger (a $n^2 = 10^{12}$ multiplicative factor than the frequency estimations). We thus estimate 5000 times in order to make the mean stabilize. In Figure 2, we subtract the estimation mean by the ground truth and plot the difference, which representing the empirical bias. It can be seen that Base and Norm are unbiased. Base-Pos introduces systematic positive bias. Base-Cut gives unbiased estimations for the first few most frequent values, as their true frequencies are much greater than the threshold T used to cut off estimation below it to 0. As the noise is close to normal distribution, the possibility that a high-frequency value is estimated to be below T is exponentially small. The similar analysis also holds for the low-frequency values, whose estimates are unlikely to be above T. On the other hand, for values in between, the two biases compete with each other. At some point, the two effects cancel out with each other, leading to unbiased estimations. But this point is dependent on the whole distribution, and thus is hard to be found analytically. For Norm-Cut, the similar reasoning also



(a) Base (Post-Pos), bias sum: -1405 (b) Base-Pos, bias sum: 711932 (c) Base-Cut, bias sum: -137449

(d) Norm, bias sum: 0 (e) Norm-Mul, bias sum: 0 (f) Norm-Cut, bias sum: 0

(g) Norm-Sub, bias sum: 0 (h) Power, bias sum: -96332 (i) PowerNS, bias sum: 0

Fig. 2. Bias of count estimation for the Zipf's dataset fixing $\epsilon = 1$.

applies, with the difference that the threshold in Norm-Cut is typically smaller. For Norm-Sub, each value is influenced by two factors: subtraction by a same amount; and converting to 0 if negative. For the high-frequency values, we mostly see the first factor; for the low-frequency values, they are mostly affected by the second factor; and for the values in between, the two factors compete against each other. We see an increasing line for Norm-Sub. Finally, Power changes little to the top estimations; but more to the low ones, thus leading to a similar shape as Norm-Cut. The shape of PowerNS is close to Power because PowerNS applies Norm-Sub, which subtract some amount to the estimations, after Power.

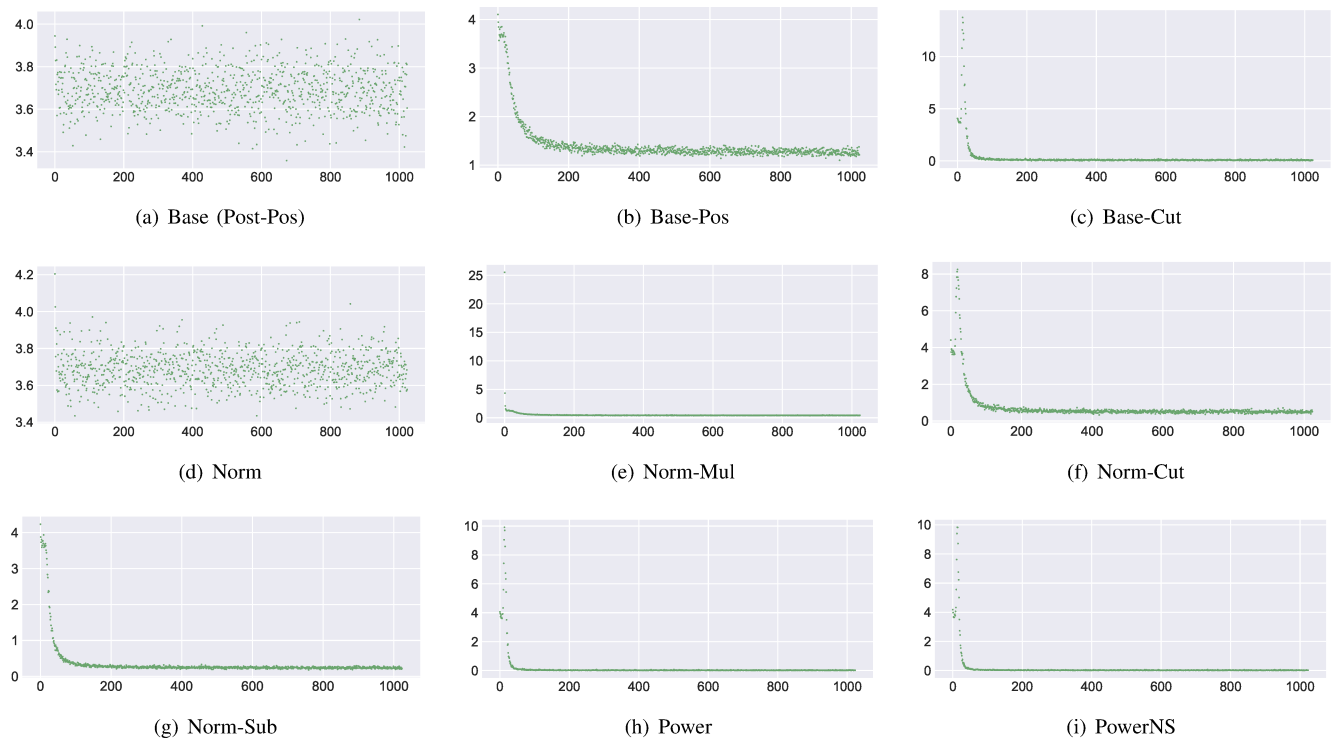
Figure 3 shows the variance of the estimations among the 5000 runs. First of all, the variance is similar for all the values in Base and Norm, with Norm being slightly better (smaller) than Base. For all other methods, the variance drops with the rank, because for low-frequency values, their estimates are mostly zeros.

C. Full-domain Evaluation

Figure 4 shows MSE when querying the frequency of every value in the domain. Note that The MSE is composed of the (square of) bias shown in Figure 2 and variance in Figure 3. We vary ϵ from 0.2 to 4. Let us fist focus on the figures on the left. Base performs very close to Norm, since the adjustment of Norm can be either positive or negative as the expected value of the estimation sum is 1. As Base-Pos (which is equivalent to Post-Pos in this setting) converts negative results to 0, its MSE is around half that of Base (note the y-axis is in log-scale). Norm-Sub is able to reduce the MSE of Base by about a factor of 10 and 100 in the Zipfs and Emoji dataset respectively. Norm-Mul behaves differently from other methods. In particular, the MSE decreases much slower than other methods. This is because Norm-Mul multiplies the original estimations by the same factor. The higher the estimate, the greater the adjustment. Since the estimations are individually unbiased, this is not the correct adjustment.

For the right part of Figure 4, we observe that, Norm-Sub and MLE-Apx perform almost exactly the same, validating the prediction from theoretical analysis. Norm-Sub, MLE-Apx, Power, PowerNS, and Base-Cut perform very similarly. In these two datasets, PowerNS performs the best. Note that PowerNS works well when the distribution is close to Power-Law. For an unknown distribution, we still recommend Base-Cut. This is because if one considers average accuracy of all estimations, the dominating source of errors comes from the fact many values have true frequencies close or equal to 0 are randomly perturbed. And Base-Cut maintains the high-frequency values unchanged, and converts results below a threshold T to 0. Norm-Cut also converts low estimations to 0, but the threshold θ is likely to be lower than T, because θ is chosen to achieve a sum of 1.

Benefit of Post-Processing. We demonstrate the benefit of post-processing by measuring the relationship between n and n', so that n records with post-processing can achieve the same accuracy for n' records without it. In particular, we vary n and measure the errors for different methods. We then calculate n' using Equation 3. In particular, the analytical MSE for n'



(a) Base (Post-Pos) (b) Base-Pos (c) Base-Cut

(d) Norm (e) Norm-Mul (f) Norm-Cut

(g) Norm-Sub (h) Power (i) PowerNS

Fig. 3. Variance of count estimation of the Zipf's dataset fixing $\epsilon = 1$. The y-axes are scaled down by $n = 10^6$ (a value a in the figure represents $a \cdot 10^6$).

records is

$$\frac{1}{d}\sum_v \sigma_v^2 = \frac{q(1-q)}{n'(p-q)^2} + \frac{1}{d}\sum_v \frac{f_v(1-p-q)}{n'(p-q)}$$
$$= \frac{q(1-q)}{n'(p-q)^2} + \frac{1}{d}\frac{1-p-q}{n'(p-q)}.$$

Given the empirical MSE, we can obtain n' that achieves the same error analytically. Note that the MSE does not depend on the distribution. Thus we only evaluate on the Zipf's dataset. The result is shown in Figure 5. We vary the size of the dataset n and plot the value of n' (note that the x-axes are in the scale of 10^6 and y-axes are 10^7). The higher the line, the better the method performs. Base and Norm are two straight lines with the slope of 1, verifying the analytical variance. The y value for Norm-Mul grows even slower than Base, indicating the harm of using Norm-Mul as a post-processing method. The performance of the other methods follow the similar trend of the full-domain MSE (as shown in the upper row of Figure 4), with PowerNS gives the best performance, which saves around 90% of users.

D. Set-value Evaluation

Estimating set-values plays an important role in the interactive data analysis setting (e.g., estimating which category of emoji's is more popular). Keeping $\epsilon = 1$, we evaluate the performance of different methods by changing the size of the set. For the set-value queries, we uniformly sample $\rho\% \times |D|$ elements from the domain and evaluate the MSE between the sum of their true frequencies and estimated frequencies. Formally, define $D_{s\rho}$ as the random subset of D that has $\rho\% \times |D|$ elements; and define $f_{D_{s\rho}} = \sum_{v \in D_{s\rho}} f_v$. We sample $D_{s\rho}$ multiple times and measure MSE between $f_{D_{s\rho}}$ and $f'_{D_{s\rho}}$. Overall, the error MSE of set-value queries is greater than that for the full-domain evaluation, because the error for individual estimation accumulates.

Vary ρ from 10 to 90. Following the layout convention, we show results for set-value estimations in Figure 6, where we first vary ρ from 10 to 90. Overall, the approaches that exploits the summing-to-1 requirement, including Norm, Norm-Mul, Norm-Sub, MLE-Apx, Norm-Cut, and PowerNS, perform well, especially when ρ is large. Moreover, their MSE is symmetric with $\rho = 50$. This is because as the results are normalized, estimating set-values for $\rho > 50$ equals estimating the rest. When $\rho = 90$, the best norm-based method, PowerNS, outperforms any of the non-norm based methods by at least 2 orders of magnitude.

For each specific method, it is observed the MSE for Base-Pos is higher than other methods, because it only turns negative estimates to 0, introducing systematic bias. Post-Pos is slightly better than Base, as it turns negative query results to 0. In the settings we evaluated, Base-Cut also outperforms Base; this happens because converting estimates below the threshold T to 0 is more likely to make the summation f'_D close to one. Finally, Power only converts negative estimations to be positive, introducing systematic bias; PowerNS further makes them sum to 1, thus achieving better utility than all

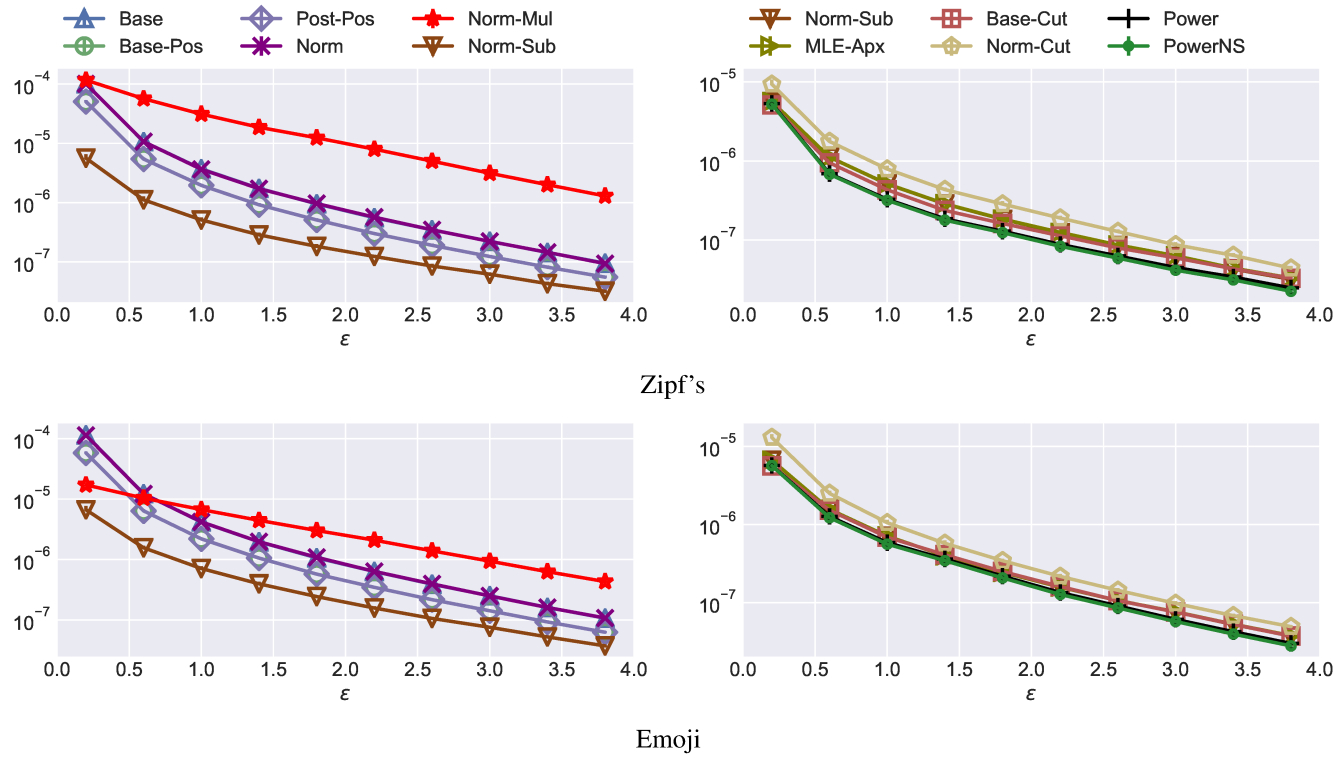


Fig. 4. MSE results on full-domain estimation, varying ϵ from 0.2 to 4.

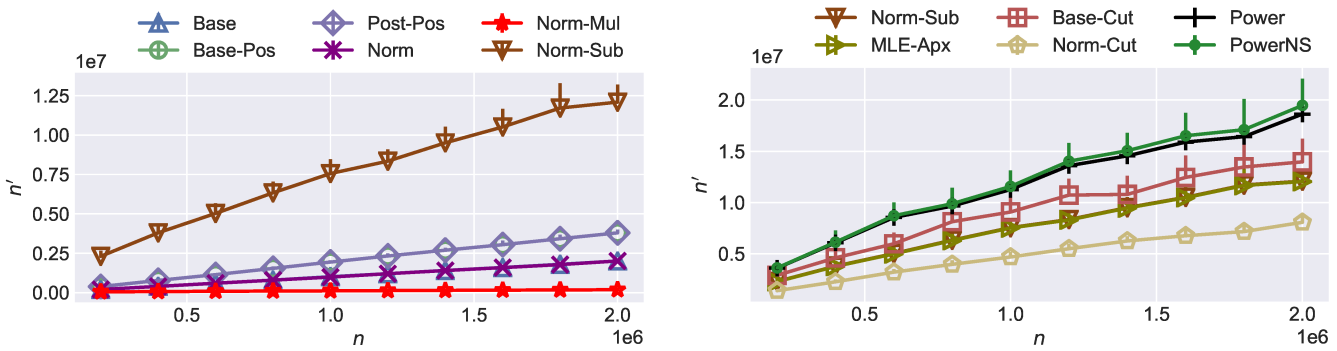


Fig. 5. MSE results on full-domain estimation on Zipfs dataset, comparing n with n', fixing $\epsilon = 1$ while varying n from 0.2×10^6 to 2.0×10^6. Three pairs of methods have similar performance: Base and Norm, Base-Pos and Post-Pos, Norm-Sub and MLE-Apx.

other methods.

Vary ρ from 1 to 10. Having examined the performance of set-queries for larger ρ, we then vary ρ from 1 to 10 and demonstrate the results in Figure 7. Within this ρ range, the errors of all methods increase with ρ, which is as expected. When ρ becomes small, the performance of different methods approaches to that of full-domain estimation.

Norm-Cut varies the threshold so that after cutting, the remaining estimates sum up to one. Thus the performance of Norm-Cut is better than Base-Cut especially when $\rho \geq 2$. Intuitively, the norm-based methods should perform better answering set-queries. But Norm-Mul does not. This is because the multiplication operation reduces the large estimates a lot, making them biased. This also demonstrates that enforcing sum-to-one is not enough. Different approaches perform significantly different.

Fixed set queries. Besides random set queries, we include a case study of fixed subset queries for the Emoji dataset. The queries ask the frequency of each category[2]. There are 68 categories with the mean of 10.4 items per set. The MSE varying ϵ is reported in Figure 8. It is interesting to see that the Post-Pos works best in the left sub-figure, and Norm-Cut from the right performs even better, especially when $\epsilon < 3$. This indicates the set-queries contain values that are infrequent.

[2]https://data.world/kgarrett/emojis

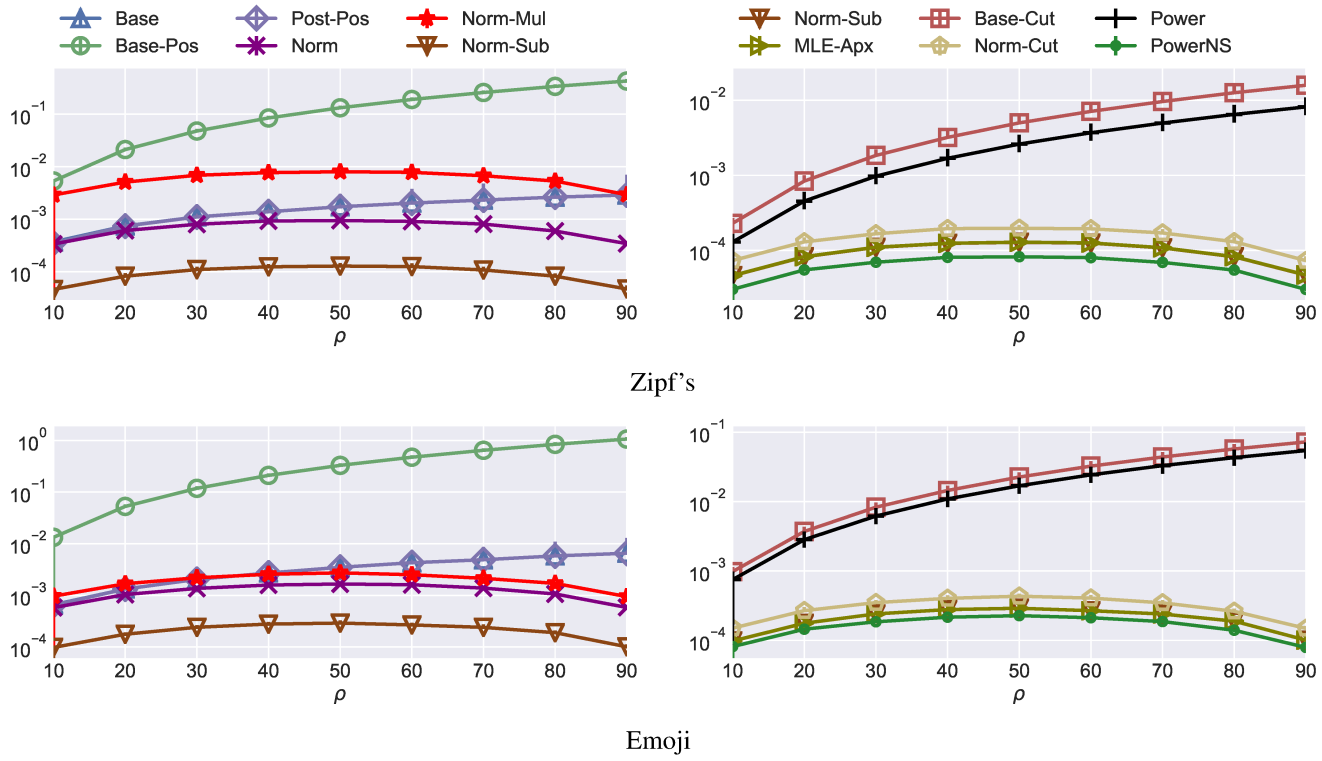


Fig. 6. MSE results on set-value estimation, varying set size percentage ρ from 10 to 90, fixing $\epsilon = 1$.

Choosing the method on synthetic dataset. As the optimal method in fixed set-values (as shown in Figure 8) is different from random set-values (shown in Figure 6 and 7), we investigate whether we can select the optimal post-processing method given the query and the LDP reports. In particular, we first fit a synthetic dataset from the estimation, then we simulate the data collection and estimation process multiple times, with different post-processing methods, and we calculate the errors taking the synthesized dataset as the ground truth. Figure 9 shows the result. Note that as we generate the synthetic dataset from the estimated distribution, the distribution itself should be consistent (non-negative and sum up to 1). We select Norm-Sub and PowerNS to process the estimated distribution first. These two methods perform well on full-domain and random set-value queries.

From the figure we can see that if the results are processed by Norm-Sub, the optimal method can be find quite accurately; if PowerNS is used, PowerNS will be selected. The reason is that PowerNS makes the distribution more close to the prior of Power-Law distribution, while Norm-Sub does not.

E. Frequent-value Evaluation

Finally, we evaluate different methods varying the top values to be considered. Define D_{tk} as $\{v \in D \mid f_v \text{ ranks top } k\}$. We measure MSE between $(f'_v)_{v \in D_{tk}}$ and $(f_v)_{v \in D_{tk}}$ for different values of k (from 2 to 32), fixing $\epsilon = 1$. Note that neither the frequency oracle nor the subsequent post-processing operation is aware of D_{tk}.

From the left column of Figure 10, we observe that Base, Base-Pos, Post-Pos, and Norm perform consistently well for different k, as the first three methods do nothing to the top values, and Norm touches them in an unbiased way. Norm-Mul performs at least $10\times$ worse than any other methods because it reduces the higher estimations a lot. Norm-Sub performs worse than Base, but better than Norm-Mul, because the same amount is subtracted from every estimate, regardless of k.

To give a better comparison, we plot both Base and Norm-Sub to the right (i.e., we ignore MLE-Apx for now, as it performs the same as Norm-Sub). These two methods have consistent MSE for different k. The rest four methods, Base-Cut, Norm-Cut, Power, and PowerNS, all have MSE that grows with k. In particular, for Base-Cut, a fixed threshold T (in Equation (7)) is used and estimates below it is converted to 0. This also suggests that at $\epsilon = 1$, around 10 values can be reliably estimated. This also happens to Norm-Cut for the similar reason. As Norm-Cut is better than Base-Cut, it suggests the threshold used in Norm-Cut is smaller than that in Base-Cut. If T is reduced, MSE of Base-Cut can be lowered until it matches that of Norm-Cut. Thus T is actually a tradeoff between frequent values and set-values. In practice, if the desired k is known in advance, one can set T to be the k-th highest estimated value. Finally, the performances of Power and PowerNS are similar, and they are worse than Base-Cut, especially when $k > 10$.

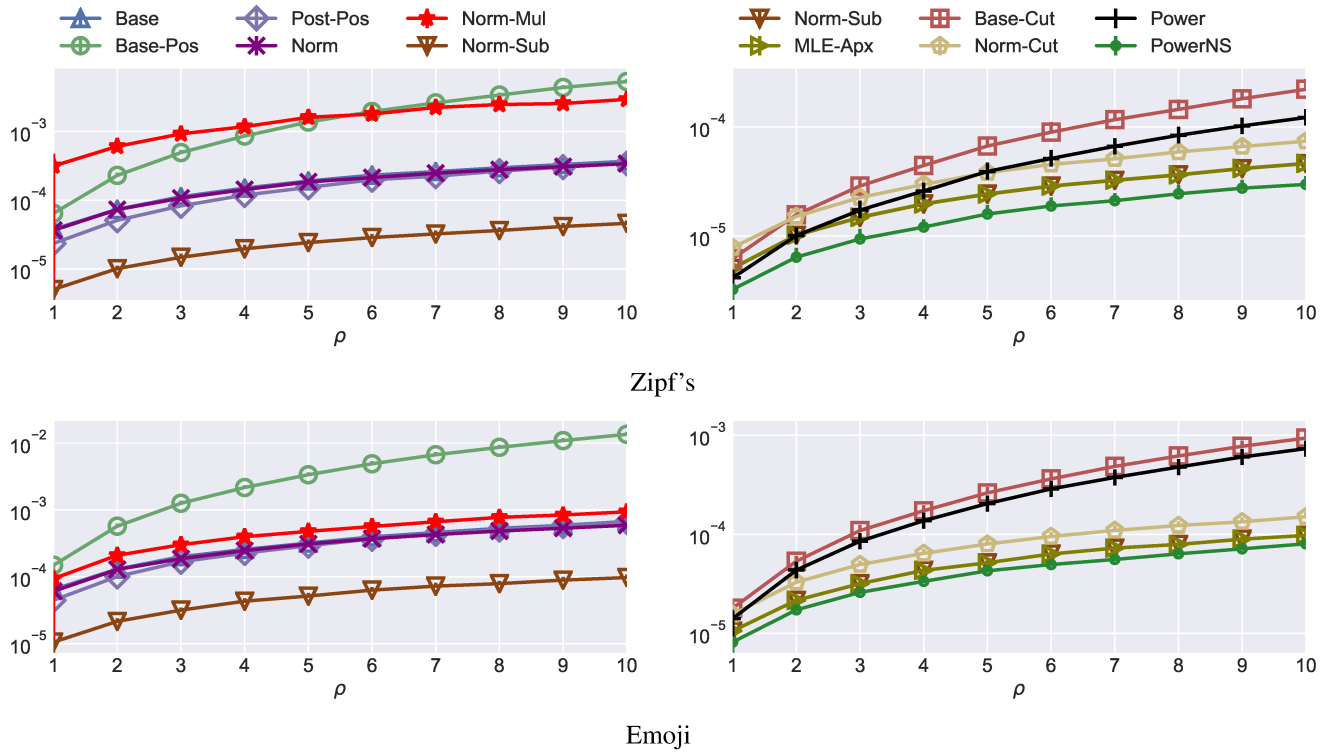


Fig. 7. MSE results on set-value estimation, varying set size percentage ρ from 1 to 10, fixing $\epsilon = 1$.

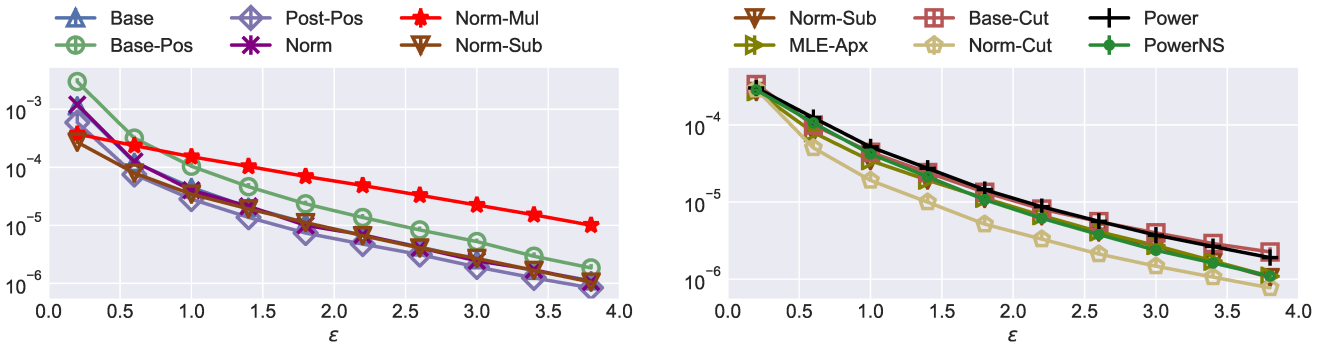


Fig. 8. MSE results on set-case estimation for the Emoji dataset, varying ϵ from 0.2 to 4.

F. Discussion

In summary, we evaluate the 10 post-processing methods on different datasets, for different tasks, and varying different parameters. We now summarize the findings and present guidelines for using the post-processing methods.

With the experiments, we verify the connections among the methods: Norm-Sub and MLE-Apx perform similarly, and Base and Norm performs similarly.

The best choice for post-processing method depends on the queries one wants to answer. If set-value estimation is needed, one should use PowerNS. When the set is fixed, one can also choose the optimal method using a synthetic dataset processed with Norm-Sub. The intuition is that PowerNS improves over the approximate MLE (i.e., Norm-Sub, which is a theoretically testified method) by making the estimates closer to the underlying distribution. If one just want to estimate results for the most frequent values, one can use Norm. While Base can also be used, Norm reduces variance by utilizing the property that the estimates sum up to 1. These two methods do not change any value dramatically. Finally, if one cares about single value queries only, Base-Cut should be used. This is because when many values in the dataset are of low frequency, converting low estimates to 0 benefit the utility. Overall, one can follow the guideline for choosing post-processing methods.

- When single value queries are desired, use Base-Cut.
- When frequent values are desired, use Norm.
- When set-value queries are desired, use PowerNS or

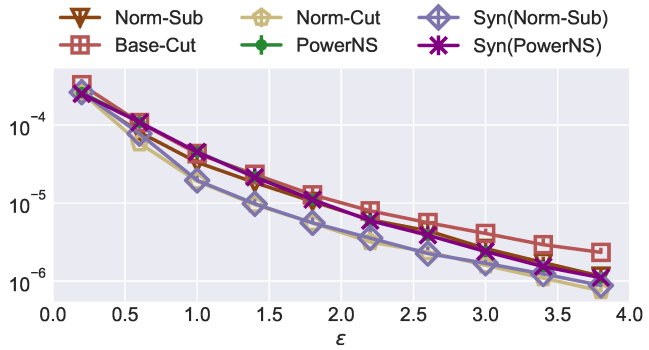


Fig. 9. Synthetic estimation for set-case query on the Emoji dataset.

select one using synthetic datasets.

VI. Related Work

LDP frequency oracle (estimating frequencies of values) is a fundamental primitive in LDP. There have been several mechanisms [14], [5], [31], [4], [2], [36] proposed for this task. Among them, [31] introduces OLH, which achieves low estimation errors and low communication costs on large domains. Hadamard Response [4], [2] is similar to OLH in essence, but uses the Hadamard transform instead of hash functions. The aggregation part is faster because evaluating a Hadamard entry is practically faster; but it only outputs a binary value, which gives higher error than OLH for larger ϵ setting. Subset selection [36], [30] achieves better accuracy than OLH, but with a much higher communication cost.

LDP frequency oracle is also a building block for other analytical tasks, e.g., finding heavy hitters [4], [7], [34], frequent itemset mining [26], [33], releasing marginals under LDP [27], [8], [38], key-value pair estimation [37], [15], evolving data monitoring [18], [13], and (multi-dimensional) range analytics [32], [22]. Mean estimation is also a building block in LDP; most of existing work transforms the numerical value to a discrete value using stochastic round, and then apply frequency oracles [11], [29], [24].

There exist efforts to post-process results in the setting of centralized DP. Most of them focus on utilizing the structural information in problems other than the simple histogram, e.g., estimating marginals [10], [25] and hierarchy structure [16]. The methods do not consider the non-negativity constraint. Other than that, they are similar to Norm-Sub and minimize L_2 distance. On the other hand, the authors of [23] started from MLE and propose a method to minimize L_1 instead of L_2 distance, as the DP noise follows Laplace distribution.

In the LDP setting, Kairouz et al. [19] study exact MLE for GRR and RAPPOR [14]; and empirically show exact MLE performs worse than Norm-Sub. In [3], Bassily proves the error bound of Norm-Sub for the Hadamard Response mechanism. Jia et al. [17] propose to use external information about the dataset's distribution (e.g., assume the underlying dataset follows Gaussian or Zipf's distribution). We note that such information may not always be available. On the other hand, we exploit the basic information in each LDP setting. That is, first, the total number of users is known; second, negative values are not possible. We found that in the LDP setting, on the contrary to [19], minimizing L_2 distance achieves MLE under the approximation that the noise is close to the Gaussian distribution. There are also post-processing techniques proposed for other settings: Blasiok et al. [6] study the post-processing for linear queries, which generalizes histogram estimation; but their method only applied to a non-optimal LDP mechanism. [28] and [22] consider the hierarchy structure and apply the technique of [16]. [37] considers mean estimation and propose to project the result into $[0, 1]$.

VII. Conclusion

In this paper, we study how to post-process results from existing frequency oracles to make them consistent while achieving high accuracy for a wide range of tasks, including frequencies of individual values, frequencies of the most frequent values, and frequencies of subsets of values. We considered 10 different methods, in addition to the baseline. We identified Norm performs similar to Base, and MLE-Apx performs similar to Norm-Sub. We then recommend that for full-domain estimation, Base-Cut should be used; when estimating frequency of the most frequent values, Norm should be used; when answering set-value queries, PowerNS or the optimal one from synthetic dataset should be used.

Acknowledgement

This project is supported by NSF grant 1640374, NWO grant 628.001.026, and NSF grant 1931443. We thank our shepherd Neil Gong and the anonymous reviewers for their helpful suggestions.

References

[1] Apple differential privacy team, learning with privacy at scale, 2017.
[2] J. Acharya, Z. Sun, and H. Zhang. Hadamard response: Estimating distributions privately, efficiently, and with little communication. In *AISTATS*, 2019.
[3] R. Bassily. Linear queries estimation with local differential privacy. In *AISTATS*, 2019.
[4] R. Bassily, K. Nissim, U. Stemmer, and A. G. Thakurta. Practical locally private heavy hitters. In *NIPS*, 2017.
[5] R. Bassily and A. D. Smith. Local, private, efficient protocols for succinct histograms. In *STOC*, 2015.
[6] J. Blasiok, M. Bun, A. Nikolov, and T. Steinke. Towards instance-optimal private query release. In *SODA*, 2019.
[7] M. Bun, J. Nelson, and U. Stemmer. Heavy hitters and the structure of local privacy. In *PODS*, 2018.
[8] G. Cormode, T. Kulkarni, and D. Srivastava. Marginal release under local differential privacy. In *SIGMOD*, 2018.
[9] B. Ding, J. Kulkarni, and S. Yekhanin. Collecting telemetry data privately. In *NIPS*, 2017.
[10] B. Ding, M. Winslett, J. Han, and Z. Li. Differentially private data cubes: optimizing noise sources and consistency. In *SIGMOD*, 2011.
[11] J. C. Duchi, M. I. Jordan, and M. J. Wainwright. Local privacy and statistical minimax rates. In *FOCS*, 2013.
[12] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, 2006.
[13] Ú. Erlingsson, V. Feldman, I. Mironov, A. Raghunathan, K. Talwar, and A. Thakurta. Amplification by shuffling: From local to central differential privacy via anonymity. In *SODA*, 2018.
[14] Ú. Erlingsson, V. Pihur, and A. Korolova. RAPPOR: randomized aggregatable privacy-preserving ordinal response. In *CCS*, 2014.

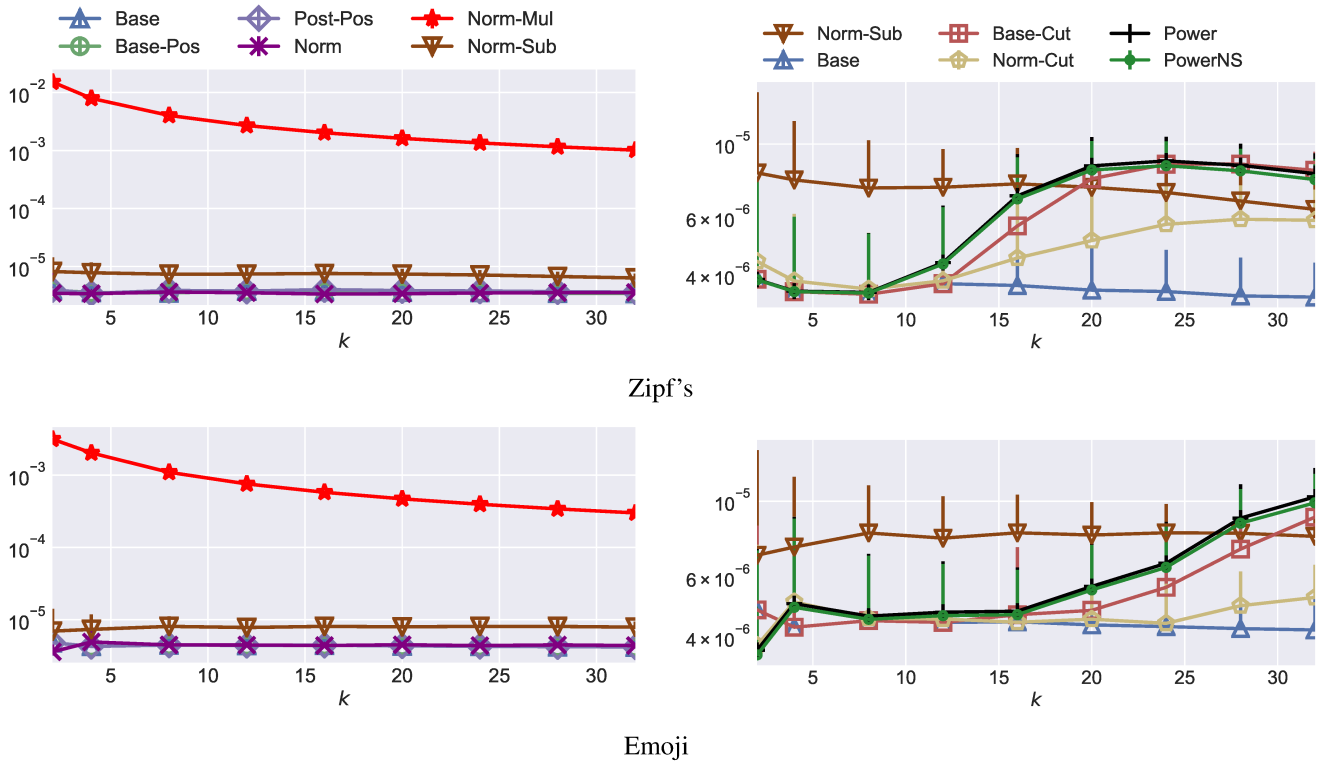


Fig. 10. MSE results on top-k value estimation varying k from 2 to 32, fixing $\epsilon = 1$.

[15] X. Gu, M. Li, Y. Cheng, L. Xiong, and Y. Cao. Pckv: Locally differentially private correlated key-value data collection with optimized utility. In *USENIX Security*, 2020.

[16] M. Hay, V. Rastogi, G. Miklau, and D. Suciu. Boosting the accuracy of differentially private histograms through consistency. *PVLDB*, 2010.

[17] J. Jia and N. Z. Gong. Calibrate: Frequency estimation and heavy hitter identification with local differential privacy via incorporating prior knowledge. In *INFOCOM*, 2019.

[18] M. Joseph, A. Roth, J. Ullman, and B. Waggoner. Local differential privacy for evolving data. In *NIPS*, 2018.

[19] P. Kairouz, K. Bonawitz, and D. Ramage. Discrete distribution estimation under local privacy. In *ICML*, 2016.

[20] W. Karush. Minima of functions of several variables with inequalities as side constraints. *M. Sc. Dissertation. Dept. of Mathematics, Univ. of Chicago*, 1939.

[21] H. W. Kuhn and A. W. Tucker. Nonlinear programming. In *Traces and emergence of nonlinear programming*. Springer, 2014.

[22] T. Kulkarni, G. Cormode, and D. Srivastava. Answering range queries under local differential privacy. *PVLDB*, 2019.

[23] J. Lee, Y. Wang, and D. Kifer. Maximum likelihood postprocessing for differential privacy under consistency constraints. In *KDD*, 2015.

[24] Z. Li, T. Wang, M. Lopuhaä-Zwakenberg, B. Skoric, and N. Li. Estimating numerical distributions under local differential privacy. *arXiv preprint arXiv:1912.01051*, 2019.

[25] W. Qardaji, W. Yang, and N. Li. Priview: practical differentially private release of marginal contingency tables. In *SIGMOD*, 2014.

[26] Z. Qin, Y. Yang, T. Yu, I. Khalil, X. Xiao, and K. Ren. Heavy hitter estimation over set-valued data with local differential privacy. In *CCS*, 2016.

[27] X. Ren, C.-M. Yu, W. Yu, S. Yang, X. Yang, J. A. McCann, and S. Y. Philip. Lopub: High-dimensional crowdsourced data publication with local differential privacy. *Trans. on Info. Forensics and Security*, 2018.

[28] N. Wang, X. Xiao, Y. Yang, T. D. Hoang, H. Shin, J. Shin, and G. Yu. Privtrie: Effective frequent term discovery under local differential privacy. In *ICDE*, 2018.

[29] N. Wang, X. Xiao, Y. Yang, J. Zhao, S. C. Hui, H. Shin, J. Shin, and G. Yu. Collecting and analyzing multidimensional data with local differential privacy. In *ICDE*, 2019.

[30] S. Wang, L. Huang, P. Wang, Y. Nie, H. Xu, W. Yang, X. Li, and C. Qiao. Mutual information optimally local private discrete distribution estimation. *CoRR*, abs/1607.08025, 2016.

[31] T. Wang, J. Blocki, N. Li, and S. Jha. Locally differentially private protocols for frequency estimation. In *USENIX Security*, 2017.

[32] T. Wang, B. Ding, J. Zhou, C. Hong, Z. Huang, N. Li, and S. Jha. Answering multi-dimensional analytical queries under local differential privacy. In *SIGMOD*. ACM, 2019.

[33] T. Wang, N. Li, and S. Jha. Locally differentially private frequent itemset mining. In *SP*, 2018.

[34] T. Wang, N. Li, and S. Jha. Locally differentially private heavy hitter identification. *Trans. Dependable Sec. Comput.*, 2019.

[35] S. L. Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 1965.

[36] M. Ye and A. Barg. Optimal schemes for discrete distribution estimation under locally differential privacy. *Transactions on Information Theory*, 2018.

[37] Q. Ye, H. Hu, X. Meng, and H. Zheng. Privkv: Key-value data collection with local differential privacy. In *SP*, 2019.

[38] Z. Zhang, T. Wang, N. Li, S. He, and J. Chen. Calm: Consistent adaptive local marginal for marginal release under local differential privacy. In *CCS*, 2018.

APPENDIX A
SOLUTION FOR CLS

Using the KKT condition [21], [20], we augment the optimization target with the following equations:

$$\begin{aligned}\text{minimize} \quad & \sum_v (f'_v - \tilde{f}_v)^2 + a + b \\ \text{where} \quad & \sum_v f'_v = 1, \ \ \forall v : 0 \le f'_v \le 1, \\ & a = \mu \cdot \sum_v f'_v, b = \sum_v \lambda_v \cdot f'_v, \forall v : \lambda_v \cdot f'_v = 0.\end{aligned}$$

Since $b = 0$, and $a = \mu$ is a constant, the condition that minimizing the target is unchanged. Given that the target is convex, we can find the minimum by taking the partial derivative with respect to each variable:

$$\begin{aligned}&\frac{\partial \left[\sum_v (f'_v - \tilde{f}_v)^2 + a + b\right]}{\partial f'_v} = 0 \\ \implies & 2(f'_v - \tilde{f}_v) + \mu + \lambda_v = 0 \\ \implies & f'_v = \tilde{f}_v - \frac{1}{2}(\mu + \lambda_v)\end{aligned}$$

Now suppose there is a subset of domain $D_0 \subseteq D$ s.t., $\forall v \in D_0, f'_v = 0$ and $\forall v \in D_1 = D \setminus D_0, f'_v > 0 \wedge \lambda_v = 0$. By summing up f'_v for all $v \in D_1$, we have

$$1 = \sum_{v \in D_1} \tilde{f}_v - \frac{|D_1|\mu}{2}$$

Thus for all $v \in D_1$, we can use the formula

$$f'_v = \tilde{f}_v - \frac{1}{|D_1|}\left(\sum_{v \in D_1} \tilde{f}_v - 1\right)$$

to derive the estimate f'_v for value $v \in D_1$, and $f'_v = 0$ for $v \in D_0$. One can also find D_0 using a similar approach when dealing with MLE. And it can also be verified $\sum_v f'_v = 1$.

APPENDIX B
SOLUTION FOR MLE-APX

From Equation (9), we first simplify the exponent plugging in the value of σ'_v as in Equation (3):

$$\sum_v \frac{(f'_v - \tilde{f}_v)^2}{2\sigma'^2_v} = \frac{n}{2} \sum_v \frac{(f'_v - \tilde{f}_v)^2 (p-q)^2}{q(1-q) + f'_v(p-q)(1-p-q)}$$

The factor $\frac{n}{2}$ in the exponent ensures that for large n the exponent will vary the most with $\mathbf{f}'$, which dominates the coefficient $\frac{1}{\sqrt{2\pi \prod_v \sigma'^2_v}}$. Thus approximately we find $\mathbf{f}'$ that achieves the following optimization goal:

$$\begin{aligned}\text{minimize:} \ & \sum_v \frac{(f'_v - \tilde{f}_v)^2 (p-q)^2}{q(1-q) + f'_v(p-q)(1-p-q)} \\ \text{subject to:} \ & \sum_v f'_v = 1, \\ & \forall v, 0 \le f'_v \le 1.\end{aligned}$$

Using the KKT condition [21], [20], we augment the optimization target with the following equations:

$$\begin{aligned}\text{minimize} \quad & \sum_v \frac{(f'_v - \tilde{f}_v)^2 (p-q)^2}{q(1-q) + f'_v(p-q)(1-p-q)} + a + b \\ \text{where} \quad & \sum_v f'_v = 1, \ \ \forall v : 0 \le f'_v \le 1, \\ & a = \mu \cdot \sum_v f'_v, b = \sum_v \lambda_v \cdot f'_v, \forall v : \lambda_v \cdot f'_v = 0.\end{aligned}$$

Since $b = 0$, and $a = \mu$ is a constant, the condition for minimizing the target is unchanged. Given that the target is convex, we can find the minimum by taking the partial derivative with respect to each variable:

$$\begin{aligned}&\frac{\partial \left[\sum_v \frac{(f'_v - \tilde{f}_v)^2 (p-q)^2}{q(1-q) + f'_v(p-q)(1-p-q)} + a + b\right]}{\partial f'_v} \\ =& \frac{-(f'_v - \tilde{f}_v)^2 (p-q)^2 \cdot (p-q)(1-p-q)}{(q(1-q) + f'_v(p-q)(1-p-q))^2} \\ &+ \frac{2(f'_v - \tilde{f}_v)(p-q)^2}{q(1-q) + f'_v(p-q)(1-p-q)} + \mu + \lambda_v = 0\end{aligned}$$

Define a temporary notation

$$\begin{aligned}x_v =& \frac{(f'_v - \tilde{f}_v)(p-q)}{q(1-q) + f'_v(p-q)(1-p-q)} \\ \text{so that } f'_v =& \frac{q(1-q)x_v + \tilde{f}_v(p-q)}{p-q-(p-q)(1-p-q)x_v} \qquad (12)\end{aligned}$$

With x_v, we can simplify the previous equation:

$$(p-q)(1-p-q)x_v^2 - 2(p-q)x_v - \mu - \lambda_v = 0 \qquad (13)$$

Now suppose there is a subset of domain $D_0 \subseteq D$ s.t., $\forall v \in D_0, f'_v = 0$ and $\forall v \in D_1 = D \setminus D_0, f'_v > 0$ and $\lambda_v = 0$. Thus for those $v \in D_1$, solution of x_v in Equation (13) does not depend on v. We solve x_v by summing up f'_v for all $v \in D_1$:

$$\begin{aligned}\sum_{v \in D_1} f'_v =& 1 = \sum_{v \in D_1} \frac{q(1-q)x_v + \tilde{f}_v(p-q)}{p-q-(p-q)(1-p-q)x_v} \\ =& \frac{|D_1|q(1-q)x_v + \sum_{v \in D_1} \tilde{f}_v(p-q)}{p-q+(p-q)(1-p-q)x_v} \\ \implies x_v =& \frac{\sum_{x \in D_1} \tilde{f}_v(p-q) - (p-q)}{(p-q)(1-p-q) - |D_1|q(1-q)}\end{aligned}$$

Given x_v, we can compute f'_v from Equation (12) for each value $v \in D_1$ efficiently; and $f'_v = 0$ for $v \in D_0$. It can be verified $\sum_v f'_v = 1$.

Finally, to find D_0, one initiates $D_0 = \emptyset$ and $D_1 = D$, and iteratively tests whether all values in D_1 are positive. In each iteration, for any negative a_x, x is moved from D_1 to D_0. The process terminates when no negative a_x is found for all $x \in D_1$.