Instance Optimal Learning of Discrete Distributions*

Gregory Valiant†
Stanford University
Stanford, CA, USA
valiant@stanford.edu

Paul Valiant‡
Brown University
Providence, RI, USA
pvaliant@gmail.com

ABSTRACT

We consider the following basic learning task: given independent draws from an unknown distribution over a discrete support, output an approximation of the distribution that is as accurate as possible in ℓ_1 distance (equivalently, total variation distance, or "statistical distance"). Perhaps surprisingly, it is often possible to "de-noise" the empirical distribution of the samples to return an approximation of the true distribution that is significantly more accurate than the empirical distribution, *without relying on any prior assumptions on the distribution.* We present an *instance optimal* learning algorithm which optimally performs this de-noising for every distribution for which such a de-noising is possible. More formally, given n independent draws from a distribution p, our algorithm returns a labelled vector whose expected distance from p is equal to the minimum possible expected error that could be obtained by any algorithm, even one that is given the true unlabeled vector of probabilities of distribution p and simply needs to assign labels—up to an additive subconstant term that is independent of p and goes to zero as n gets large. This somewhat surprising result has several conceptual implications, including the fact that, for any large sample from a distribution over discrete support, prior knowledge of the rates of decay of the tails of the distribution (e.g. power-law type assumptions) is not significantly helpful for the task of learning the distribution.

As a consequence of our techniques, we also show that given a set of n samples from an arbitrary distribution, one can accurately estimate the expected number of distinct elements that will be observed in a sample of any size up to $n \log n$. This sort of extrapolation is practically relevant, particularly to domains such as genomics where it is important to understand how much more might be discovered given larger sample sizes, and we are optimistic that our approach is practically viable.

Categories and Subject Descriptors

G.3 [**Probability and Statistics**]: Nonparametric statistics

Keywords

Distribution learning, property estimation, Good-Turing frequency estimation, instance optimality, the unseen species problem

1. INTRODUCTION

Given independent draws from an unknown distribution over an unknown discrete support, what is the best way to aggregate those samples into an approximation of the true distribution? The most obvious and most widely employed approach is to simply output the empirical distribution of the sample. To what extent can one improve over this naive approach? To what extent can one "de-noise" the empirical distribution, without relying on any assumptions on the structure of the underlying distribution?

Perhaps surprisingly, there are many settings in which denoising can be done without a priori assumptions on the distribution. We begin by presenting two motivating examples illustrating rather different settings in which significant de-noising of the empirical distribution is possible.

EXAMPLE 1. *Suppose you are given 100,000 independent draws from some unknown distribution, and you find that there are roughly 1,000 distinct elements, each of which appears roughly* 100 ± 10 *times. Furthermore, suppose you compute the variance in the number of times the different domain elements occur, and it is close to 100. Based on these samples, you can confidently deduce that the true distribution is very close to a uniform distribution over 1,000 domain elements, and that the true probability of a domain element seen 90 times is roughly the same as that of an element observed 110 times. The basic reasoning is as follows: if the true distribution were the uniform distribution, then the noise from the random sampling would exhibit the observed variance in the number of occurrences; if there was any significant variation in the true probabilities of the different domain elements, then, combined with the noise added via the random sampling, the observed variance would be noticeably larger than 100. The* ℓ_1 *error of the empirical*

†This work is supported in part by NSF CAREER Award CCF-1351108.

‡This work is supported in part by a Sloan Research Fellowship.

*A full version of this paper is available at `http://arxiv.org/abs/1504.05321`

STOC'16, June 19–21, 2016, Cambridge, MA, USA
© 2016 ACM. 978-1-4503-4132-5/16/06...$15.00
http://dx.doi.org/10.1145/2897518.2897641

distribution would be roughly 0.1*, whereas the "de-noised" distribution would have error less than* 0.01.

EXAMPLE 2. *Suppose you are given 1,000 independent draws from an unknown distribution, and all 1000 samples are unique domain elements. You can safely conclude that the combined probability of all the observed domain elements is likely to be much less than 1/100—if this were not the case, one would expect at least one of the observed elements to occur twice in the sample. Hence the empirical distribution of the samples is likely to have ℓ_1 distance nearly* 2 *from the true distribution, whereas this reasoning would suggest that one should ascribe a total probability mass of at most* $1/100$ *to the observed domain elements.*

In both of the above examples, the key to the "de-noising" was the realization that the true distributions possessed some structure—structure that was both easily deduced from the samples, and structure that, once known, could then be leveraged to de-noise the empirical distribution. Our main result is an algorithm which de-noises the empirical distribution as much as is possible, whenever such denoising is possible. Specifically, our algorithm achieves, up to a subconstant term that vanishes as the sample size increases, the best error that any algorithm could achieve—even an algorithm that is given the unlabeled vector of true probabilities and simply needs to correctly label the probabilities. It is in this sense that we mean our algorithm is instance-optimal: if there is any instance-specific algorithm that can take advantage of the structure of the true probabilities to better label the domain elements, then our general algorithm will essentially match the performance of this specialized algorithm on this instance, even *without* knowing the true probabilities ahead of time.

THEOREM 1. *There is a function $err(n)$ that goes to zero as n gets large, and an algorithm, which given n independent draws from any distribution p of discrete support, outputs a labelled vector q, such that*

$$E[||p - q||_1] \leq opt(p, n) + err(n),$$

where $opt(p, n)$ is the minimum expected ℓ_1 error that any algorithm could achieve on the following learning task: given p, and given n samples drawn independently from a distribution that is identical to p up to an arbitrary relabeling of the domain elements, learn the distribution.

The performance guarantees of the above algorithm can be equivalently stated as follows: let $S \underset{n}{\leftarrow} p$ denote that S is a set of n independent draws from distribution p, and let $\pi(p)$ denote a distribution that is identical to p, up to relabeling the domain elements with arbitrary distinct new labels given by the mapping π. Our algorithm, which maps a set of samples S to a labelled vector $q = f(S)$, satisfies the following: For any distribution p,

$$\mathop{\mathrm{E}}_{S \underset{n}{\leftarrow} p}[||p - q||_1] \leq \min_{\text{algs } \mathcal{A}} \max_{\pi} \left(\mathop{\mathrm{E}}_{S \underset{n}{\leftarrow} \pi(p)} [\pi(p) - \mathcal{A}(S)] \right) + o_n(1),$$

where $o_n(1) \to 0$ as $n \to \infty$ is independent of p and depends only on n.

A worst-case bound on the magnitude of the error term in Theorem 1 is $err(n) = 1/\log^{\theta(1)} n$, and no dependence better than $1/\log n$ is possible for worst-case combinations of p and n. For any fixed distribution p, the error decays much faster in the asymptotic regime in which n is large in comparison to the support size of p, as $err(n) = O_p(1/\sqrt{n})$. The critical regime in which our theorem yields the most surprising results is where $opt(p, n)$ is somewhat larger than $err(n) = 1/\log^{\theta(1)} n$, namely, for a distribution p and sample size n for which the labeling problem incurs non-negligible error—for example when p is a distribution with large support and n is a constant multiple of the support size.

One surprising implication of the above result is that, for large samples, prior knowledge of the "shape" of the distribution, or knowledge of the rate of decay of the tails of the distribution, cannot significantly improve the accuracy of the learning task. For example, typical Bayesian assumptions that the frequency of words in natural language satisfy Zipf distributions, or the frequencies of different species of bacteria in the human gut satisfy Gamma distributions or various power-law distributions, etc, can improve the expected error of the learned distribution by at most a vanishing function of the sample size.

The key intuition behind this optimal de-noising, and the core of our algorithm, is the ability to very accurately approximate the *unlabeled* vector of probabilities of the true distribution, given access to independent samples. In some sense, our result can be interpreted as the following statement: up to an additive subconstant factor, one can always recover an approximation of the unlabeled vector of probabilities more accurately than one can disambiguate and label such a vector. That is, if one has enough samples to accurately label the unlabeled vector of probabilities, then one also has more than enough samples to accurately learn that unlabeled vector. Of course, this statement can only hold up to some additive error term, as the following example illustrates.

EXAMPLE 3. *Given samples drawn from a distribution supported on two unknown domain elements, if one is told that both probabilities are exactly* $1/2$*, then as soon as one observes both domain elements, one knows the distribution exactly, and thus the expected error given n samples will be $opt(p, n) = O(1/2^n)$ as this bounds the probability that one of the two domain elements is not observed in a set of n samples. Without the prior knowledge that the two probabilities are* $1/2$*, the best algorithm will have expected error $\approx 1/\sqrt{n}$. (In general, given n samples drawn from a uniform distribution over k elements, $opt(Unif[k], n) = Bin(n, 1/k, 0) \approx e^{-n/k}$, namely the expected probability mass consisting of unseen domain elements.)*

The above example illustrates that prior knowledge of the vector of probabilities can be helpful. Our result, however, shows that this phenomenon only occurs to a significant extent for very small sample sizes; for larger samples, no distribution exists for which prior knowledge of the vector of probabilities improves the expected error of a learning algorithm beyond a universal subconstant additive term that goes to zero as a function of the sample size.

1.1 Our Approach

Our algorithm proceeds via two steps. In the first step, the samples are used to output an approximation of the vector of

true probabilities. We show that, with high probability over the randomness of the n independent draws from the distribution, we accurately recover the portion of the vector of true probabilities consisting of values asymptotically larger than $1/(n \log n)$. This is a strengthened version of the results of [32, 34]. Note that the empirical distribution accurately estimates probabilities down to $\approx 1/n$—indeed the vector of empirical probabilities are all multiples of $1/n$. The characterization of the first phase of our algorithm can be interpreted as showing that we recover the vector of probabilities essentially to the accuracy that the empirical distribution would have if it were based on $n \log n$ samples, rather than only n samples. Of course, this surprisingly accurate reconstruction comes with the caveat that we are only recovering the unlabeled vector of probabilities—we do not know which domain elements correspond to the various probabilities.

The second step of our algorithm leverages the accurate approximation of the unlabeled vector of probabilities to optimally assign probability values to each of the observed domain elements. For some intuition into this step, first suppose you know the exact vector of unlabelled probabilities. Consider the following optimization problem: given n independent draws from distribution p, and an unlabeled vector v representing the true vector of probabilities of distribution p, for each observed domain element x, assign the probability $q(x)$ that minimizes the expected ℓ_1 distance $|q(x) - p(x)|$. As the following example illustrates, this problem is more subtle than it might initially seem; intuitive schemes such as assigning the ith largest probability in v to the element with the ith largest empirical probability is *not* optimal.

EXAMPLE 4. *Consider n independent draws from a distribution in which 90% of the domain elements occur with probability $10/n$, and the remaining 10% occur with probability $11/n$. If one assigns probability $11/n$ to the 10% of the domain elements with largest empirical frequencies, the ℓ_1 distance will be roughly 0.2, because the vast majority of the elements with the largest empirical frequencies will actually have true probability $10/n$ rather than $11/n$. In contrast, if one ignores the samples and simply assigns probability $10/n$ to all the domain elements, the ℓ_1 error will be exactly 0.1.*

This optimization task of assigning probabilities $q(x)$ to each observed domain element x (as a function of the unlabelled vector of true probabilities v, and set of n samples) so as to minimize the expected ℓ_1 error is a well-defined optimization problem. Nevertheless, this task seems to be computationally intractable. Part of the computational challenge is that the optimal probability to assign to a domain element x might be a function of 1) the true probabilities v, 2) the number of occurrences of x in the sample, and 3) the number of occurrences of all the other domain elements. Nevertheless, we describe a very natural and computationally efficient scheme which assigns a probability $q(x)$ to each x that is a function of only v and the number of occurrences of x; we show that this scheme incurs an expected error within $o(1)$ of the expected error of the optimal scheme (which assigns $q(x)$ as a function of v and the entire set of samples). Of course, there is the additional complication that, in the context of our two step algorithm, we do not actually have the exact vector of true probabilities—only an approximation of such a vector—and hence this second phase of our algorithm must be robust to some noise in the recovered probabilities.

Beyond yielding a near optimal learning algorithm, there are several additional benefits to our approach of first accurately reconstructing the unlabeled vector of probabilities. For instance, such an unlabeled vector allows us to estimate properties of the underlying distribution including estimating the error of our returned vector, and estimating the error in our estimate of *each* observed domain element's probability. Additionally, as the following proposition quantifies, this unlabeled vector of probabilities can be leveraged to produce an accurate estimate of the expected number of distinct elements that will be observed in sample sizes up to a logarithmic factor larger:

PROPOSITION 1. *Given n samples from an arbitrary distribution p, with probability $1 - e^{-n^{\Omega(1)}}$ over the randomness of the samples, one can estimate the expected number of unique elements that would be seen in a set of k samples drawn from p, to within error $k \cdot c\sqrt{\frac{k}{n \log n}}$ for some universal constant c.*

This proposition is tight, and it is slightly surprising in that the *factor* by which one can accurately extrapolate increases with the sample size. This ability to accurately predict the expected number of new elements observed in larger sample sizes is especially applicable to such settings as genomics, where data is relatively costly to gather, and the benefit of data acquisition is largely dependent on the number of new phenomena discovered.[1]

1.2 Related Work

Much of the work on correcting the empirical distribution of a set of samples builds on the seminal work of Turing, and I.J. Good [22] (see also [23]). In the context of their work at Bletchley Park as part of the British WWII effort to crack the German enigma machine ciphers, Turing and Good developed a simple estimator that corrected the empirical distribution, in some sense to capture the "missing" probability mass of the distribution. This estimator and its variants have been employed widely, particularly in the contexts of genomics, natural language processing, and other settings in which significant portions of the distribution are comprised of domain elements with small probabilities (e.g. [13]). In its most simple form, the Good-Turing frequency estimation scheme estimates the total probability of all domain elements that appear exactly i times in a set of n samples as $\frac{(i+1)\mathcal{F}_{i+1}}{n}$, where $\mathcal{F}_j$ is the total number of domain elements that occur exactly j times in the samples. The total probability mass consisting of domain elements that are not seen in the samples—the "missing" mass, or, equivalently, the probability that the next sample drawn will be a new domain element that has not been seen previously—can be estimated by plugging $i = 0$ into this formula to yield $\mathcal{F}_1/n$, namely the fraction of the samples consisting of domain elements seen exactly once.

The Good–Turing estimate is especially suited to estimating the total mass of elements that appear few times; for

[1]One of the medical benefits of "genome wide association studies" is the compilation of catalogs of rare mutations that occur in healthy individuals; these catalogs are being used to rule out genetic causes of illness in patients, and help guide doctors to accurate diagnoses (see e.g. [18, 19]). Understanding how these catalogs will grow as a function of the number of genomes sequenced may be an important factor in designing such future datasets [37].

more frequently occurring domain elements, this estimate has high variance—for example, if $\mathcal{F}_{i+1} = 0$, as will be the case for most large i, then the estimate is 0. However, for frequently occurring domain elements, the empirical distribution will give an accurate estimate of their probability mass. There is an extremely long and successful line of work, spanning the past 60 years, from the computer science, statistics, and information theory communities, proposing approaches to "smoothing" the Good–Turing estimates, and combining such smoothed estimates with the empirical distribution (e.g. [23, 20, 26, 27, 28, 16, 4]). As was recently shown by Orlitsky and Suresh [29], for the task of learning a distribution of known support with respect to KL divergence, such methods achieve "instance optimal" results analogous to our results for this different distance/divergence function.

Our approach—to first recover an estimate of the unlabeled vector of probabilities of the true distribution and then assign probabilities to the observed elements informed by this recovered vector of probabilities—deviates fundamentally from this previous line of work. This previous work attempts to accurately estimate the total probability mass corresponding to the set of domain elements observed i times, for each i. Even if one knows these quantities *exactly*, such knowledge does not translate into an optimal learning algorithm, and could result in an ℓ_1 error that is a factor of two larger than that of our approach. The following rephrasing of Example 4 from above illustrates this point:

EXAMPLE 5. *Consider n independent draws from a distribution in which* 90% *of the domain elements occur with probability* $10/n$*, and the remaining* 10% *occur with probability* $11/n$*. All variants of the Good-Turing frequency estimation scheme would end up, at best, assigning probability* $10.1/n$ *to most of the domain elements, incurring an ℓ_1 error of roughly* 0.2*. This is because, for elements seen roughly* 10 *times, the scheme would first calculate that the* average *mass of such elements is* $10.1/n$*, and then assign this probability to all such elements. Our scheme, on the other hand, would realize that approximately* 90% *of such elements have probability* $10/n$*, and* 10% *have probability* $11/n$*, but then would assign the probability minimizing the expected error—namely, in this case, our algorithm would assign the* median *probability,* $10/n$*, to all such elements, incurring an ℓ_1 error of approximately* 0.1*.*

Worst-case vs. Instance Optimal Testing and Learning. Sparked by the seminal work of Goldreich, Goldwasser and Ron [21] and that of Batu et al. [7, 6], there has been a long line of work considering distributional property testing, estimation, and learning questions from a *worst case* standpoint—typically parameterized via an upper bound on the support size of the distribution from which the samples are drawn (e.g. [8, 31, 5, 24, 10, 30, 36, 33, 32, 12, 34]).

The desire to go beyond this type of worst-case analysis and develop algorithms which provably perform better on "easy" distributions has led to two different veins of further work. One vein considers different common types of structure that a distribution might possess—such as monotonicity, unimodality, skinny tails, etc., and how such structure can be leveraged to yield improved algorithms (e.g. [14, 9, 15, 11]). While this direction is still within the framework of worst-case analysis, the emphasis is on developing a more nuanced understanding of why "easy" instances are easy, leading to sophisticated algorithms with significantly better performance when the underlying distribution is assumed to be monotonic, unimodal, etc. (We emphasize that assumptions like "monotonicity", that a distribution on labels $1 \dots n$ has probabilities that are a monotonic function of its labels, fall outside the scope of "knowing the *unlabelled* probabilities" against which we benchmark the algorithms of the current paper: monotonicity requires comparing *labelled* probabilities, relative to their labels. Thus our results are not directly comparable to this line of work.)

The current paper lies in a different vein of very recent work going beyond worst-case analysis: the aim is to develop "instance–optimal" algorithms that are capable of exploiting whatever structure is present in the instance, instead of *a priori* designing the algorithm to take advantage of particular foreseen structure.For the problem of identity testing—given the explicit description of p, deciding whether a set of samples was drawn from p versus some distribution with ℓ_1 distance at least ϵ from p—recent work gave an algorithm and an explicit function of p and ϵ that represents the precise sample complexity of this task, for each distribution p [35]. In a similar spirit, with the dual goals of developing optimal algorithms as well as understanding the fundamental limits of when such instance–optimality is not possible, Acharya et al. have a line of work from the perspective of competitive analysis [1, 2, 3, 4]. Broadly, this work explores the following question: to what extent can an algorithm perform as well as if it knew, a priori, the structure of the problem instance on which it was run? For example, the work [2] considers the two-distribution identity testing question: given samples drawn from two unknown distributions, p and q, how many samples are required to distinguish the case that $p = q$ from $||p - q||_1 \geq \epsilon$? They show that if $n_{p,q}$ is the number of samples required by an algorithm that knows, ahead of time, the unlabeled vector of probabilities of p and q, then the sample complexity is bounded by $n_{p,q}^{3/2}$, and that, in general, a polynomial blowup is necessary—there exists p, q for which no algorithm can perform this task using fewer than $n_{p,q}^{7/6}$ samples.

Relation to [32, 34]. The papers [32, 34] were concerned with developing estimators for entropy, support size, etc.—properties that depend only on the unlabeled vector of probabilities of a distribution. The goal in those papers was to give tight *worst-case* bounds on these estimation tasks in terms of a given upper bound on the support size of the distribution in question. In contrast, this work considers the question of *learning* probabilities for each labeled domain element, and considers the more ambitious and practically relevant goal of "instance-optimality". This present paper has two technical components corresponding to the two stages of our algorithm: the first component is recovering an approximation to the unlabeled vector of probabilities, and the second part leverages the recovered unlabeled vector of probabilities to output a labeled vector. While the second part is novel, the majority of the technical machinery that we employ for the first part is based on algorithms and techniques developed in [32, 34], though analyzed here in a more nuanced and general way (a main tool from these works is a Chebyshev polynomial earthmover scheme, which was also repurposed for a rather different end in [33]; the main improvement in the analysis is that our results no longer require any bound on the support size of the distribution, and the results no longer degrade with increasing support

size). We are surprised and excited that these techniques, originally developed for establishing worst-case optimal algorithms for property estimation can be fruitfully extended to yield tight instance-optimal results for such a fundamental and classic learning question.

1.3 Definitions

We refer to the unlabeled vector of probabilities of a distribution as the *histogram* of the distribution. This is simply the histogram, in the usual sense of the word, of the vector of probabilities of the domain elements. We give a formal definition:

DEFINITION 1. *The* histogram *of a distribution p, with a finite or countably infinite support, is a mapping $h_p : (0,1] \to \mathbb{N} \cup \{0\}$, where $h_p(x)$ is equal to the number of domain elements that occur in distribution p with probability x. Formally, $h_p(x) = |\{\alpha : p(\alpha) = x\}|$, where $p(\alpha)$ is the probability mass that distribution p assigns to domain element α. We will also allow for "generalized histograms" in which h_p does not necessarily take integral values.*

In analogy with the histogram of a distribution, it will be convenient to have an unlabeled representation of the set of samples. We define the *fingerprint* of a set of samples, which essentially removes all the label-information. We note that in some of the literature, the fingerprint is alternately termed the *pattern*, *histogram*, *histogram of the histogram* or *collision statistics* of the samples.

DEFINITION 2. *Given samples $X = (x_1, \ldots, x_n)$, the associated* fingerprint*, $\mathcal{F} = (\mathcal{F}_1, \mathcal{F}_2, \ldots)$, is the "histogram of the histogram" of the sample. Formally, $\mathcal{F}$ is the vector whose i^{th} component, $\mathcal{F}_i$, is the number of elements in the domain that occur exactly i times in X.*

The following earthmover metric will be useful for comparing histograms. This metric is similar to that leveraged in [32], but allows for discrepancies in sufficiently small probabilities to be suppressed. This turns out to be the "right" metric for establishing our learning result, as well as our result for the accurate estimation of the expected number of distinct elements that will be observed for larger sample sizes (Proposition 1). In both of these settings, we do not need to worry about accurately estimating extremely small probabilities, as long as we can accurately estimate the total aggregate probability mass comprised of such elements.

DEFINITION 3. *For two distributions p_1, p_2 with respective histograms h_1, h_2, and a real number $\tau \in [0,1]$, we define the* τ-truncated relative earthmover distance *between them, $R_\tau(p_1, p_2) := R_\tau(h_1, h_2)$, as the minimum over all schemes of moving the probability mass in the first histogram to yield the second histogram, where the cost per unit mass of moving from probability x to probability y is $|\log \max(x, \tau) - \log \max(y, \tau)|$.*

The following fact, whose proof is contained in Appendix B, relates the τ-truncated relative earthmover distance between two distributions, p_1, p_2, to an analogous but weaker statement about the ℓ_1 distance between p_1 and a distribution obtained from p_2 by choosing an optimal relabeling of the support:

FACT 1. *Given two distributions p_1, p_2 satisfying*

$$R_\tau(p_1, p_2) \leq \epsilon,$$

there exists a relabeling π of the support of p_2 such that

$$\sum_i |\max(p_1(i), \tau) - \max(p_2(\pi(i)), \tau)| \leq 2\epsilon.$$

The Poisson distribution will also feature prominently in our algorithms and analysis:

DEFINITION 4. *For $\lambda \geq 0$, we define $Poi(\lambda)$ to be the Poisson distribution of parameter λ, where the probability of drawing $j \leftarrow Poi(\lambda)$ equals $poi(\lambda, j) = \frac{e^{-\lambda}\lambda^j}{j!}$.*

2. RECOVERING THE HISTOGRAM

This section adapts the techniques of [32, 34] to accurately estimate the histogram of the distribution in a form that will be useful for Algorithm 2, our ultimate instance-optimal algorithm for learning the distribution, presented and analyzed in Section 3.

The first phase of our algorithm, the step in which we recover an accurate approximation of the histogram of the distribution from which the samples were drawn, consists of solving an intuitive linear program. The variables of the linear program represent the histogram entries $h(x_1), h(x_2), \ldots$ corresponding to a fine discretization of the set of probability values $0 < x_1 < x_2 < \ldots < 1$. The constraints of the linear program represent the fact that h corresponds to the histogram of a distribution, namely all the probabilities sum to 1, and the histogram entries are non-negative. Finally, the objective value of the linear program attempts to ensure that the histogram h output by the linear program will have the property that, if the samples had been drawn from a distribution with histogram h, the expected number of domain elements observed once, twice, etc., would closely match the corresponding actual statistics of the sample. Namely, the objective function tries to ensure that the expected fingerprint of the histogram returned by the linear program is close to the actual fingerprint of the samples.

One minor subtlety is that we will only solve this linear program for the portion of the histogram corresponding to domain elements that are not seen too many times. For elements seen very frequently (at least n^α times for some appropriately chosen absolute constant $\alpha > 0$) their empirical probabilities are likely quite accurate, and we simply use these probabilities. A similar approach was fruitfully leveraged in [32, 34] with the goal of creating worst-case optimal estimators for entropy, and other distributional properties of interest, and a related heuristic was proposed in the 1970's by Efron and Thisted [17], also with the goal of estimating properties of the underlying distribution. As discussed in [34], the fact that our linear program is only responsible for a small portion of the histogram means that the linear program will be small, and in practice can be solved in time sublinear in the number of samples, and thus constitutes only a small fraction of the total (linear) time needed to process n samples.

We state the algorithm and its analysis in terms of two positive constants, $\mathcal{B}, \mathcal{C}$, which can be defined arbitrarily provided the following inequalities hold: $0.1 > \mathcal{B} > \mathcal{C} > \frac{\mathcal{B}}{2} > 0$.

ALGORITHM 1.

Input: Fingerprint $\mathcal{F}$ obtained from n-samples.
Output: Histogram h_{LP}.

- Define the set $X := \{\frac{1}{n^2}, \frac{2}{n^2}, \frac{3}{n^2}, \ldots, \frac{n^{\mathcal{B}}+n^{\mathcal{C}}}{n}\}$.
- For each $x \in X$, define the associated variable v_x, and solve the LP:

$$\text{Minimize } \sum_{i=1}^{n^{\mathcal{B}}} \left| \mathcal{F}_i - \sum_{x \in X} poi(nx, i) \cdot v_x \right|$$

Subject to:

· $\sum_{x \in X} x \cdot v_x + \sum_{i > n^{\mathcal{B}}+2n^{\mathcal{C}}}^{n} \frac{i}{n} \mathcal{F}_i = 1$

(total prob. mass = 1)

· $\forall x \in X, v_x \geq 0$

(histogram entries are non-negative)

- Let h_{LP} be the histogram formed by setting $h_{LP}(x_i) = v_{x_i}$ for all i, where (v_x) is the solution to the linear program, and then for each integer $i > n^{\mathcal{B}} + 2n^{\mathcal{C}}$, incrementing $h_{LP}(\frac{i}{n})$ by $\mathcal{F}_i$.

The following theorem quantifies the performance of the above algorithm:

THEOREM 2. *There exists an absolute constant c such that for sufficiently large n and any $w \in [1, \log n]$, given n independent draws from a distribution p with histogram h, with probability $1 - e^{-n^{\Omega(1)}}$ the generalized histogram h_{LP} returned by Algorithm 1 satisfies*

$$R_{\frac{w}{n \log n}}(h, h_{LP}) \leq \frac{c}{\sqrt{w}}.$$

This theorem is a stronger and more refined version of the results in [32], in that these results no longer require any bound on the support size of the distribution, and the results no longer degrade with increasing support size. Instead, we express our results in terms of a lower bound, $\tau = \frac{w}{n \log n}$,on the probabilities that we are concerned with accurately reconstructing. As in [32], this theorem also implies the following generic framework for estimating "symmetric" properties of distributions: return the value of the property on the distribution output by Algorithm 1. In many cases, this procedure will emulate the performance of the naive "plug-in" estimator applied to $n \log n$ samples, thus providing a generic way to effectively "amplify" the sample size by a factor of $\log n$. (See also [25] for another perspective on emulating $n \log n$ samples using only n.)

We interpret Theorem 2 as saying that Algorithm 1, when run on n independent draws from a distribution, will accurately reconstruct the histogram, in the relative earthmover sense, all the way down to probability $\frac{1}{n \log n}$ (significantly below the $1/n$ threshold where the empirical distribution becomes ineffective). One corollary of independent interest is that this earthmover bound implies that we can accurately extrapolate the number of unique elements that will be seen if we run a new, larger experiment, of size up to $n \log n$. This ability to extrapolate out an extra $\log n$ factor is perhaps unsurprising given the punchline of our earlier work, that for many distributional properties, one can construct estimators that can perform as well as the naive estimator would when given an extra $\log n$ factor of samples [32, 34].

Given a histogram h, for each element of probability x, the probability that it will be seen (at least once) in a sample of size k equals $1 - (1-x)^k$; thus, the expected number of unique elements seen in a sample of size k for a distribution with histogram h equals $\sum_{x:h(x)\neq 0}(1-(1-x)^k) \cdot h(x)$. The following lemma, whose proof is given in Appendix C, shows that this quantity is Lipschitz continuous with respect to truncated relative earthmover distance.

LEMMA 1. *Given two (possibly generalized) histograms g and h, a number of samples k, and a threshold $\tau \in (0, 1]$,*

$$\left| \sum_{x:g(x)\neq 0} (1-(1-x)^k) \cdot g(x) - \sum_{x:h(x)\neq 0} (1-(1-x)^k) \cdot h(x) \right|$$

$$\leq (0.3(k-1)+1) R_\tau(g, h) + \tau \frac{k}{2}.$$

The above lemma together with Theorem 2 yields Proposition 1, which is tight, in the sense that one cannot expect meaningful extrapolation beyond sample sizes of $n \log n$, as shown by the lower bounds in [32].[2]

Towards our goal of devising an optimal learning algorithm, the following corollary of Theorem 2 formalizes the sense that the quality of the histogram output by Algorithm 1 will be sufficient to achieve our optimal learning result, provided that the second phase of our algorithm, described in Section 3, is able to successful label the histogram.

COROLLARY 1. *There exists an algorithm such that, for any function $f(n) = \omega_n(1)$ that goes to infinity as n gets large (e.g. $f(n) = \log \log n$), there is a function $o_n(1)$ that goes to zero as n gets large, such that given n samples drawn independently from any distribution p, the algorithm outputs an unlabeled vector, q, such that, with probability $1-e^{-n^{\Omega(1)}}$, there exists a labeling $\pi(q)$ of the vector q such that*

$$\sum_i \left| \max\left(p(x), \frac{f(n)}{n \log n}\right) - \max\left(\pi(q)(x), \frac{f(n)}{n \log n}\right) \right| < o_n(1),$$

where $p(x)$ denotes the true probability of domain element x in distribution p.

[2]Namely, for some constant c, there exist two families of distribution, $\mathcal{D}_1$ and $\mathcal{D}_2$ such that the distributions in $\mathcal{D}_1$ are close to uniform distributions on $cn \log n$ elements, and the distributions in $\mathcal{D}_2$ are close to uniform distributions over $2cn \log n$ elements, yet given n samples drawn from either a distribution in $\mathcal{D}_1$ selected uniformly at random or from a distribution in $\mathcal{D}_2$ selected uniformly at random, it is information theoretically impossible to decide whether the distribution from which the samples were drawn belonged to $\mathcal{D}_1$ versus $\mathcal{D}_2$ (with probability of success greater than some fixed constant less than 1). The indistinguishability of the families of distributions $\mathcal{D}_1$ and $\mathcal{D}_2$ demonstrates the impossibility of extrapolating beyond sample sizes of $O(n \log n)$, since, with $2cn \log n$ samples, the number of distinct elements observed will—analogously to the corresponding uniform distributions—be either $\approx (1 - \frac{1}{e^2})cn \log n \approx 0.9cn \log n$ for $\mathcal{D}_1$ or $\approx (1 - \frac{1}{e})2cn \log n \approx 1.3cn \log n$ from $\mathcal{D}_2$, which are significantly different from each other.

This corollary is not immediate: the histogram returned by the algorithm might be non-integral, however in the full version, we provide a simple algorithm that rounds a generalized histogram to an (integral) histogram, while changing it very little in relative earthmover distance $R_0(\cdot,\cdot)$. This rounding, together with Fact 1, obtains this corollary.

The utility of the above corollary lies in the following observation: for any function $g(n) = o(1/n)$, the domain elements x that both occur in the n samples and have true probability $p(x) < g(n)$, can account for at most $o(1)$ probability mass, in aggregate. In other words, while the true distribution might have a constant amount, c, of probability mass consisting of domain elements that occur with probability $o(1/n)$, we would observe at most a $o(1)$ fraction of such domain elements in the n samples. Hence, even an optimal scheme that knows the true probabilities would be unable to achieve an ℓ_1 error less than $c - o(1)$ because it does not know the labels of the elements that have not been observed. As we show in the following section, our algorithm will achieve an ℓ_1 error of roughly c.

3. DISAMBIGUATING THE HISTOGRAM

In this section we present our instance-optimal algorithm for learning a distribution from n samples, making use of Algorithm 1 of Section 2 to first accurately infer the histogram of the distribution (in the sense of Corollary 1). As a motivating intuition for the second phase of our algorithm—the phase in which we assign probabilities to the observed elements—consider the behavior of an optimal algorithm that not only knows the true histogram h of the distribution, but also knows for each positive integer j the entire multiset of probabilities of elements that appear exactly j times in the n samples. Since the algorithm has no basis to distinguish between the different elements that each appear j times in the samples, the algorithm may as well assign a single probability m_j to all the items that appear j times in the samples. The optimal m_j in this setting is easily seen to be the *median* of the multiset of probabilities of items appearing j times, as the median is the estimate that minimizes the total (ℓ_1) error of the probabilities.

Our algorithm aims to emulate this idealized optimal algorithm. Of course, we must do this using only an estimate of the histogram, and computing medians based on the likelihoods that elements of probability x will be seen j times in the sample, as opposed to actual knowledge of the multiset of probabilities of the elements observed j times (which was an unreasonably strong assumption, that we made in the previous paragraph because it let us argue about the behavior of the optimal algorithm in that case).

Because our algorithm needs to work in terms of a histogram estimate u, bounded only by the guarantees of Corollary 1, we add an additional "regularization" step that was not needed in the idealized medians setting described above. We "fatten" the histogram u to a new histogram $\bar{u}$ by adding a small amount of probability mass across the range $[\frac{1}{n}, \frac{1}{n}\log^2 n]$, which acts to mollify the effect on the medians of any small errors in the histogram estimate.

Given this "fattened" approximate histogram, we then apply the "medians" intuition: we compute, for each integer j, an appropriate probability with which to label those elements occurring j times in the sample. These estimates are computed via the following thought experiment: imagining $\bar{u}$ to be the true histogram, if we take n samples from the corresponding distribution, *in expectation*, what is the median of the (multiset) of probabilities of those elements seen exactly j times in the sample? We denote this "expected median" by $m_{\bar{u},j,n}$, and our algorithm assigns this probability to each element seen j times in the sample, for $j < \log^2 n$, and assigns the empirical probability $\frac{j}{n}$ for larger j. We formalize this process with the following definition for "Poisson-weighted medians":

DEFINITION 5. *Given a histogram h, let S_h be the multiset of probabilities of domain elements—that is, for each probability x for which $h(x)$ is some positive integer i, add i copies of x to S. Given a number of samples n, and an index j, consider weighting each element $x \in S_h$ by $poi(nx, j)$. Define $m_{h,j,n}$ to be the median of this weighted multiset.*

Explicitly, the median of a *weighted* set of real numbers is a number m such that at most half the weight lies on numbers greater than m, and at most half lies on numbers less than m. Taking advantage of the medians defined above, our learning algorithm follows:

ALGORITHM 2.

Input: n samples from a distribution h.
Output: An assignment of a probability to each nonzero entry of h.

- Run Algorithm 1 to return a histogram u.
- Modify u to create $\bar{u}$ by, for each $j \leq \log^2 n$ adding $\frac{n}{j \log^4 n}$ elements of probability $\frac{j}{n}$ and removing corresponding mass arbitrarily from the rest of the distribution.
- Then to each fingerprint entry $j < \log^2 n$, assign those domain elements probability $m_{\bar{u},j,n}$, (as defined in Definition 5) and to each higher fingerprint entry $j \geq \log^2 n$ assign those domain elements their empirical probability $\frac{j}{n}$.

Theorem 1 *There is a function $err(n)$ that goes to zero as n gets large, such that Algorithm 2, when given as input n independent draws from any distribution p of discrete support, outputs a labeled vector q, such that*

$$E[||p - q||_1] \leq opt(p, n) + err(n),$$

where $opt(p, n)$ is the minimum expected error that any algorithm could achieve on the following learning task: given p, and given n samples drawn independently from a distribution that is identical to p up to an arbitrary relabeling of the domain elements, learn the distribution.

The core of the proof of Theorem 1 relies on constructing an estimate, $dev_{j,n}(A, m_{B,j,n})$, that captures the expected contribution to the ℓ_1 error due to elements that occur exactly j times, given that the true distribution we are trying to reconstruct has histogram A, and our reconstruction is based on the medians $m_{B,j,n}$ derived from a (possibly different) histogram B. The proof then has two main components. First we show that $dev_{j,n}(h, m_{h,j,n})$ approximately captures the performance of the optimal algorithm with very high probability, namely that using the true histogram h to choose medians $m_{h,j,n}$ lets us estimate the performance of

the best possible algorithm. This step is slightly subtle, and implies that, given h, an algorithm to compute the probability assigned to an element that occurs j times can glean at most $o(1)$ added benefit by using 1) j, 2) h and 3) the entire set of samples, rather than just 1) j and 2) h.

Next, we show that the clean functional form of $dev(\cdot,\cdot)$ implies that this performance estimate varies slowly with respect to changes in the histogram used to choose the median that is input to the second term, and thus that with only negligible performance loss we may reconstruct distributions using medians derived from an *estimate* u of the true histogram (instead of the inaccessible true histogram itself), thus allowing us to analyze the actual performance of Algorithm 2.

Beyond these two core steps, the analysis of Algorithm 2 is somewhat delicate—because our algorithm is instance-optimal to $o(1)$ error, it must reuse samples both for the Algorithm 1 histogram reconstruction and for the final labeling step, and we must carefully separate the probabilistic portion of the analysis via a clean set of assumptions which 1) will hold with near certainty over the sampling process, and 2) are sufficient to guarantee the performance of both stages of our algorithm. The complete proof is contained in Appendix A.

4. REFERENCES

[1] J. Acharya, H. Das, A. Jafarpour, A. Orlitsky, and S. Pan. Competitive closeness testing. In *Conference on Learning Theory (COLT)*, 2011.

[2] J. Acharya, H. Das, A. Jafarpour, A. Orlitsky, S. Pan, and A. Suresh. Competitive classification and closeness testing. In *Conference on Learning Theory (COLT)*, 2012.

[3] J. Acharya, H. Das, A. Jafarpour, A. Orlitsky, S. Pan, and A. Suresh. A competitive test for uniformity of monotone distributions. In *AISTATS*, 2013.

[4] J. Acharya, A. Jafarpour, A. Orlitsky, and A.T. Suresh. Optimal probability estimation with applications to prediction and classification. In *COLT*, 2013.

[5] T. Batu, S. Dasgupta, R. Kumar, and R. Rubinfeld. The complexity of approximating the entropy. In *Symposium on Theory of Computing (STOC)*, 2002.

[6] T. Batu, E. Fischer, L. Fortnow, R. Kumar, R. Rubinfeld, and P. White. Testing random variables for independence and identity. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, 2001.

[7] T. Batu, L. Fortnow, R. Rubinfeld, W.D. Smith, and P. White. Testing that distributions are close. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, 2000.

[8] T. Batu, R. Kumar, and R. Rubinfeld. Sublinear algorithms for testing monotone and unimodal distributions. In *Symposium on Theory of Computing (STOC)*, 2004.

[9] M. Brautbar and A Samorodnitsky. Approximating entropy from sublinear samples. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2007.

[10] A. Chakrabarti, G. Cormode, and A. McGregor. A near-optimal algorithm for computing the entropy of a stream. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2007.

[11] S. Chan, I. Diakonikolas, R. Servedio, and X. Sun. Efficient density estimation via piecewise polynomial approximation. In *STOC*, pages 604–613, 2014.

[12] S. Chan, I. Diakonikolas, G. Valiant, and P. Valiant. Optimal algorithms for testing closeness of discrete distributions. In *SODA*, 2014.

[13] S. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. *Computer Speech and Language*, 13:359–394, 1999.

[14] C. Daskalakis, I. Diakonikolas, R. A. Servedio, G. Valiant, and P. Valiant. Testing k-modal distributions: Optimal algorithms via reductions. *Manuscript*, 2011.

[15] I. Diakonikolas, D. Kane, and V. Nikishkin. Testing identity of structured distributions. In *SODA*, 2015.

[16] E. Drukh and Y. Mansour. Concentration bounds for unigrams language model. In *COLT*, 2004.

[17] B. Efron and R. Thisted. Estimating the number of unseen species: how many words did Shakespeare know? *Biometrika*, 63(3):435–447, 1976.

[18] 1000 Genomes Project Consortium et al. A global reference for human genetic variation. *Nature*, 526(7571):68–74, 2015.

[19] D. G. MacArthur et al. A systematic survey of loss-of-function variants in human protein-coding genes. *Science*, 335(6070):823–828, 2012.

[20] W.A. Gale and G. Sampson. Good-Turing frequency estimation without tears. *Journal of Quantitative Linguistics*, 2(3):217–237, 1995.

[21] O. Goldreich, S. Goldwasser, and D. Ron. Property testing and its connection to learning and approximation. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, 1996.

[22] I. J. Good. The population frequencies of species and the estimation of population parameters. *Biometrika*, 40(16):237–264, 1953.

[23] I.J. Good and G.H. Toulmin. The number of new species, and the increase in population coverage, when a sample is increased. *Biometrika*, 43:45–63, 1956.

[24] S. Guha, A. McGregor, and S. Venkatasubramanian. Streaming and sublinear approximation of entropy and information distances. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2006.

[25] Jiantao Jiao, Kartik Venkat, Yanjun Han, and Tsachy Weissman. Minimax estimation of functionals of discrete distributions. *IEEE Trans. Information Theory*, 61(5):2835–2885, 2015.

[26] D.A. McAllester and R.E. Schapire. On the convergence rate of Good-Turing frequency estimators. In *COLT*, 2000.

[27] A. Orlitsky, N.P. Santhanam, and J. Zhang. Always Good Turing: asymptotically optimal probability estimation. *Science*, 302(5644):427–431, October 2003.

[28] A. Orlitsky, N.P. Santhanam, and J. Zhang. Always Good Turing: asymptotically optimal probability estimation. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, 2003.

[29] A. Orlitsky and A. Suresh. Competitive distribution estimation: Why is good-turing good. In *Neural Information Processing Systems (NIPS)*, 2015.

[30] L. Paninski. Estimating entropy on m bins given fewer than m samples. *IEEE Trans. on Information Theory*, 50(9):2200–2203, 2004.

[31] S. Raskhodnikova, D. Ron, A. Shpilka, and A. Smith. Strong lower bounds for approximating distribution support size and the distinct elements problem. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, 2007.

[32] G. Valiant and P. Valiant. Estimating the unseen: an $n/\log(n)$–sample estimator for entropy and support size, shown optimal via new CLTs. In *Proceedings of the ACM Symposium on Theory of Computing (STOC)*, 2011.

[33] G. Valiant and P. Valiant. The power of linear estimators. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, 2011.

[34] G. Valiant and P. Valiant. Estimating the unseen: improved estimators for entropy and other properties. In *Neural Information Processing Systems (NIPS)*, 2013.

[35] G. Valiant and P. Valiant. An automatic inequality prover and instance optimal identity testing. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, 2014.

[36] P. Valiant. Testing symmetric properties of distributions. *SIAM Journal on Computing*, 40(6):1927–1968, 2011.

[37] James Zou, Gregory Valiant, Paul Valiant, Konrad Karczewski, Siu On Chan, Kaitlin Samocha, Mokol Lek, Shamil Sunyaev, Mark Daly, Daniel MacArthur, et al. Quantifying the unobserved protein-coding variants in human populations provides a roadmap for large-scale sequencing projects. *bioRxiv*, page 030841, 2015.

APPENDIX

In this appendix we provide complete proofs for the results of this paper, with the exception of the proof of Theorem 2—the theorem characterizing the performance of Algorithm 1 that recovers an accurate estimate of the histogram. This proof leverages the techniques developed in [32, 34], in particular a Chebyshev polynomial earthmover scheme. This proof is omitted from this extended abstract due to the space constraints of the conference proceedings, and is given in the full version, available at `http://arxiv.org/abs/1504.05321`.

A. PROOF OF THEOREM 1

In this section we give a self-contained proof of the correctness of Algorithm 2, establishing Theorem 1.

A.1 Separating the Probabilistic Portion of the Analysis

Our analysis is somewhat delicate because we reuse the same samples both to estimate the histogram h, and then to label the domain elements given an approximate histogram. For this reason, we will very carefully separate out the probabilistic portion of the sampling process, identifying a list of convenient properties which happen with very high probability in the sampling process, and then deterministically analyze the case when these properties hold, which we will refer to as a "faithful" set S of samples from the distribution.

We first describe a simple discretization of histograms h, dividing the domain into buckets which will simplify further analysis, and is a crucial component of the definition of "faithful".

DEFINITION 6. *Given a histogram h, and a number of samples n, define the kth* bucket *of h to consist of those histogram entries with probabilities in the half-open interval $(\frac{k}{n\log^2 n}, \frac{k+1}{n\log^2 n}]$. Letting h_k be h restricted to its kth bucket, define $B_{poi}(j,k) = \sum_{x:h_k(x)\neq 0} h(x)poi(nx,j)$ to be the expected number of elements from bucket k that are seen exactly j times, if $Poi(n)$ samples are taken. Given a set of samples S, let $B_S(j,k)$ be the number of elements in bucket k of h that are seen exactly j times in the samples S, where in both cases h and n are implicit in the notation.*

Given this notion of "buckets", we define faithful to mean 1) each domain element is seen roughly the number of times we would expect to see it, and 2) for each pair (j,k), the number of domain elements from bucket k that are seen exactly j times is very close to its expectation (where we compute expectations under a Poisson distribution of samples, because "Poissonization" will simplify subsequent analysis). The first condition of "faithful" gives weak control on which fingerprint entry each domain element will contribute to, while the second condition gives much stronger control over the aggregate contribution to fingerprint entries by all domain elements within a certain probability "bucket".

DEFINITION 7. *Given a histogram h and a number of samples n, a set of n samples, S, is called* faithful *if the following two conditions are satisfied:*

1. *Each item of probability x appears in the samples a number of times j satisfying*
$$|nx - j| < \max\{\log^{1.5} n, \sqrt{nx\log^{1.5} n}\}.$$
2. *For each $j < \log^2 n$ and k, we have*
$$|B_{poi}(j,k) - B_S(j,k)| < n^{0.6}.$$

This notion of "faithful" holds with near certainty, as shown in the following lemma, allowing us to assume (when specified) in the results in the rest of this section that our learning algorithm receives a faithful set of samples.

LEMMA 2. *For any histogram h and number of samples n, with probability $1 - n^{-\omega(1)}$, a set of n samples drawn from h will be faithful.*

PROOF. Since the number of times an item of probability x shows up in n samples is the binomial distribution $Bin(n,x)$, the first condition of "faithful"—essentially that this random variable will be within $\log^{3/4} n$ standard deviations of its mean— follows with probability $1 - n^{-\omega(1)}$ from standard Chernoff/Hoeffding bounds.

For the second condition, since $Poi(n)$ has probability $\Theta(1/\sqrt{n})$ of equaling n, we consider the related process where $Poi(n)$ samples are drawn. The number of times each domain element x is seen is now distributed as $Poi(nx)$, independent of each other domain element. Thus the number of

elements from bucket k seen exactly j times is the sum of independent Bernoulli random variables, one for each domain element in bucket k. The expected number of such elements is $B_{poi}(j,k)$ by definition. Since $B_{poi}(j,k) \leq n$ by definition, we have that the variance of this random variable is also at most n, and thus Chernoff/Hoeffding bounds imply that the probability that it deviates from its expectation by more than $n^{0.6}$ is at most $exp(-n^{0.1})$. Thus the probability of such a deviation is at most a $\Theta(\sqrt{n})$ factor higher when taking exactly n samples than when taking $Poi(n)$ samples; taking a union bound over all j and k yields the desired result. □

A.2 An Estimate of the Optimal Error

We now introduce the key definition of $dev(\cdot,\cdot)$, which underpins our analysis of the error of estimation algorithms. The definition of $dev(\cdot,\cdot)$ captures the following process: Suppose we have a probability value m_j, and will assign this probability value to every domain element that occurs exactly j times in the samples. We estimate the expected error of this reconstruction, in terms of the probability that each domain element shows up exactly j times. While the below definition, stated in terms of a Poisson process, is neither clearly related to the optimal error $opt(h,n)$, nor the actual error of any specific algorithm, it has particularly clean properties which will help us show that it can be related to both $opt(h,n)$ (in this subsection) as well as the expected error achieved by Algorithm 2 (shown in Section A.3).

DEFINITION 8. *Given a histogram h, a real number m, a number of samples n, and a nonnegative integer j, define $dev_{j,n}(h,m) = \sum_{x:h(x)\neq 0} |x-m|h(x)poi(nx,j)$.*

Intuitively, $dev_{j,n}(h,m)$ describes the expectation—over taking $Poi(n)$ samples from h—of the sum of the deviations between m and each probability x of a element seen j times among the samples. Namely, $dev_{j,n}(h,m)$ describes to what degree m is a good probability to which we can ascribe all domain elements seen j times, among $\approx n$ samples from h.

This definition provides crucial motivation for how Definition 5 sets the medians $m_{h,j,n}$ used in Algorithm 2, since $m_{h,j,n}$ is the value of m that minimizes the previous definition, $dev_{j,n}(h,m)$, since both are defined via the same Poisson weights $poi(nx,j)$. (The median of a—possibly weighted—set of numbers is the location m that minimizes the total—possibly weighted—distance from the set to m.)

We now show the key result of this section, that the definition of "faithful" induces precise guarantees on the spread of probabilities of those elements seen j times. Subsequent lemmas will relate this to the performance of both the optimal algorithm and to our own Algorithm 2.

LEMMA 3. *Given a histogram h, let S be the multiset of probabilities of a faithful set of samples of size n. For each index $j < \log^2 n$, consider those domain elements that occur exactly j times in the samples and let S_j be the multiset of probabilities of those domain elements. Let σ_j be the sum over S_j of each element's distance from the median (counting multiplicity) of S_j. Then $\sum_{j<\log^2 n} |\sigma_j - dev_{j,n}(h, m_{h,j,n})| = O(\log^{-2} n)$.*

PROOF. Recall that σ computes the total distance of the (unweighted) multiset S_j from its median, while $dev_{j,n}(h, m_{h,j,n})$ is an analogous (weighted) quantity for the true histogram, with each entry x having multiplicity $h(x)$ and weight $poi(nx,j)$. In the first case, sampling means that each element of probability x either shows up exactly j times (with some binomial probability) and is counted with weight 1, or does not show up j times and is not counted; in the second case, instead of sampling, each entry x from the histogram is counted with weight $poi(nx,j) < 1$, capturing roughly the average effect of sampling (except with Poisson instead of binomial weight). By the definition of "faithful", the total weight coming from each bucket k in both cases is within $n^{0.6}$ of each other (since $j < \log^2 n$). We consider only buckets $k \leq 2\log^4 n$, corresponding to probabilities less than $\frac{2}{n}\log^2 n$, since the first condition of "faithful" means that no higher probability elements will be seen $j < \log^2 n$ times.

Consider transforming one weighted multiset into the other (where elements of S_j are interpreted as having weight 1 each), maintaining a bound on how much the total distance from the median changes. We make crucial use of the fact that the "total distance to the median" is robust to small changes in the weighted multiset, since the median is the location that minimizes this total distance. Moving α weight by a distance of β can increase the total (weighted) distance to the median by at most $\alpha\cdot\beta$ since this is how much the total weighted distance to the *old* median changes, and the new median must be at least as good; conversely, such a move cannot decrease the total distance by more than $\alpha\cdot\beta$ as the inverse move would violate the previous bound. Adding α weight to the distribution at distance β from the current median similarly cannot decrease the total distance, but also cannot increase the total distance by more than $\alpha\cdot\beta$, with the corresponding statements holding for removing α weight.

Thus, transforming all the S_j into the weighted multiset where each entry x has multiplicity $h(x)$ and weight $poi(nx,j)$ requires two types of transformations: 1) moving up to n samples within their buckets; 2) adding or removing up to $n^{0.6}$ weight from buckets for various combinations of j and k. Since buckets have width $1/(n\log^2 n)$, transformations of the first type change the total distance to the median by at most $\log^{-2} n$; since $j < \log^2 n$ and all buckets above probability $\frac{2}{n}\log^2 n$ are empty, transformations of the second type change the total distance by at most the product of the weight adjustment $n^{0.6}$, the number of j,k pairs $2\log^6 n$, and size of the probability range under consideration which is $\frac{2}{n}\log^2 n$, yielding a bound of $\frac{4}{n^{0.4}}\log^8 n$. Thus in total the change is $O(\log^{-2} n) = o(1)$ as desired. □

The above lemma essentially shows that $dev_{j,n}(h, m_{h,j,n})$ captures how well we could hypothetically estimate the probabilities of all the domain elements seen j times, under the unrealistically optimistic assumption that we know the (unlabeled) multiset of probabilities of elements seen j times and estimate all these probabilities optimally by their median. Before showing how our algorithm can perform almost this well based on only the samples, we first formalize this reasoning.

DEFINITION 9. *We call a distribution learner "simple" if all the domain elements seen exactly j times in the samples get assigned the same probability.*

Given n samples from a distribution p, with $p_{(j)}$ being those domain elements that occurred exactly j times in the sample, we note that the probability of obtaining these samples

is invariant to any permutation of $p_{(j)}$. Thus if a hypothetical learner L assigns different probabilities to different elements seen j times in the sample, then its average performance over a random permutation of the domain elements can only improve if we simplify L, by having it instead assign to all the elements seen j times the *median* of the multiset that it was originally assigning.

For this reason, when we are discussing an optimal distribution learner, we will henceforth assume it is simple.

LEMMA 4. *Given a histogram h, let S be the multiset of probabilities of a faithful set of samples of size n. Given an index $j < \log^2 n$, consider those domain elements that occur exactly j times in the sample; let S_j be the multiset of probabilities of those domain elements. Let σ_j be the sum over S_j of each element's distance from the median of S_j (counting multiplicity). Then any simple learner, when given the sample, must have error at least σ_j on the domain elements that appear j times in the sample.*

PROOF. The median of S_j is the best possible estimate any simple learner can yield—even given the true distribution—so the error of this estimate bounds the performance of a simple learner. □

Combining this with Lemma 3 immediately yields:

COROLLARY 2. *For any distribution h, the total error of any simple learning algorithm, given n faithful samples from h, is at least $\left(\sum_{j<\log^2 n} dev_{j,n}(h, m_{h,j,n})\right) - O(\log^{-2} n)$. Further, for any algorithm—simple or not—if we average its performance over all relabelings of the domain of h and the corresponding relabeled samples, it will have expected error bounded by the same expression.*

A.3 The Error Estimate is Lipschitz with respect to Mis-estimating the Distribution

We now relate the error bound of Corollary 2 to the performance of our algorithm, via two steps. The bound in the corollary is in terms of $m_{h,j,n}$, the medians computed in terms of the true histogram h which is unknown to the algorithm; instead the algorithm works with an estimate $\bar{u}$ of the true histogram. The next lemma shows that estimating in terms of $\bar{u}$ is almost as good as using h.

FACT 2. *For any distribution h, index $j \geq 1$, and real parameter $t \geq 1$, weighting each domain element x by $poi(nx, j)$, the total weight on domain elements that are at least t standard deviations away from $\frac{j}{n}$—namely, for which $|nx - j| \geq t\sqrt{j}$ is at most $n \cdot exp(-\Omega(t))$.*

LEMMA 5. *Given a number of samples n, a histogram h and a second histogram $\bar{u}$ that is 1) close to h in the sense of Corollary 1, in that there exists distributions p, q corresponding to $h, \bar{u}$ respectively for which $\sum_i |\max(p(i), \frac{1}{n}\log^{-0.25} n) - \max(q(i), \frac{1}{n}\log^{-0.25} n)| \leq \log^{-0.25} n$, and 2) the histogram $\bar{u}$ is "fattened" in the sense that for each $j \leq \log^2 n$ there are at least $\frac{n}{j\log^4 n}$ elements of probability $\frac{j}{n}$. Then*

$$\sum_{j<\log^2 n} dev_{j,n}(h, m_{\bar{u},j,n}) \leq o(1) + \sum_{j<\log^2 n} dev_{j,n}(h, m_{h,j,n}).$$

Since for each j, as noted earlier, $m_{h,j,n}$ is the quantity which minimizes $dev_{j,n}(h, m)$, each term $dev_{j,n}(h, n_{\bar{u},j,n})$ on the left hand side is greater than or equal to the corresponding $dev_{j,n}(h, n_{h,j,n})$ on the right hand side, so the lemma implies that the left and right hand sides of the expression in the lemma, beyond having related sums, are in fact term-by-term close to each other.

The proof relies on first comparing $m_{h,j}$ and $m_{\bar{u},j}$ to $\frac{j}{n}$, and then showing that $dev_j(h, m)$ is Lipshitz with respect to changes in h of the type described by the guarantees of Corollary 1.

PROOF OF LEMMA 5. We drop the "$, n$" subscripts here for notational convenience.

Recall that the quantities $m_{h,j}$ and $m_{\bar{u},j}$ are medians computed after weighting by a Poisson function centered at j, and thus we would expect these medians to be close to $\frac{j}{n}$. We first show that the "fattening" condition makes $m_{\bar{u},j}$ well-behaved (namely, close to $\frac{j}{n}$), and then show, given this, that the lemma works both in the case that $m_{h,j}$ is far from $\frac{j}{n}$, and then for the case where it is close.

By condition 2 of the lemma, the "fattening" assumption, for any index $j < \log^2 n$, we have $\sum_{x:\bar{u}(x)\neq 0} h(x)poi(nx, j) = 1/\log^{O(1)} n$. Thus, by Fact 2, the median $m_{\bar{u},j}$ must satisfy $|n \cdot m_{\bar{u},j} - j| < \sqrt{j}\log^{o(1)} n$, since the fraction of the Poisson-weighted distribution that is at locations more than $\frac{1}{n}\sqrt{j}\log^{\Theta(1)} n$ distance from $\frac{j}{n}$ is (much) less than 1/2.

Given the above bound on $m_{\bar{u},j}$, we now turn to $m_{h,j}$. Consider the case $|n \cdot m_{h,j} - j| > \sqrt{j}\log^{0.1} n$. By Fact 2, weighting each domain element x by $poi(nx, j)$, the total weight on the far side of the median $m_{h,j}$ from $\frac{j}{n}$, is at most $n \cdot exp(-\Omega(\log^{0.1} n))$. Since (by definition of "median") half the weight is on each side of the median, the total weight $\sum_{x:h(x)\neq 0} h(x)poi(nx, j)$ must also be bounded by $n \cdot exp(-\Omega(\log^{0.1} n))$. Recall the definition of the left hand side of the inequality of the lemma, $dev_j(h, m_{\bar{u},j}) = \sum_{x:h(x)\neq 0} |x - m_{\bar{u},j}|h(x)poi(nx, j)$. Thus for the portion of this sum where $x < \frac{2}{n}\log^2 n$, since from the previous paragraph $m_{\bar{u},j}$ is also bounded by $\frac{2}{n}\log^2 n$ for large enough n, we can bound $\sum_{x<\frac{2}{n}\log^2 n:h(x)\neq 0} |x - m_{\bar{u},j}|h(x)poi(nx, j)$ by the product $n \cdot exp(-\Omega(\log^{0.1} n)) \cdot \frac{2}{n}\log^2 n = exp(-\Omega(\log^{0.1} n))$. For those $x \geq \frac{2}{n}\log^2 n$, since $j < \frac{1}{n}\log^2 n$, we have the tail bounds $poi(nx, j) = n^{-\omega(1)}$, implying the total for such x is also bounded by $exp(-\Omega(\log^{0.1} n))$, which is our final bound for this case—summing these bounds over all $j < \log^2 n$ yields the desired bound $\sum_{j<\log^2 n} dev_j(h, m_{\bar{u},j}) \leq o(1)$, where the sum is over those j for which this case applies, $|n \cdot m_{h,j} - j| > \sqrt{j}\log^{0.1} n$.

Thus it remains to prove the claim when both $m_{h,j}$ and $m_{\bar{u},j}$ are close to $\frac{j}{n}$. To analyze this case, we show that $dev_j(h, m)$ is Lipschitz with respect to the closeness in h and $\bar{u}$ guaranteed by condition 1 of the lemma, provided $|n \cdot m - j| \leq \sqrt{j}\log^{0.1} n$. The guarantee on h and $\bar{u}$ means that one can transform one distribution into the other by two kinds of transformations: 1) changing the distributions by $\log^{-0.25} n$ in the ℓ_1 sense, and 2) arbitrary mass-preserving transformation of elements of probability less than $\frac{1}{n}\log^{-0.25} n$. We thus bound the change in $dev_j(h, m)$ under both types of transformations.

To analyze ℓ_1 modifications, consider an arbitrary probability x, and consider the derivative of $dev_j(h, m)$ as we take an element of probability x and change x. Recalling the definition $dev_j(h, m) = \sum_{x:h(x)\neq 0} |x - m|h(x)poi(nx, j)$, we see that this derivative equals $\frac{d}{dx}|x - m|poi(nx, j)$, which

is bounded (by the product rule and triangle inequality) as $poi(nx,j) + |x-m|\frac{d}{dx}poi(nx,j)$, where $\frac{d}{dx}poi(nx,j) = n \cdot poi(nx,j-1)\cdot(1-\frac{nx}{j})$. Rewriting m as m_j to indicate its dependence on j, we want to bound the sum of this derivative over $j < \log^2 n$, since the exact dependence for each individual j is much harder to talk about than the overall dependence. We have $\sum_j poi(nx,j)+|x-m_j|n\cdot poi(nx,j-1)\cdot(1-\frac{nx}{j})$, where $\sum_j poi(nx,j) \le 1$. To bound the remaining part of the sum, we first consider the case $x < \frac{1}{n}$, in which case we bound $|x-m_j| \le \frac{1}{n}(1+j+\sqrt{j}\log^{0.1} n)$ and $(1-\frac{nx}{j}) \le 1$, thus yielding the bound $\sum_{j\ge 1}|x-m_j|n\cdot poi(nx,j-1)\cdot(1-\frac{nx}{j}) \le \sum_{j\ge 0}(2+j+\sqrt{j+1}\log^{0.1} n)poi(nx,j) \le \sum_{j\ge 0}(2+j+\sqrt{j+1}\log^{0.1} n)/j! = O(\log^{0.1} n)$. For $x \ge \frac{1}{n}$, since $poi(nx,j-1)$ decays exponentially fast for j more than $\sqrt{nx}$ away from nx, we can bound this sum as being on the order of $\sqrt{nx}$ times its maximum value when j is in this range. In this range we have $|x-m_j| \le |x-\frac{j}{n}| + |\frac{j}{n}-m_j| = \frac{1}{n}O(\sqrt{nx}\log^{0.1} n)$, and $poi(nx,j-1) = O(\frac{1}{\sqrt{nx}})$, and $(1-\frac{nx}{j}) = O(1/\sqrt{nx})$, yielding a total bound of $O(\sqrt{nx}\sqrt{nx}\frac{1}{\sqrt{nx}}\frac{1}{\sqrt{nx}}\log^{0.1} n) = O(\log^{0.1} n)$ as in the previous case. Thus we conclude that the sum over all j of the amount $dev_j(h,m)$ changes with respect to ℓ_1 changes in h is $O(\log^{0.1} n)$.

We next bound the total change to $dev_j(h,m)$ induced by the second type of modification, arbitrary mass-preserving transformations of elements of probability $x < \frac{1}{n}\log^{-0.25} n$. For $j = 1$, we bound the components of

$$dev_j(h,m) = \sum_{x:h(x)\neq 0} |x-m|h(x)poi(nx,j),$$

by bounding the two terms in the product: $|x-m| \in [m-\frac{1}{n}\log^{-0.25} n, m+\frac{1}{n}\log^{-0.25} n]$, and $poi(nx,1) = nx\cdot e^{-nx} \in [nx(1-\log^{-0.25} n)^2, nx]$. Thus for m either m_h or $m_{\bar{u}}$, since by the assumption of this case $m \le \frac{1}{n}(1+\log^{0.1} n)$, from the bounds above, the contribution to $dev_j(h,m)$ from those $x < \frac{1}{n}\log^{-0.25} n$ is within $o(1)$ of mn times the total mass in the distribution below $\frac{1}{n}\log^{-0.25} n$, showing that arbitrary modifications of the second type modify $dev_1(h,m)$ by $o(1)$.

Analyzing the remaining $j \ge 2$ terms, omitting the $|x-m|$ multiplier for the moment, we have that

$$\sum_{x<\frac{1}{n}\log^{-0.25} n : h(x)\neq 0} h(x)poi(nx,j) \le n(\log^{-0.25} n)^{j-1}.$$

Because of the bound that $m_h, m_{\bar{u}}$ are each within $\frac{1}{n}\sqrt{j}\log^{0.1} n$ of $\frac{j}{n}$, we have that $|x-m| \le \frac{1}{n}(\log^{-0.25} n + j + \sqrt{j}\log^{0.1} n)$. Thus the change to $dev_j(h,m)$ from changes of the second type, summed over all $j \ge 2$, is bounded by the sum

$$\sum_{j\ge 2}(\log^{-0.25} n)^{j-1}\left(\log^{-0.25} n + j + \sqrt{j}\log^{0.1} n\right) = o(1),$$

as desired.

Putting the pieces together, the closeness of h and $\bar{u}$ implies by the above Lipschitz argument that changing the distribution between h and $\bar{u}$, under the fixed median $m_{\bar{u},j}$ does not increase $dev(\cdot,\cdot)$ too much: $\sum_{j<\log^2 n} dev_j(h,m_{\bar{u},j}) \le o(1)+\sum_{j<\log^2 n} dev_j(\bar{u},m_{\bar{u},j})$. Further, since $m_{\bar{u},j}$ minimizes this last expression, the right hand side can only increase if we replace $dev_j(\bar{u},m_{\bar{u},j})$ by $dev_j(\bar{u},m_{h,j})$ in this last inequality. Finally, a second application of the same Lipschitz property implies

$$\sum_{j<\log^2 n} dev_j(\bar{u},m_{h,j}) \le o(1) + \sum_{j<\log^2 n} dev_j(h,m_{h,j}).$$

Combining these three inequalities yields the bound of the lemma:

$$\sum_{j<\log^2 n} dev_j(h,m_{\bar{u},j}) \le o(1) + \sum_{j<\log^2 n} dev_j(h,m_{h,j}).$$

□

The following lemma characterizes the effect of "fattening" in the second step of Algorithm 2, showing that this slight modification to the histogram keeps the resulting medians small enough that me may apply the following Lemma 7.

LEMMA 6. *For sufficiently large n, given a fattened distribution $\bar{\mu}$, for any $j < \log^2 n$, the median $m_{\bar{\mu},j,n}$ is at most $\frac{2}{n}\log^2 n$.*

PROOF. Recall that $m_{\bar{\mu},j,n}$ is defined as the median of the multiset of probabilities of $\bar{u}$ after each probability x has been weighted by $poi(xn,j)$. For $x \ge \frac{2}{n}\log^2 n$ and $j < \log^2 n$, these weights will each be $n^{-\Omega(1)}$ small by Poisson tail bounds; and because of the fattening, the elements added at probability $\frac{j}{n}$ will contribute inverse polylogarithmic weight. Since the median must have at most half the weight to its left, the median cannot be as large as our bound $\frac{2}{n}\log^2 n$, as desired. □

Given the above bound on the size of medians for small j, the following lemma shows that our $dev(\cdot,\cdot)$ estimates accurately capture the performance of these medians on any faithful set of samples.

LEMMA 7. *Given a histogram h, a number of samples n, and for each fingerprint entry $j < \log^2 n$ a probability $m_j < \frac{2}{n}\log^2 n$ to which we attribute each domain element that shows up j times in the sample, then for any faithful set of samples from h, the total error made for all $j < \log^2 n$ is within $o(1)$ of $\sum_{j<\log^2 n} dev_{j,n}(h,m_j)$.*

PROOF. Recalling the "buckets" from Definition 6, consider for arbitrary integer k, those elements of h in bucket k, which we denote h_k—namely, those probabilities of h lying in the interval $(\frac{k}{n\log^2 n}, \frac{k+1}{n\log^2 n}]$, where by the first condition of "faithful", none of these probabilities are above $\frac{2}{n}\log^2 n$ for large enough n. Further, let $S_{j,k}$ be the multiset of probabilities of those domain elements from bucket k of h that each get seen exactly j times in the sample. The total error of our estimate m_j on bucket k is thus $\sum_{x\in S_{j,k}}|m_j - x|$, which since buckets have width $1/(n\log^2 n)$, is within $|S_{j,k}|/(n\log^2 n)$ of $|S_{j,k}|\cdot|m_j - k/(n\log^2 n)|$, where we have approximated each x by the left endpoint of the bucket containing x. By the second condition of "faithful", $S_{j,k}$ is within $n^{0.6}$ of its expectation, $B_{poi}(j,k)$, and since by assumption $m_j < \frac{2}{n}\log^2 n$, we have that our previous error bound $|S_{j,k}|\cdot|m_j - k/(n\log^2 n)|$ is within $\frac{2}{n^{0.4}}\log^2 n$ of $B_{poi}(j,k)\cdot|m_j - k/(n\log^2 n)|$. We rewrite this final expression via the definition of B_{poi} as

$$\sum_{x:h_k(x)\neq 0} |m - k/(n\log^2 n)|h(x)poi(nx,j).$$

We compare this final expression to the portion of the deviation $dev_{j,n}(h, m_j)$ that comes from bucket k, namely

$$\sum_{x:h_k(x)\neq 0} |m_j - x|h(x)poi(nx, j),$$

where since $\sum_{x:h_k(x)\neq 0} |m_j - x|h(x)poi(nx, j) = B_{poi}(j, k)$ and x is within $1/(n\log^2 n)$ of $k/(n\log^2 n)$, the difference between them is clearly bounded by $B_{poi}(j, k)/(n\log^2 n)$. Using the triangle inequality to add up the three error terms we have accrued yields that our estimate for the ℓ_1 error we make for elements seen j times from bucket k is accurate to within

$$|S_{j,k}|/(n\log^2 n) + \frac{2}{n^{0.4}}\log^2 n + B_{poi}(j, k)/(n\log^2 n).$$

We sum this error bound over all $2\log^4 n$ buckets k and all indices $j < \log^2 n$. The middle term $\frac{2}{n^{0.4}}\log^2 n$ clearly sums up to $o(1)$ over all j, k pairs. Further, since $S_{j,k}$ is within $n^{0.6}$ of $B_{poi}(j, k)$ by the definition of faithful, the sum of the first term is within $o(1)$ of the sum of the third term and it remains only to analyze the third term involving $B_{poi}(j, k)$. From its definition, $\sum_{j,k} B_{poi}(j, k)$ is the expected number of distinct items seen, when making $Poi(n)$ draws from the distribution, throwing out those elements which violate the j and k constraints; hence this sum over all j, k pairs is at most n, bounding the total error of our "*dev*" estimates by $O(1/\log^2 n)$, as desired. □

A.4 Proof of Theorem 1

We now assemble the pieces and prove Theorem 1.

Proof of Theorem 1. Consider the output of Algorithm 1 as run in the first step of Algorithm 2. Corollary 1 outlines two cases: with $o(1)$ probability the closeness property outlined in the proposition fails to hold, and in this case, Algorithm 2 may output a distribution up to ℓ_1 distance 2 from the true distribution; because this is a low-probability event, this contributes $2 \cdot o(1) = o(1)$ to the expected error. Otherwise, u is close to h, and the fattened version $\bar{u}$ is similarly close, which lets us apply Lemma 5 to conclude that $\sum_{j<\log^2 n} dev_{j,n}(h, m_{\bar{u},j,n}) \leq o(1)+\sum_{j<\log^2 n} dev_{j,n}(h, m_{h,j,n})$. Corollary 2 says that $\sum_{j<\log^2 n} dev_{j,n}(h, m_{h,j,n})$ essentially lowerbounds the optimal error $opt(h, n)$, which we combine with the previous bound to yield

$$\sum_{j<\log^2 n} dev_{j,n}(h, m_{\bar{u},j,n}) \leq opt(h, n) + o(1).$$

Lemma 2 guarantees that the samples will be faithful except with $o(1)$ probability, which, as above, means that even if these unfaithful cases contribute the maximum possible distance 2 to the ℓ_1 error, the expected contribution from these cases is still $o(1)$, and thus we will assume a faithful set of samples below. Lemmas 6 and 7 imply that for any faithful sample, the error made by Algorithm 2 on attributing those elements seen fewer than $\log^2 n$ times is within $o(1)$ of $\sum_{j<\log^2 n} dev_{j,n}(h, m_{\bar{u},j,n})$, and hence at most $o(1)$ worse than $opt(h, n)$.

Condition 1 of the definition of faithful (Definition 7) implies that all of the elements seen at least $\log^2 n$ times originally had probability at least $\frac{1}{n}(\log^2 n - \log^{1.75} n)$ and that the relative error between the number of times each of these elements is seen and its expectation is thus at most $\log^{-1/4} n$. Thus using the empirical estimate on those elements appearing at least $\log^2 n$ times—as Algorithm 2 does—contributes $O(\log^{-1/4} n)$ total error on these elements. Thus all the sources of error add up to at most $o(1)$ worse than $opt(h, n)$ in expectation, yielding the theorem. □

B. PROOF OF FACT 1

For convenience, we restate Fact 1:

Fact 1 *Given two distributions p_1, p_2 satisfying $R_\tau(p_1, p_2) \leq \epsilon$, there exists a relabeling π of the support of p_2 such that*

$$\sum_i |\max(p_1(i), \tau) - \max(p_2(\pi(i)), \tau)| \leq 2\epsilon.$$

Proof of Fact 1. We relate relative earthmover distance to the minimum L_1 distance between relabled histograms, with a proof that extends to the case where both distances are defined above a cutoff threshold τ. The main idea is to point out that "minimum rearranged" L_1 distance can be expressed in a very similar form to earthmover distance. Given two histograms h_1, h_2, the minimum L_1 distance between any labelings of h_1 and h_2 is clearly the L_1 distance between the labelings where we match up elements of the two histograms in sorted order. Further, this is seen to equal the (regular, not relative) earthmover distance between the histograms h_1 and h_2, where we consider there to be $h_1(x)$ "histogram mass" at each location x (instead of $h_1(x) \cdot x$ "probability mass" as we did for relative earthmover distance), and place extra histogram entries at 0 as needed so the two histograms have the same total mass.

Given this correspondence, consider an optimal *relative* earthmoving scheme between h_1 and h_2, and in particular, consider an arbitrary component of this scheme, where some probability mass α gets moved from some location x in one of the distributions to some location y in the other, at cost $\alpha \log \frac{\max(x,\tau)}{\max(y,\tau)}$, and suppose without loss of generality that $x \geq y$.

We now reinterpret this move in the L_1 sense, translating from moving probability mass to moving histogram mass. In the non-relative earthmover problem, α probability mass at location x corresponds to $\frac{\alpha}{x}$ "histogram mass" at x, which we then move to y at cost $(\max(x, \tau) - \max(y, \tau))\frac{\alpha}{x}$; however, to simulate the relative earthmover scheme, we need the full $\frac{\alpha}{y}$ mass to appear at y, so we move the remaining $\frac{\alpha}{y} - \frac{\alpha}{x}$ mass up from 0, at cost $(\frac{\alpha}{y} - \frac{\alpha}{x})(\max(y, \tau) - \tau)$.

To relate these 3 costs (the original relative earthmover cost, and the two components of the non-relative histogram earthmover cost), we note that if both x and y are less than or equal to τ then all 3 costs are 0. Otherwise, if $x, y > \tau$ then the first component of the histogram cost equals $(1 - \frac{y}{x})\alpha$ and the second is bounded by this, as $(\frac{\alpha}{y} - \frac{\alpha}{x})(\max(y, \tau) - \tau) < (\frac{\alpha}{y} - \frac{\alpha}{x})y = (1 - \frac{y}{x})\alpha$. Further, for the case under consideration where $\tau < y \leq x$, we have $(1 - \frac{y}{x})\alpha \leq \alpha \log \frac{x}{y}$, which equals the relative earthmover cost. Thus the histogram cost in this case is at most twice the relative earthmover cost.

In the remaining case, $y \leq \tau < x$, and the second component of the histogram cost equals 0 because $\max(y, \tau) - \tau = 0$. The first component simplifies as $(\max(x, \tau) -$

$\max(y,\tau))\frac{\alpha}{x} = (x-\tau)\frac{\alpha}{x} = (1-\frac{\tau}{x})\alpha \leq \alpha \log \frac{x}{\tau}$, where this last expression is the relative earthmover cost. Thus in all cases, the histogram cost is at most twice the relative earthmoving cost.

Since the histogram cost was one particular "histogram moving scheme", and as we argued above, the "minimum permuted L_1 distance" is the minimum over all such schemes, we conclude that this L_1 distance is at most twice the relative earthmover distance, as desired. □

C. PROOF OF LEMMA 1

For convenience, we restate the lemma:

Lemma 1 *Given two (possibly generalized) histograms g, h, a number of samples k, and a threshold $\tau \in (0,1]$,*

$$\left| \sum_{x:g(x)\neq 0} (1-(1-x)^k)\cdot g(x) - \sum_{x:h(x)\neq 0} (1-(1-x)^k)\cdot h(x) \right| \leq (0.3(k-1)+1)R_\tau(g,h) + \tau\frac{k}{2}.$$

PROOF. We prove the inequality by considering each step of an earthmoving scheme that transforms g to h, and show that if in one step m probability mass is moved, at τ-truncated relative earthmover cost r, then the sum $\sum_{x:g(x)\neq 0}(1-(1-x)^k)\cdot g(x)$ changes by at most $(1+0.3(k-1))\cdot r + mk\tau$, meaning that an entire earthmoving scheme to transform g into h with total cost $R_\tau(g,h)$ and total mass at most 1 changes the g term on the left hand side into the h term on the left hand side by changing it at most $(1+0.3(k-1))\cdot R_\tau(g,h) + k\tau$.

To prove this we first analyze the region of probability below τ. By the definition of a histogram, m units of probability mass at probability x corresponds to a histogram entry $h(x) = \frac{m}{x}$, and binomial bounds yield $\frac{m}{x}(1-(1-x)^k) \in [km(1-x\frac{k-1}{2}), km]$, which means that when an earthmoving scheme moves m mass in the range $x \in (0,\tau]$, the expression $\frac{m}{x}(1-(1-x)^k)$ changes by at most $km\frac{k-1}{2}\tau$. Thus, summed over the entire earthmoving scheme, where the mass moved sums to at most 1, the change in $\sum_{x:g(x)\neq 0}(1-(1-x)^k)\cdot g(x)$ from changes below probability τ is at most $k\frac{k-1}{2}\tau$.

To bound the remaining term, changes in $\sum_{x:g(x)\neq 0}(1-(1-x)^k)\cdot g(x)$ from changes in probability above τ in the earthmoving scheme, we note that to move probability mass m from probability value x to y costs $m|\log x - \log y|$ in the earthmoving scheme, and changes the sum by

$$\left| \frac{m}{x}\left(1-(1-x)^k\right) - \frac{m}{y}\left(1-(1-y)^k\right) \right|.$$

We bound the ratio of these last two expressions by $1 + 0.3(k-1)$, in order to bound the total contribution of the portion of the earthmoving scheme above probability τ by $(1+0.3(k-1))R_\tau(g,h)$, yielding the desired overall bound.

We thus seek to bound the maximum change in

$$\frac{1}{x}\left(1-(1-x)^k\right)$$

relative to the change in $\log x$ as x changes, namely the maximum ratio of their derivatives, where we add a negative sign since $\frac{1}{x}\left(1-(1-x)^k\right)$ is a decreasing function. Since $\frac{d}{dx}\log x = 1/x$, the ratio of derivatives is

$$-x\frac{d}{dx}\frac{\left(1-(1-x)^k\right)}{x} = \frac{1-(1-x)^{k-1}((k-1)x+1)}{x} \quad (1)$$

Consider the approximation $(1-x)^{k-1} \approx e^{-x(k-1)}$. Taking logarithms of both sides, and using the fact that, for $x \leq \frac{1}{2}$, we have $\log 1 - x \geq -x - x^2$, we have that for $x \leq \frac{1}{2}$ the inequality $(k-1)\log(1-x) \geq -(k-1)(x+x^2)$; exponentiating yields $(1-x)^{k-1} \geq e^{-x(k-1)}\cdot e^{-x^2(k-1)} \geq e^{-x(k-1)}(1-x^2(k-1))$.

Thus for $x \leq \frac{1}{2}$ the ratio of derivatives is bounded as

$$-x\frac{d}{dx}\frac{\left(1-(1-x)^k\right)}{x} \leq \frac{1-(e^{-x(k-1)}(1-x^2(k-1)))((k-1)x+1)}{x} = \frac{1-e^{-x(k-1)}((k-1)x+1)}{x} + \frac{e^{-x(k-1)}x^2(k-1)((k-1)x+1)}{x}.$$

The first term of the right hand side, after dividing by $k-1$, can be reexpressed in terms of $y = x(k-1)$ as $\frac{1-e^{-y}(y+1)}{y}$, which has a global maximum less then 0.3; the second term in the right hand side, after the same variable substitution, equals $e^{-y}y(y+1)$, which has a global maximum less than 1. Thus, for $x \leq \frac{1}{2}$, the absolute value of the ratio of derivatives is bounded as $0.3(k-1)+1$. For $x \geq \frac{1}{2}$, the right hand side of Equation 1 is $\frac{1}{x}$ minus some positive quantity, and is hence at most 2. Since $0.3(k-1)+1 \geq 2$ for any $k \geq 5$, all that remains is to checking the $k = 2,3,4$ cases where $0.3(k-1)+1 < 2$ by hand to confirms that $0.3(k-1)+1$ is in fact a global bound. □