On Sample Reuse Methods for Answering k-wise Statistical Queries

Lev Reyzin and Duan Tu(✉)

University of Illinois at Chicago, Chicago, IL 60607, USA
{lreyzin,dtu4}@uic.edu

Abstract. This paper examines the computational and sample complexity of answering k-wise statistical queries, which were introduced by Felman and Ghazi [9] as a generalization to the standard statistical query model of Kearns [11]. In particular, our paper studies two sample reuse schemes: (1) reusing independent "pseudo-samples" for adaptive queries and (2) reusing dependent k-wise samples for non-adaptive queries. Comparing to a baseline non-reuse strategy, we show that the first reuse method offers a trade-off between k, the arity of the query, and M, the total number of queries to be answered. We also show that the second reuse method performs no worse than the baseline, and possibly better, from the perspective of variance reduction.

Keywords: Statistical query learning · Sample reuse · Differential privacy

1 Introduction

In this paper we study the sample complexity of answering M different k-wise statistical queries. k-wise statistical queries are a generalization of the statistical query model introduced by Kearns [11] and widely studied thereafter [3,4]. While unary statistical queries look at the expectation of a function $q : X \to \{0,1\}$ from one data point onto a binary range, k-wise queries $q : X^k \to \{0,1\}$ use samples of size $k \geq 1$. The importance of being able to answer k-wise queries for larger values of k is illustrated by Felman and Ghazi [9], who showed that as k increases, strictly more problems can be solved using k-wise queries.

Known methods for answering statistical queries (SQs) include strategies ranging from straightforward sampling methods to more involved approaches involving principled sample reuse from the perspective of adaptive data analysis [5]. In this paper we analyze these varying approaches for the more general k-wise case and find a trade-off in which method is the best depending on the relative values of k, the arity of the query, and M, the total number of queries to be answered. We also give a different view for a known strategy for sample reuse, and show that it performs no worse than the original non-reuse strategy, and possibly better.

R. P. Barneva et al. (Eds.): IWCIA 2024/ISAIM 2024, LNCS 14494, pp. 136–149, 2024.
https://doi.org/10.1007/978-3-031-63735-3_8

There are two natural ways to improve beyond the straightforward sampling approach: (1) reuse independent *pseudo-samples* for adaptive queries and (2) reuse dependent k-wise samples for non-adaptive queries. In the first method, each pseudo-sample is composed by drawing k i.i.d points from the sample domain, so pseudo-samples are mutually independent and identically distributed according to the product distribution. Intuitively, the first method reuses the same set of i.i.d. pseudo-samples among all queries, while the second method draws a new set of points for each query and takes all possible size k subsets to create dependent k-wise samples. It is worth noting that we are only concerned with the case of adaptive queries for method (1), since results for non-adaptive queries are already given by VC theory.

In the remaining parts of the paper, we shall provide rigorous statements of definitions and useful technical tools in Preliminaries, introduce the naive sampling approach in Baseline simulation of an SQ oracle, and then discuss results of the two sample reuse methods in the last two sections.

2 Preliminaries

We first give the definition of a k-wise SQ oracle.

Definition 1 (Feldman and Ghazi [9]). *Let $\mathcal{D}$ be a distribution over a domain X and $\tau > 0$. A k-wise statistical query oracle $\mathrm{STAT}_{\mathcal{D}}^{(k)}(\phi, \tau)$ is an oracle that given as input any query function $\phi : X^k \to \{0, 1\}$ and a value τ, returns some value v such that $|v - \mathbb{E}_{\mathbf{x} \sim \mathcal{D}^k}[\phi(\mathbf{x})]| \leq \tau$.*

The goal of statistical query learning, as originally defined by Kearns [11], is to learn a target class of functions efficiently, achieving PAC guarantees while using the SQ oracle instead of labeled examples. In this case, to efficiently SQ-learn a function class, one wants to make a polynomial number of calls to the $\mathrm{STAT}_{\mathcal{D}}^{(k)}$ oracle, using tolerances τ such that $\frac{1}{\tau}$ is polynomially bounded away from 0, and using query functions ϕ evaluable in polynomial time. For detailed definitions and more about the SQ model, see Reyzin [12].

Next, we are interested in the sensitivity of query functions that are fed to the SQ oracle. The ℓ_1-sensitivity of a query measures the magnitude by which perturbing a single data point can change the query output in the worst case. It is an important parameter in determining the algorithm's required accuracy when answering queries.

Definition 2 (Dwork and Roth [7]). *The ℓ_1-sensitivity of a function $f : \mathbb{N}^{|X|} \to \mathbb{R}$ is*

$$\begin{aligned}\Delta f &= \max_{\substack{x,y\in\mathbb{N}^{|X|}\\ \|x-y\|_1=1}} \|f(x) - f(y)\|_1 \\ &= \max_{\substack{x,y\in\mathbb{N}^{|X|}\\ \|x-y\|_1=1}} \sum_{i=1}^{|X|} |f(x_i) - f(y_i)|.\end{aligned}$$

One main technique we use to study reusing pseudo-samples among adaptive queries is the Transfer Theorem developed in Bassily et al. [2]. The theorem says a differentially private learner that is accurate with respect to its samples generalizes to the population from which the samples were drawn. Bassily et al. [2] uses the term "max-KL stability" to refer to the differential privacy model of Dwork et al. [6], emphasizing it as one of the various notions of stability in machine learning. We state the definition of a differentially private learner and the Transfer Theorem as follows.

Definition 3 (Dwork and Roth [7]). *A randomized algorithm $\mathcal{M}$ with domain $\mathbb{N}^{|\mathcal{X}|}$ is (ε, δ)-differentially private if for all $\mathcal{S} \subseteq \text{Range}(\mathcal{M})$ and for all $x, y \in \mathbb{N}^{|\mathcal{X}|}$ such that $\|x - y\|_1 \leq 1$:*

$$\mathbb{P}[\mathcal{M}(x) \in \mathcal{S}] \leq \exp(\varepsilon)\Pr[\mathcal{M}(y) \in \mathcal{S}] + \delta,$$

where the probability space is over the randomness of $\mathcal{M}$.

Before introducing the Transfer Theorem (Lemma 1), let us define what it means for an algorithm to be accurate with respect to a collection of samples and with respect to a population.

Definition 4 (Bassily et al. [2]). *A mechanism $\mathcal{M}$ is (α, β)-accurate with respect to samples of size n from $\mathcal{X}$ for M adaptively chosen queries from Φ, if for every adversary $\mathcal{A}$, which gives output $(a_1, \cdots, a_M)$,*

$$\mathbb{P}\left[\max_{j\in[M]} \left|a_j - \frac{1}{n}\sum_{i\in[n]} \phi_j(\mathbf{x}_i)\right| \leq \alpha\right] \geq 1 - \beta.$$

A mechanism $\mathcal{M}$ is (α, β)-accurate with respect to the population for M adaptively chosen queries from Φ given n samples $\mathbf{x} \in \mathcal{X}$, if for every adversary $\mathcal{A}$, which gives output $(a_1, \cdots, a_M)$,

$$\mathbb{P}\left[\max_{j\in[M]} \left|a_j - \mathbb{E}\,\phi_j(\mathbf{x})\right| \leq \alpha\right] \geq 1 - \beta.$$

Now we are ready to state the Transfer Theorem.

Lemma 1 (Transfer Theorem, by Bassily et al. [2]). *Let Φ be a family of Δ-sensitive queries on X^k. Assume that for some $\alpha, \beta \in (0, 0.1)$, an algorithm $\mathcal{A}$ is*

1. *$(\varepsilon' = \alpha/64\Delta n, \delta' = \alpha\beta/32\Delta n)$-max-KL stable for M adaptively chosen queries from Φ and*
2. *$(\alpha' = \alpha/8, \beta' = \alpha\beta/16\Delta n)$-accurate with respect to its n samples from X^k for M adaptively chosen queries from Φ.*

Then $\mathcal{A}$ is (α, β)-accurate with respect to the population for M adaptively chosen queries from Φ given n samples from X^k.

One can achieve the privacy requirement (via max-KL stability) in the Transfer Theorem through the *Laplace mechanism*. Recall that Δf denotes the ℓ_1-sensitivity of function f. Recall that the Laplace Distribution centered at 0 with scale parameter b has probability density function

$$\mathrm{Lap}(b) = \frac{1}{2b} \exp\left(-\frac{|x|}{b}\right).$$

Definition 5 (Dwork and Roth [7]). *Given any function $f : \mathbb{N}^{|\mathcal{X}|} \to \mathbb{R}^k$, the Laplace mechanism is defined as:*

$$\mathcal{M}_L(x, f(\cdot), \varepsilon) = f(x) + (Y_1, \cdots, Y_k),$$

where Y_i are i.i.d. random variables drawn from the Laplace Distribution of scale parameter $\Delta f/\varepsilon$, denoted as $\mathrm{Lap}(\Delta f/\varepsilon)$.

Let $\mathcal{A}$ be an algorithm that calculates the average of a function $\phi : X^k \to \{0, 1\}$ over n samples. Suppose ϕ has ℓ_1-sensitivity Δ. After computing the true average value a, the Laplace mechanism outputs $v = a + y$ where $y \sim \mathrm{Lap}(\Delta/\varepsilon)$ is drawn from the Laplace distribution with scale parameter Δ/ε.

Lemma 2 (Dwork and Roth [7]). *For any function $f : \mathbb{N}^{|\mathcal{X}|} \to \mathbb{R}^k$, the Laplace mechanism guarantees $(\varepsilon, 0)$-differential privacy.*

It is easy to come up with a high probability bound on the amount of noise added by the Laplace mechanism.

Lemma 3 (Dwork and Roth [7]). *Let $f : \mathbb{N}^{|\mathcal{X}|} \to \mathbb{R}^k$. Let the output of the Laplace mechanism be $y = \mathcal{M}_L(x, f(\cdot), \varepsilon)$. Then $\forall \delta \in (0, 1]$:*

$$\mathbb{P}\left[\|f(x) - y\|_\infty \geq \ln\left(\frac{k}{\delta}\right) \cdot \left(\frac{\Delta f}{\varepsilon}\right)\right] \leq \delta.$$

3 Baseline Simulation of an SQ Oracle

In this section we discuss the sample complexity of learning with k-wise SQs without sample reuse. An algorithm simulates a k-wise SQ oracle by taking empirical averages. This simulation is extended from the one given by Kearns [11] for unary SQ oracles. In the k-wise case, to each query function ϕ_i the learner feeds it a fresh batch of $\tilde{n}$ i.i.d. pseudo-samples $S_i = \{\mathbf{x}_1, \cdots, \mathbf{x}_{\tilde{n}}\}$, where each pseudo-sample $\mathbf{x}_j = (x_{j_1}, \cdots x_{j_k})$ consists of k sample points. Then the learner computes the empirical average of $\phi_i(\mathbf{x})$ over the set of pseudo-sample S_i. With high probability, the empirical average will fall within the amount of tolerance allowed by the SQ oracle from the true expectation of ϕ_i, thanks to concentration inequalities. Proposition 1 provides the quantitative result of this baseline approach.

Proposition 1. *Suppose there exists an SQ learner that makes M k-wise statistical queries of tolerance τ to learn over a class $\mathcal{C}$, then there exists a simulation algorithm, which does not reuse any samples, for which a set of i.i.d. samples of size*

$$n = O\left(k\frac{M}{\tau^2}\log\left(\frac{M}{\delta}\right)\right)$$

is sufficient to PAC learn $\mathcal{C}$ with error bounded by ε and probability of failure bounded by δ.

Proof. We take the specified number of samples and partition them into

$$\tilde{n} = \frac{n}{Mk} = O\left(\frac{1}{\tau^2}\log\left(\frac{M}{\delta}\right)\right)$$

i.i.d. pseudo-samples for each query function ϕ_i. The Hoeffding bound guarantees that the empirical average over $\tilde{n}$ pseudo-samples falls within $\pm\tau$ from $\mathbb{E}[\phi_i(\mathbf{x})]$ with probability $\geq 1 - \frac{\delta}{M}$. Then apply the union bound and we obtain that with probability $\geq 1 - \delta$, the empirical average falls within $\pm\tau$ from the true expectation for all queries. Hence we have successfully simulated the k-wise SQ learner with high probability, fulfilling the PAC requirements. □

4 Independent Pseudo-Samples for Adaptive Queries

In this section we discuss the reuse of independent pseudo-samples for adaptive queries. Suppose there exists a k-wise SQ learner that efficiently SQ-learns a function class by asking M adaptive k-wise queries $\phi_1, \cdots, \phi_M$. Similar to the baseline case, our algorithm (Algorithm 1) simulates a k-wise SQ oracle through taking empirical averages. However, what is different from the baseline case is that Algorithm 1 partitions the set of n samples $x \sim \mathcal{D}$ into $\tilde{n} = n/k$ parts to create $\tilde{n}$ i.i.d. pseudo-samples $\mathbf{x} = (x_1, \cdots, x_k) \sim \mathcal{D}^k$. It then reuses the same set of pseudo-samples among all queries when taking their empirical averages.

Now we state our main sample complexity result. Theorem 1 provides the optimal sample complexity for an algorithm that reuses the same set of independent pseudo-samples while answering adaptive queries.

Theorem 1. *Suppose there exists an SQ learner that makes M k-wise statistical queries of tolerance τ to learn a class $\mathcal{C}$, then there exists a simulation algorithm, which reuses independent pseudo-samples among the M queries, for which a set of i.i.d. samples of size*

$$\begin{aligned} n &= O\left(\frac{k^2\sqrt{M}}{\tau^2}\log\left(\max\left\{M, \frac{k}{\tau}\right\}\frac{1}{\delta}\right)\right) \\ &= O\left(\frac{k^2\sqrt{M}}{\tau^2}\log\left(\frac{Mk}{\tau\delta}\right)\right) \end{aligned}$$

is sufficient to PAC-learn $\mathcal{C}$ with error bounded by ε and probability of failure bounded by δ.

Comparing Theorem 1 to the naive bound in Proposition 1, we observe an interesting trade-off between the arity of the query k, and the total number of queries M. The trade-off suggests that only when a learner uses a large amount of short queries ($k < \sqrt{M}$) is it worth to reuse pseudo-samples.

It is worth noting that Algorithm 1 is specific to k-wise statistical queries and it differs from approaches that work for low-sensitivity queries in general. In addition to having low sensitivity, statistical queries and their k-wise generalizations have the additional property that they can be evaluated on k points at a time, and are therefore amenable to sampling techniques, which can produce potential speedups (see Fish, Reyzin, and Rubinstein [10]). This allows us to evaluate our queries on pseudo-samples, each of which consists of k sample points.

Algorithm 1. Reusing Independent Pseudo-samples for Adaptive Queries

Inputs. Sample points $x \in X$ and k-wise Statistical Queries $\phi_1, \cdots, \phi_M$, where $\phi_i : X^k \to \{0,1\}$ for all $i \in [M]$.
Outputs. $v \in \mathbb{R}^M$.

1: Draw $O\left(\frac{k^2\sqrt{M}}{\tau^2}\log(\frac{Mk}{\tau\delta})\right)$ i.i.d. sample points $x \sim \mathcal{D}$. Create $\tilde{n} = \frac{n}{k}$ copies of i.i.d. pseudo-samples $\mathbf{x}_j = (x_{j_1}, \cdots, x_{j_k}) \sim \mathcal{D}^k$, where $j = 1, \cdots, \tilde{n}$.
2: **for** $i = 1, \cdots, M$ **do**
3: **for** $j = 1, \cdots, \tilde{n}$ **do**
4: $a_{ij} \leftarrow \phi_i(\mathbf{x}_j)$
5: **end for**
6: Draw Laplace noise $y \sim \text{Lap}\left(\frac{128k^2\sqrt{M}}{\tau n}\right)$.
7: $v_i \leftarrow \left(\frac{1}{\tilde{n}}\sum_{j=1}^{\tilde{n}} a_{ij}\right) + y$
8: **end for**
9: $v \leftarrow (v_1, \cdots, v_M)$

The sample complexity achieved by Algorithm 1 is no worse than the bound $\tilde{O}\left(\frac{\sqrt{M}}{\tau^2}\right)$ known for general low-sensitivity queries in Bassily et al. [2]. Their approach for general low-sensitivity queries takes time $\text{poly}(n, \log|X|)$ per oracle call, whereas our Algorithm 1 runs in $\text{poly}(k, \log|X|)$ time per call to the oracle (assuming polynomial-time evaluability of the respective queries). This can potentially create a dramatic improvement in running time, since the straightforward non-sampling approach for exactly evaluating a k-wise query on a sample of n points would be to evaluate it on all k-point subsets, which is indeed polynomial in n but exponential in k. In fact, we go on to analyze that particular approach towards the end of the paper.

In the remaining parts of this section, we first discuss a couple technical tools used to prove Theorem 1 and then we give the proof itself.

4.1 Privacy Composition

To ensure that the simulation generalizes to the sample distribution, we apply Lemma 1 (Transfer Theorem), which demands the algorithm be differentially private. The algorithm composes multiple query functions, so in order to achieve the required level of privacy overall, we need to use results on privacy composition to figure out what level of privacy is required for each individual query.

There are two well-known bounds on the privacy of query composition: simple composition and advanced composition. Simple composition provides the elementary bound that, when a learner uses independent queries, its privacy equals to the sum of privacy of all queries. Advanced composition deals with the more complicated situation, one where the learner poses adaptive queries to the same database repeatedly. We shall see that under appropriate choice of parameters, advanced composition offers tighter privacy bound than simple composition (by a factor of $\sqrt{M}$). The exact statements of the two composition results are provided by Lemma 4 and Lemma 5.

Lemma 4 (Simple Composition, as presented in Dwork and Roth [7]). *Let $\mathcal{A}_i : X^k \to \{0,1\}$ be an $(\varepsilon_i, \delta_i)$-differentially private algorithm for $i = 1, \cdots, M$. Then $\mathcal{A} = (\mathcal{A}_1, \cdots, \mathcal{A}_M)$ is $\left(\sum_{i=1}^M \varepsilon_i, \sum_{i=1}^M \delta_i\right)$-differentially private.*

Lemma 5 (Advanced Composition, by Dwork, Rothblum, and Vadhan [8]). *For all $\varepsilon, \delta, \delta' \geq 0$, the class of (ε, δ)-deferentially private mechanisms satisfies $(\varepsilon', M\delta + \delta')$-differential privacy under M-fold adaptive composition for*

$$\varepsilon' = \varepsilon\sqrt{2M\ln(1/\delta')} + M\varepsilon(e^\varepsilon - 1).$$

Observe that

$$\varepsilon' \leq \varepsilon\sqrt{2M\ln(1/\delta')} + M\varepsilon^2 = O\left(\varepsilon\sqrt{M\ln(1/\delta')}\right)$$

when ε is small. By choosing δ' small, say $\delta' = 1/e$, it can be shown that the M-fold adaptive composition satisfies $(\varepsilon\sqrt{M}, \delta M)$ - differential privacy [2].

Theorem 1 uses advanced composition of privacy. It is important to mention that advanced composition is necessary when analyzing pseudo-sample reuse. Since the algorithm uses adaptive queries, it needs to be strict when budgeting the privacy level for each query. Otherwise, an excess amount of Laplace noise would need to be added, which will overturn the effect of sample reuse. As shown in Theorem 2, if the algorithm composed privacy of the queries as if they were independent, the resulting sample complexity is actually worse than the baseline bound.

Theorem 2. *Under the setting of Theorem 1, except that suppose the simulation algorithm treats the M queries as if they were independent and calculates their overall privacy through simple composition, a set of i.i.d. samples of size*

$$\begin{aligned} n &= O\left(\frac{k^2 M}{\tau^2}\log\left(\max\left\{M, \frac{k}{\tau}\right\}\frac{1}{\delta}\right)\right) \\ &= O\left(\frac{k^2 M}{\tau^2}\log\left(\frac{Mk}{\tau\delta}\right)\right) \end{aligned}$$

is sufficient to PAC learn $\mathcal{C}$ with error bounded by ε and probability of failure bounded by δ.

We omit the proof for Theorem 2 since it closely resembles that of Theorem 1, with the only difference being the privacy composition calculations.

4.2 Laplace Mechanism

Now that we know to use advanced composition, let us consider how to achieve the desired level of privacy for each query function. As suggested by Lemma 2, we adopt Laplace mechanism, the standard technique that offers privacy guarantee for algorithms.

For each ϕ_i, Algorithm 1 outputs $v_i = a_i + y$, where a_i is the empirical average of ϕ_i over a large set of pseudo-samples and y is a small Laplace noise parameter. There are two key considerations when choosing the parameters. First, the sample set needs to be large enough so that the empirical average is close to the true expectation with high probability. Second, the Laplace noise needs to be small enough so that it does not steer the empirical average away from the expected average too far, but in the meantime still large enough to maintain privacy.

Using Lemma 2, we choose $y \sim \text{Lap}(\Delta \cdot \frac{128k}{\tau})$, which preserves $(\frac{\tau}{128k}, 0)$-differential privacy for each query, surpassing the requirement of the Transfer Theorem (Lemma 1). Here Δ is the ℓ_1-sensitivity of the empirical average of ϕ over all pseudo-samples. In an attempt to simplify the writing, we abuse notation and use $\Delta\phi$ to represent the aforementioned ℓ_1-sensitivity.

Proposition 2. *The ℓ_1-sensitivity of the empirical average of $\phi : X^k \to \{0, 1\}$ is $\Delta\phi \leq \frac{k}{n}$.*

Proof. Among all pseudo-samples $\mathbf{x}_i \in S$ and $\mathbf{x}'_i \in S'$ where $i = 1, \cdots, \tilde{n}$, exactly one pair is different $\mathbf{x}_j \neq \mathbf{x}'_j$. Then $|\phi(x_i) - \phi(x'_i)| = 0$ for all $i \neq j$, while $|\phi(x_j) - \phi(x'_j)| \leq 1$. Therefore,

$$\Delta\phi = \max_{\substack{S,S' \subseteq X^k \\ s.t.\ \|S - S'\|_1 = 1}} \left\| \frac{1}{\tilde{n}} \sum_{i=1}^{\tilde{n}} \left(\phi(\mathbf{x}_i) - \phi(\mathbf{x}'_i) \right) \right\|_1,$$

which can trivially be bounded as $\Delta\phi \leq 1/\tilde{n} = k/n$. □

Proof of Theorem 1

Given an efficient k-wise SQ learner that learns $\mathcal{C}$ approximately correct (to an error ε), the empirical average simulation wishes to mimic the learner's query outputs with high probability. In the language of the Transfer Theorem (Lemma 1), that is to say the simulator needs to be (τ, δ)-accurate with respect to the population. We prove Theorem 1 using the Transfer Theorem.

*Proof (*Proof of Theorem 1**).** Given the total allowed error of τ, we allocate $\tau/2$ to the empirical average and $\tau/2$ to the added Laplace noise. We first analyze the empirical average. To achieve $(\tau/2, \delta)$-accuracy with respect to the population for M adaptively chosen queries, the Transfer Theorem demands

(i) the simulation is $\left(\frac{\tau}{128k}, \frac{\tau\delta}{64k}\right)$-differentially private for M adaptive queries,
(ii) the simulation is $\left(\frac{\tau}{16}, \frac{\tau\delta}{32k}\right)$-accurate with respect to n samples for M adaptive queries.

To satisfy (i), we adopt advanced composition. According to Lemma 5, each of the M queries needs to be $\left(\frac{\tau}{128k\sqrt{M}}, \frac{\tau\delta}{64kM}\right)$-differentially private to obtain the composed privacy stated in (i). We know each query has ℓ_1-sensitivity k/n through Lemma 2. Then following the standard technique stated in Lemma 2, we add Laplace noise of scale $\frac{128k^2\sqrt{M}}{\tau n}$ to each query average, which achieves $\left(\frac{\tau}{128k\sqrt{M}}, 0\right)$-differential privacy, surpassing the needed amount. Lemma 3 verifies that the added Laplace noise is bounded above by $\frac{128k^2\sqrt{M}}{\tau n} \log \frac{2M}{\delta}$ with probability $\geq 1 - \frac{\delta}{2M}$. In order to restrict the amount of Laplace noise within $\tau/2$ with high probability, we ask that

$$\frac{128k^2\sqrt{M}}{\tau n} \log \frac{2M}{\delta} \leq \frac{\tau}{2},$$

which implies

$$n = O\left(\frac{k^2\sqrt{M}}{\tau^2} \log \frac{M}{\delta} \right) \tag{1}$$

is sufficient. Now let us consider (ii). It suffices to show that for all queries ϕ_i, the simulator's output a_i satisfies

$$\mathbb{P}\left[|\mathrm{err}_{\mathbf{x}}(\phi_i, a_i)| \leq \frac{\tau}{16} \right] \geq 1 - \frac{\tau\delta}{32k}.$$

We know that

$$a_i = \frac{1}{\tilde{n}} \sum_{j=1}^{\tilde{n}} \phi_i(\mathbf{x}_j) + \mathrm{Lap}\left(\frac{128k\sqrt{M}}{\tau\tilde{n}}\right),$$

so for all i,

$$\begin{aligned} |\mathrm{err}_{\mathbf{x}}(\phi_i, a_i)| &= |a_i - \phi_i(\mathbf{x})| \\ &= \left| a_i - \frac{1}{\tilde{n}} \sum_{j=1}^{\tilde{n}} \phi_i(x_j) \right| \\ &= \mathrm{Lap}\left(\frac{128k\sqrt{M}}{\tau\tilde{n}}\right). \end{aligned}$$

According to Lemma 3, it is easy to verify that with probability $\geq 1 - \frac{\tau\delta}{32k}$, the Laplace noise of scale $\frac{128k\sqrt{M}}{\tau\tilde{n}}$ is $O(\frac{k^2\sqrt{M}}{\tau n} \log \frac{k}{\tau\delta})$. To satisfy (ii), we ask that $\frac{128k^2\sqrt{M}}{\tau n} \log \frac{32k}{\tau\delta} \leq \frac{\tau}{16}$, which implies

$$n = O\left(\frac{k^2\sqrt{M}}{\tau^2} \log \frac{k}{\tau\delta}\right) \tag{2}$$

is sufficient. Combining inequalities (1) and (2), we get

$$\begin{aligned} n =& O\left(\max\left\{\frac{k^2\sqrt{M}}{\tau^2} \log \frac{M}{\delta}, \frac{k^2\sqrt{M}}{\tau^2} \log \frac{k}{\tau\delta}\right\}\right) \\ =& O\left(\frac{k^2\sqrt{M}}{\tau^2} \log\left(\max\left\{M, \frac{k}{\tau}\right\} \frac{1}{\delta}\right)\right) \\ =& O\left(\frac{k^2\sqrt{M}}{\tau^2} \log\left(\frac{Mk}{\tau\delta}\right)\right), \end{aligned}$$

which completes the proof. □

5 Dependent k-wise Samples for Non-adaptive Queries

Now we examine the second reuse method. Algorithm 2 draws n i.i.d. sample points $x \sim X$ and partitions them into M equal parts, $S_1, \cdots, S_M$, to be used by M queries. Denote the size of each part as $|S_i| = \hat{n}$, so the total number of samples is $n = M\hat{n}$. For each query, the algorithm calculates its empirical average over $\binom{\hat{n}}{k}$ k-wise samples, which are generated by taking all size k subsets of S_i.

Algorithm 2. Dependent k-wise Samples for Non-adaptive Queries

Inputs. Sample points $x \in X$ and k-wise Statistical Queries $\phi_i : X^k \to \{0,1\}$, where $i = 1, \cdots, M$.

Outputs. $v = (v_1, \cdots, v_M) \in \mathbb{R}^M$.

1: Draw n i.i.d. sample points $x \sim \mathcal{D}$ and partition them into M equal parts $S_1, \cdots, S_M$, where $|S_i| = \hat{n}$.
2: **for** $i = 1, \cdots, M$ **do**
3: Take all size k subsets of S_i to create k-wise samples $\mathbf{x}_j = (x_{j_1}, \cdots, x_{j_k})$, where $j = 1, \cdots, \binom{\hat{\mathbf{n}}}{\mathbf{k}}$.
4: Compute the empirical average of ϕ_i
$v_i \leftarrow \frac{1}{\binom{\hat{n}}{k}} \sum_{j=1}^{\binom{\hat{n}}{k}} \phi_i(\mathbf{x}_j)$.
5: **end for**
6: $v \leftarrow (v_1, \cdots, v_M)$.

In contrast to creating independent pseudo-samples, Algorithm 2 uses all k-subsets of the provided sample set, yielding additional k-wise samples, although it fails to maintain their independence since each point contributes to $(k-1)$ samples.

We can analyze these dependent k-wise samples from the perspective of a hypergraph. In the language of hypergraphs, we can think of each sample point as a vertex and each k-wise sample as a k-hyperedge. The learner is given K_n^k, a complete hypergraph on n vertices, whose hyperedges contain k vertices (assuming k divides n). The learner uses k-hyperedges as inputs to the queries. Notice that the hyperedges are not independent with each other. Fortunately, we can bypass the hyperedge dependency through Baranyai's Theorem.

Theorem 3 (Baranyai [1]). *Every K_n^k hypergraph decomposes into a disjoint collection of 1-factors.*

Recall that a 1-factor is a set of hyperedges that touch each vertex in K_n^k exactly once. Intuitively, we can think of a 1-factor as a perfect matching. With the guarantee of decomposition given by Baranyai's Theorem, we are able to interpret the collection of dependent hyperedges as a set of perfect matchings. Although these matchings are dependent on one another, they each contain independent hyperedges within themselves. Figure 1 gives an example of when $n = 6$ and $k = 2$. As shown by Fig. 1, K_6^2 can be decomposed into a disjoint union of 1-factors, each of which consists of three mutually independent edges.

How well do dependent k-wise samples perform when we use them to estimate the expected value through empirical average? In each 1-factor, the independent hyperedges act like pseudo-samples introduced in Algorithm 1. Accordingly, in Theorem 4 we provide accuracy bounds of dependent k-wise samples by comparing its variance to that of independent pseudo-samples.

To set up Theorem 4, let $\phi : X^k \to \{0,1\}$ be a k-wise statistical query, S be a set of samples $x \sim \mathcal{D}$, and suppose $|S| = n$, where k divides n. Let Y_p, Y_a be random variables that represent the empirical average of ϕ under the two

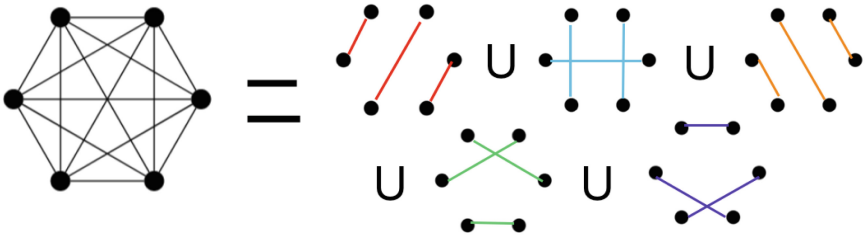

Fig. 1. An illustration of a decomposition of K_6^2 into five disjoint perfect matchings

sampling schemes respectively: creating n/k independent pseudo-samples and taking all $\binom{n}{k}$ k-subsets of S. The expected value of Y_p and Y_a both equal to $\mathbb{E}(\phi)$.

Theorem 4. *The variance of Y_p and Y_a satisfy*

$$\frac{1}{\binom{n-1}{k-1}}\mathrm{Var}(Y_p) \leq \mathrm{Var}(Y_a) \leq \mathrm{Var}(Y_p).$$

Proof. We first study the upper bound. Construct a complete hypergraph K_n^k with the given n sample points. With guarantee from Baranyai's theorem, we can decompose K_n^k into 1-factors $G_1, \cdots, G_m$, where $m = \binom{n-1}{k-1}$. Each G_i contains n/k i.i.d. hyperedges of length k. The vertices in these i.i.d. hyperedges form i.i.d. pseudo-samples used in Algorithm 1. Let Y_{G_i} be a random variable that represents the empirical average of ϕ over pseudo-samples in G_i. By previous analysis, we know for all $i = 1, \cdots, \binom{n-1}{k-1}$,

$$\mathrm{Var}(Y_p) = \mathrm{Var}(Y_{G_i}).$$

Observe that taking an empirical average over all $\binom{n}{k}$ hyperedges in K_n^k is equivalent to taking an average of all the empirical averages over $G_1, \cdots, G_m$. Therefore,

$$\mathrm{Var}(Y_a) = \mathrm{Var}\left(\frac{1}{m}\sum_{i=1}^{m} Y_{G_i}\right).$$

Since Y_{G_i} are i.i.d. random variables, we can denote $\mathrm{Var}(Y_{G_i}) = \sigma^2$ for all $i = 1, \cdots, m$. Then we can prove the upper bound

$$\begin{aligned}
\mathrm{Var}(Y_a) &= \mathrm{Var}\left(\frac{1}{m}\sum_{i=1}^{m} Y_{G_i}\right) \\
&= \frac{1}{m^2}\left(\sum_{i}\mathrm{Var}(Y_{G_i}) + \sum_{i\neq j}\mathrm{Cov}(Y_{G_i}, Y_{G_j})\right) \\
&\leq \frac{1}{m^2}\left(m\sigma^2 + (m^2-m)\sqrt{\sigma^2\sigma^2}\right) \\
&= \sigma^2.
\end{aligned}$$

The inequality uses the well-known fact that for any two random variables X_i, X_j,

$$\mathrm{Cov}(X_i, X_j) \le \sqrt{\mathrm{Var}(X_i)\mathrm{Var}(X_j)}.$$

The lower bound follows similar reasoning.

$$\begin{aligned}\mathrm{Var}(Y_a) &= \frac{1}{m^2}\left(\sum_i \mathrm{Var}(Y_{G_i}) + \sum_{i\neq j}\mathrm{Cov}(Y_{G_i}, Y_{G_j})\right)\\ &\ge \frac{1}{m^2}\sum_i^m \sigma^2\\ &= \frac{\sigma^2}{m}.\end{aligned}$$

This completes the proof. □

Therefore, we find that while Algorithm 2 may take longer to run than baseline sampling (due to its exponential dependence on k), the variance in its estimates will never be worse, which should lead to an improved (or at least not degraded) sample complexity.

Our analysis in Algorithm 2 corresponds to exact evaluation of k-wise statistical queries. If we added, e.g. Laplace noise, to add stability to Algorithm 2, this would be closer to the approach of Bassily et al. [2] for adaptive data reuse. As it turns out, we can achieve the same bound of $\tilde{O}(\frac{k^2\sqrt{M}}{\tau^2})$ in Algorithm 1 but at lower computational cost. This gives an improvement over the work of Bassily et al. [2] for the case of k-wise statistical queries. Their approach, however, is more general.

Acknowledgments. This work was supported in part by National Science Foundation grant ECCS-2217023. We thank an anonymous reviewer of this paper for detailed and helpful comments.

References

1. Baranyai, Z.: On the factrization of the complete uniform hypergraphs. Infinite Finite Sets 1 (1974). https://cir.nii.ac.jp/crid/1571417125147391744
2. Bassily, R., Nissim, K., Smith, A., Steinke, T., Stemmer, U., Ullman, J.: Algorithmic stability for adaptive data analysis. SIAM J. Comput. **50**(3), STOC16-377–STOC16-405 (2021). https://doi.org/10.1137/16M1103646
3. Blum, A., Furst, M.L., Jackson, J.C., Kearns, M.J., Mansour, Y., Rudich, S.: Weakly learning DNF and characterizing statistical query learning using fourier analysis. In: Leighton, F.T., Goodrich, M.T. (eds.) Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing, 23–25 May 1994, Montréal, Québec, Canada, pp. 253–262. ACM (1994). https://doi.org/10.1145/195058.195147
4. Blum, A., Kalai, A., Wasserman, H.: Noise-tolerant learning, the parity problem, and the statistical query model. J. ACM **50**(4), 506–519 (2003)

5. Dwork, C., Feldman, V., Hardt, M., Pitassi, T., Reingold, O., Roth, A.: The reusable holdout: preserving validity in adaptive data analysis. Science **349**(6248), 636–638 (2015)
6. Dwork, C., McSherry, F., Nissim, K., Smith, A.D.: Calibrating noise to sensitivity in private data analysis. J. Priv. Confid. **7**(3), 17–51 (2016). https://doi.org/10.29012/jpc.v7i3.405
7. Dwork, C., Roth, A.: The algorithmic foundations of differential privacy. Found. Trends Theor. Comput. Sci. **9**(3–4), 211–407 (2014). https://doi.org/10.1561/0400000042
8. Dwork, C., Rothblum, G.N., Vadhan, S.P.: Boosting and differential privacy. In: 51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, Las Vegas, Nevada, USA, pp. 51–60. IEEE Computer Society (2010)
9. Feldman, V., Ghazi, B.: On the power of learning from k-wise queries. In: Papadimitriou, C.H. (ed.) 8th Innovations in Theoretical Computer Science Conference, ITCS 2017, 9–11 January 2017, Berkeley, CA, USA. LIPIcs, vol. 67, pp. 41:1–41:32. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2017)
10. Fish, B., Reyzin, L., Rubinstein, B.I.P.: Sampling without compromising accuracy in adaptive data analysis. In: Kontorovich, A., Neu, G. (eds.) Algorithmic Learning Theory, ALT 2020, 8–11 February 2020, San Diego, CA, USA. Proceedings of Machine Learning Research, vol. 117, pp. 297–318. PMLR (2020)
11. Kearns, M.J.: Efficient noise-tolerant learning from statistical queries. J. ACM **45**(6), 983–1006 (1998)
12. Reyzin, L.: Statistical queries and statistical algorithms: foundations and applications. CoRR abs/2004.00557 (2020). https://arxiv.org/abs/2004.00557