Quantum query complexity of entropy estimation

Tongyang Li∗ Xiaodi Wu†

Department of Computer Science, Institute for Advanced Computer Studies, and Joint Center for Quantum Information and Computer Science, University of Maryland, USA

Abstract

Estimation of Shannon and Rényi entropies of unknown discrete distributions is a fundamental problem in statistical property testing and an active research topic in both theoretical computer science and information theory. Tight bounds of the number of samples to estimate these entropies have been established in the classical setting, while little is known about their quantum counterparts. In this paper, we give the first quantum algorithms for estimating α-Rényi entropies (Shannon entropy being 1-Rényi entropy). In particular, we demonstrate a quadratic quantum speedup for Shannon entropy estimation and a generic quantum speedup for α-Rényi entropy estimation for all α > 0, including a tight bound for the collision-entropy (2-Rényi entropy) and an analysis for the min-entropy case (i.e., α = +∞). Moreover, we complement our results with quantum lower bounds on α-Rényi entropy estimation for all α > 0.

Our approach is inspired by the pioneering work of Bravyi, Harrow, and Hassidim (BHH) [11] on quantum algorithms for distributional property testing, however, with many new technical ingredients. For Shannon entropy estimation, we improve the performance of the BHH framework, especially its error dependence, using Montanaro’s approach to estimating the expected output value of a quantum subroutine with bounded variance [25] and giving a fine-tuned error analysis. For general α-Rényi entropy estimation, we further develop a procedure that recursively approximates α-Rényi entropy for a sequence of αs, which is in spirit similar to the cooling schedules in simulated annealing. For special cases like integer α ≥ 2 and α = +∞ (i.e., the min-entropy case), we reduce the entropy estimation problem to the α-distinctness and the ⌈log n⌉-distinctness problems respectively. Finally, we exploit reductions as well as the polynomial method to obtain our lower bounds.

∗tongyang@cs.umd.edu
†xwu@cs.umd.edu
1 Introduction

Motivations. Property testing is a rapidly developing field in theoretical computer science (e.g., see the survey [35]). It aims to determine properties of an object with the least number of independent samples of the object. Property testing is a theoretically appealing topic with intimate connections to statistics, learning theory, and algorithm design. One important topic in property testing is to estimate statistical properties of unknown distributions (e.g., [40]), which are fundamental questions in statistics and information theory, given that much of science relies on samples furnished by nature. The Shannon [36] and Rényi [34] entropies are central measures of randomness compressibility. In this paper, we focus on estimating these entropies for an unknown distribution.

Specifically, given a distribution \( p \) over a set \( X \) of size \( n \) (w.l.o.g. let \( X = [n] \)) where \( p_x \) denotes the probability of \( x \in X \), the Shannon entropy \( H(p) \) of this distribution \( p \) is defined by

\[
H(p) := \sum_{x \in X : p_x > 0} p_x \log \left( \frac{1}{p_x} \right). \tag{1.1}
\]

A natural question is to determine the sample complexity (i.e., the necessary number of independent samples from \( p \)) to estimate \( H(p) \), with error \( \epsilon \) and high probability. This problem has been intensively studied in the classical literature. For multiplicative error \( \epsilon \), Batu et al. [6] provided the upper bound of \( O(n^{1+o(1)}/(1+\epsilon)^2) \), while an almost matching lower bound of \( \Omega(n^{1-o(1)}/(1+\epsilon)^2) \) was shown by Valiant [40]. For additive errors, Paninski gave a nonconstructive proof of the existence of sublinear estimators in [31, 32], while an explicit construction using \( \Theta(n/\log n) \) samples was shown by Valiant and Valiant in [39] when \( \epsilon > n^{-0.03} \). A sequence of works in information theory [20, 21, 42] studied the minimax mean-squared error, which becomes \( O(1) \) also using \( \Theta(n/\log n) \) samples.

One important generalization of Shannon entropy is the Rényi entropy of order \( \alpha > 0 \), denoted \( H_\alpha(p) \), which is defined by

\[
H_\alpha(p) := \begin{cases} 
\frac{1}{1-\alpha} \log \sum_{x \in X} p_x^\alpha, & \text{when } \alpha \neq 1, \\
\lim_{\alpha \to 1} H_\alpha(p), & \text{when } \alpha = 1.
\end{cases} \tag{1.2}
\]

The Rényi entropy of order 1 is simply the Shannon entropy, i.e., \( H_1(p) = H(p) \). General Rényi entropy can be used as a bound on Shannon entropy, making it useful in many applications (e.g., [5, 13]). Rényi entropy is also of interest on its own right. One prominent example is the Rényi entropy of order 2, \( H_2(p) \) (also known as the collision entropy), which measures the quality of random number generators (e.g., [41]) and key derivation in cryptographic applications (e.g., [9, 19]). Motivated by these and other applications, the estimation of Rényi entropy has also been actively studied [3, 20, 21]. In particular, Acharya et al. [3] have shown almost tight bounds on the classical query complexity of computing Rényi entropy. Specifically, for any non-integer \( \alpha > 1 \), the classical query complexity of \( \alpha \)-Rényi entropy is \( \Omega(n^{1-o(1)}) \) and \( O(n) \). Surprisingly, for any integer \( \alpha > 1 \), the classical query complexity becomes sublinear, which is \( \Theta(n^{1-1/\alpha}) \). When \( 0 \leq \alpha < 1 \), the classical query complexity is \( \Omega(n^{1/\alpha-o(1)}) \) and \( O(n^{1/\alpha}) \), which is always superlinear.

The extreme case (\( \alpha \to \infty \)) is known as the min-entropy, denoted \( H_\infty(p) \), which is defined by

\[
H_\infty(p) := \lim_{\alpha \to \infty} H_\alpha(p) = -\log \max_{i \in [n]} p_i. \tag{1.3}
\]

Min-entropy plays an important role in the randomness extraction (e.g., [38]) and characterizes the maximum number of uniform bits that can be extracted from a given distribution. Classically, the query complexity of min-entropy estimation is \( \Theta(n/\log n) \), which follows directly from [39].
Main question. In this paper, we study the impact of quantum computation on estimation of general Rényi entropies. Specifically, we ask\textsuperscript{1}

Is there any quantum speedup for estimating Shannon and Rényi entropies?

Our question aligns with the emerging topic called “quantum property testing” (see the survey [26]) and focuses on investigating the quantum advantage in testing classical statistical properties. To the best of our knowledge, the first research paper on distributional quantum property testing is by Bravyi, Harrow, and Hassidim (BHH) [11], where they discovered quantum speedups of testing uniformity, orthogonality, and statistical difference on unknown distributions. Some of these results were subsequently improved by Chakraborty et al [12]. There is also a related line of research on spectrum testing or tomography of quantum states [16, 28–30]. However, these works aim to test properties of general quantum states, while we focus on using quantum algorithms to test properties of classical distributions (i.e., diagonal quantum states)\textsuperscript{2}.

Distributions as oracles. The sampling model in the classical literature assumes that a tester is presented with independent samples from an unknown distribution. One of the contributions of BHH is an alternative model that allows coherent quantum access to unknown distributions. Specifically, BHH models a discrete distribution $p = (p_i)_{i=1}^n$ on $[n]$ by an oracle $O_p : [S] \rightarrow [n]$ for some $S \in \mathbb{N}$. The probability $p_i$ ($i \in [n]$) is proportional to the size of pre-image of $i$ under $O_p$. Namely, an oracle $O_p : [S] \rightarrow [n]$ generates $p$ if and only if for all $i \in [n],

$$p_i = |\{s \in [S] : O_p(s) = i\}|/S. \tag{1.4}$$

(note that we assume $p_i$s are rational numbers). If one samples $s$ uniformly from $[S]$, then the output $O_p(s)$ is from distribution $p$. Instead of considering sample complexity—that is, the number of used samples—we consider the query complexity in the oracle model that counts the number of oracle uses. Note that a tester interacting with an oracle can potentially be more powerful due to the possibility of learning the internal structure of the oracle as opposed to the sampling model. However, it is shown in [11] that the query complexity of the oracle model and the sample complexity of the sampling model are in fact the same classically.

The merit of the oracle model is that it naturally allows coherent access when extended to the quantum case, where we transform $O_p$ into a unitary operator $\hat{O}_p$ acting on $\mathbb{C}^S \otimes \mathbb{C}^{n+1}$ such that

$$\hat{O}_p|s\rangle|0\rangle = |s\rangle|O_p(s)\rangle \quad \forall s \in [S]. \tag{1.5}$$

Moreover, this oracle model can also be readily obtained in some algorithmic settings, e.g., when distributions are generated by some classical or quantum sampling procedure. Thus, statistical property testing results in this oracle model can be potentially leveraged in algorithm design.

Our Results. Our main contribution is a systematic study of both upper and lower bounds for the quantum query complexity of estimation of Rényi entropies (including Shannon entropy as a special case). Specifically, we obtain the following quantum speedups for different ranges of $\alpha$.

**Theorem 1.1.** There are quantum algorithms that approximate $H_\alpha(p)$ of distribution $p$ on $[n]$ within an additive error $0 < \epsilon \leq O(1)$ with success probability at least $2/3$ using\textsuperscript{3}

- $\tilde{O}(n^{1/\alpha-1/2}/\epsilon^2)$ quantum queries\textsuperscript{4} when $0 < \alpha < 1$. See Theorem 4.2.

\textsuperscript{1}The discussion in [11] hinted about extending their techniques to estimate Shannon entropy, however, without details. Indeed, our technique is inspired by [11], however, with significantly new ingredients.

\textsuperscript{2}Note that one can also leverage the results of [16, 28–30] to test properties of classical distributions. However, they are less efficient because they deal with a much harder problem involving general quantum states.

\textsuperscript{3}It should be understood that the success probability $2/3$ can be boosted to close to 1 without much overhead, e.g., see Lemma D.5 in Appendix D.

\textsuperscript{4}$O$ hides factors that are polynomial in $\log n$ and $\log 1/\epsilon$. 

2
Table 1: Summary of classical and quantum query complexity of $H_\alpha(p)$ for $\alpha > 0$, assuming $\epsilon = \Theta(1)$.

<table>
<thead>
<tr>
<th>$\alpha$</th>
<th>classical bounds</th>
<th>quantum bounds (this paper)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$0 &lt; \alpha &lt; 1$</td>
<td>$O\left(\frac{n^{1/2}}{\log n}\right), \Omega\left(n^{1/2-o(1)}\right)$ [3]</td>
<td>$\tilde{O}\left(n^{1/2-\frac{\epsilon}{3}}\right), \Omega\left(\max{\frac{1}{n^{2/3}}, n^{1/3}}\right)$</td>
</tr>
<tr>
<td>$\alpha = 1$</td>
<td>$\Theta\left(\frac{n}{\log n}\right)$ [21, 39]</td>
<td>$O(\sqrt{n}), \Omega\left(n^{\frac{1}{3}}\right)$</td>
</tr>
<tr>
<td>$\alpha &gt; 1, \alpha \notin \mathbb{N}$</td>
<td>$O\left(\frac{n^{1/2}}{\log n}\right), \Omega\left(n^{1-o(1)}\right)$ [3]</td>
<td>$\tilde{O}\left(n^{1-\frac{\epsilon}{3}}\right), \Omega\left(\max{\frac{1}{n^{2/3}}, \Omega\left(n^{\frac{1}{3}}\right)}\right)$</td>
</tr>
<tr>
<td>$\alpha = 2$</td>
<td>$\Theta\left(\frac{n}{\log n}\right)$ [3]</td>
<td>$\Theta\left(n^{\frac{1}{3}}\right)$</td>
</tr>
<tr>
<td>$\alpha &gt; 2, \alpha \in \mathbb{N}$</td>
<td>$\Theta\left(n^{1-1/\alpha}\right)$ [3]</td>
<td>$\tilde{O}\left(n^{\nu(1-1/\alpha)}, \Omega\left(n^{\frac{1}{3}-\frac{\epsilon}{3}}\right), \nu &lt; 3/4\right)$</td>
</tr>
<tr>
<td>$\alpha = \infty$</td>
<td>$\Theta\left(\frac{n}{\log n}\right)$ [39]</td>
<td>$O\left(Q\left(\left[\frac{16\log n}{\epsilon^2}\right]\right)\right)$-distinctness), $\Omega\left(\sqrt{n}\right)$</td>
</tr>
</tbody>
</table>

- $\tilde{O}\left(n^{\frac{1}{3}}\right)$ quantum queries when $\alpha = 1$, i.e., Shannon entropy. See Theorem 3.1.
- $\tilde{O}\left(n^{\nu(1-1/\alpha)}\right)$ quantum queries when $\alpha > 1, \alpha \in \mathbb{N}$ for some $\nu < \frac{3}{4}$. See Theorem 5.1.
- $\tilde{O}\left(n^{1-1/2\alpha}\right)$ quantum queries when $\alpha > 1, \alpha \notin \mathbb{N}$. See Theorem 4.1.
- $\tilde{O}\left(Q\left(\left[\frac{16\log n}{\epsilon^2}\right]\right)\right)$-distinctness) quantum queries when $\alpha = \infty$, where $Q\left(\left[\frac{16\log n}{\epsilon^2}\right]\right)$-distinctness is the quantum query complexity of the $\left[\frac{16\log n}{\epsilon^2}\right]$-distinctness problem. See Theorem 6.1.

Our quantum testers demonstrate advantages over classical ones for all $0 < \alpha < \infty$; in particular, our quantum tester has a quadratic speedup in the case of Shannon entropy. When $\alpha = \infty$, our quantum upper bound depends on the quantum query complexity of the $\left[\log n\right]$-distinctness problem, which is open to the best of our knowledge\(^5\) and might demonstrate a quantum advantage.

We have also obtained corresponding quantum lower bounds as follows. We summarize both bounds in Table 1 and visualize them in Figure 1.

**Theorem 1.2.** [See Theorem 7.1] Any quantum algorithm that approximates $H_\alpha(p)$ of distribution $p$ on $[n]$ within a constant additive error with success probability at least 2/3 must use

- $\Omega\left(n^{1/3-o(1)}\right)$ quantum queries when $0 < \alpha < \frac{3}{7}$.
- $\Omega\left(n^{\frac{1}{3}}\right)$ quantum queries when $\frac{3}{7} \leq \alpha \leq 3$.
- $\Omega\left(n^{\frac{1}{3}-\frac{1}{3\alpha}}\right)$ quantum queries when $3 \leq \alpha < \infty$.
- $\Omega\left(\sqrt{n}\right)$ quantum queries when $\alpha = \infty$.

**Techniques.** At a high level, our upper bound is inspired by BHH [11], where we formulate a framework (in Section 2) that generalizes the technique in BHH and makes it applicable in our case. Let $F(p) = \sum_x p_x f(p_x)$ for some function $f(\cdot)$ and distribution $p$. Similar to BHH, we design a master algorithm that samples $x$ from $p$ and then use the quantum counting primitive [10] to obtain an estimate $\tilde{p}_x$ of $p_x$ and outputs $f(\tilde{p}_x)$. It is easy to see that the expectation of the output of the master algorithm is roughly\(^6\) $F(p)$. By choosing appropriate $f(\cdot)$s, one can recover $H(p)$ or $H_\alpha(p)$ as well as the ones used in BHH. It suffices then to obtain a good estimate of the output expectation of the master algorithm, which was achieved by multiple independent runs of the master algorithm in BHH.

\(^5\) Existing quantum algorithms for the $k$-distinctness problem (e.g. [4, 8]) do not behave well for super-constant $ks$.

\(^6\) The accurate expectation is $\sum_x p_x E[f(\tilde{p}_x)]$. Intuitively, we expect $\tilde{p}_x$ to be a good estimate of $p_x$.  

3
The performance of the above framework (and its analysis) critically depends on how close the expectation of the algorithm is to $F(p)$ and how concentrated the output distribution is around its expectation, which in turn heavily depends on the specific $f(\cdot)$ in use. Our first contribution is a fine-tuned error analysis for specific $f(\cdot)$s, such as in the case of Shannon entropy (i.e., $f(p_x) = -\log(p_x)$) whose values could be significant for boundary cases of $p_x$. Instead of only considering the case when $\tilde{p}_x$ is a good estimate of $p_x$ as in BHH, we need to analyze the entire distribution of $\tilde{p}_x$ using quantum counting. We also leverage a generic quantum speedup for estimating the expectation of the output of any quantum procedure with additive errors [25], which significantly improves our error dependence as compared to BHH. This improvement (elaborated in Section 3) already gives a quadratic quantum speedup for Shannon entropy estimation.

For general $\alpha$-Rényi entropy $H_\alpha(p)$, we choose $f(p_x) = p_x^{\alpha-1}$ and let $P_\alpha(p) = F(p)$ so that $H_\alpha(p) \leq \log P_\alpha(p)$. Instead of estimating $F(p)$ with additive errors in the case of Shannon entropy, we switch to work with multiplicative errors which is a harder case since the aforementioned quantum algorithm [25] is much weaker in this setting. Indeed, by following the same technique, we can only obtain quantum speedups for $\alpha$-Rényi entropy when $1/2 < \alpha < 2$. Therefore, we have developed several new techniques to obtain quantum speedups for general $\alpha > 0$.

For general $\alpha > 0$, our first observation is that if one knew the output expectation $E[X]$ is within $[a, b]$ such that $b/a = \Theta(1)$, then one can slightly modify the technique in [25] (as shown in Theorem 2.2) and obtain a quadratic quantum speedup similar to the additive error setting. This approach, however, seems circular since it is unclear how to obtain such $a, b$ in advance. Our second observation is that for any close enough $\alpha_1, \alpha_2$, $P_{\alpha_1}(p)$ can be used to bound $P_{\alpha_2}(p)$. Precisely, when $\alpha_1/\alpha_2 = 1 \pm 1/\log(n)$, we have $P_{\alpha_1}(p) = \Theta(P_{\alpha_2}(p)^{\alpha_1/\alpha_2})$ (see Lemma D.3). As a result, when estimating $P_\alpha(p)$, we can first estimate $P_{\alpha'}$ to provide a bound on $P_\alpha$, where $\alpha', \alpha$ differ by a $1 \pm 1/\log(n)$ factor and $\alpha'$ moves toward 1. We apply this strategy recursively on estimating $P_{\alpha'}$ until $\alpha'$ is very close to 1 from above when initial $\alpha > 1$ or from below when initial $\alpha < 1$, where a quantum speedup is already known. At a high level, we will recursively estimate a sequence (of size $O(\log(n))$) of such that eventually converges to 1, where in each iteration we establish some quantum speedup which leads to a overall quantum speedup. We remark that our approach is in spirit similar to the cooling schedules in simulated annealing (e.g. [37]). (See Section 4.)

For integer $\alpha \geq 2$, we observe a connection between $P_\alpha(p)$ and the $\alpha$-distinctness problem which leads to a more significant quantum speedup. Precisely, let $O_p : [S] \to [n]$ be the oracle in (1.5), we observe that $P_\alpha(p)$ is proportional to the $\alpha$-frequency moment of $O_p(1), \ldots, O_p(S)$ which can be solved quantumly [24] based on any quantum algorithm for the $\alpha$-distinctness problem (e.g., [8]).
However, there is a catch that a direct application of [24] will lead to a dependence on $S$ rather than $n$. We remedy this situation by tweaking the algorithm and its analysis in [24] to remove the dependence on $S$ for our specific setting. (see Section 5.)

The integer $\alpha$ algorithm fails to extend to the min-entropy case (i.e., $\alpha = +\infty$) because the hidden constant in $O(\cdot)$ has a poor dependence on $\alpha$ (see Remark C.1). Instead, we develop another reduction to the $\lceil \log n \rceil$-distinctness problem by exploiting the so-called “Poissonized sampling” technique. At a high level, we construct Poisson distributions that are parameterized by $p_i$s and leverage the "threshold" behavior of Poisson distributions (see Lemma F.1). Roughly, if $\max_i p_i$ passes some threshold, with high probability, these parameterized Poisson distributions will lead to a collision of size $\lceil \log n \rceil$ that will be caught by the $\lceil \log n \rceil$-distinctness algorithm. Otherwise, we run again with a lower threshold until the threshold becomes trivial. (see Section 6.)

Some of our lower bounds come from reductions to existing ones in quantum query complexity, such as the quantum-classical separation of symmetric boolean functions [1], the collision problem [2, 22], and the Hamming weight problem [27], for different ranges of $\alpha$. We also obtain lower bounds with a better error dependence by the polynomial method, which is inspired by the celebrated quantum lower bound for the collision problem [2, 22]. (See Section 7.)

Open questions. Our paper raises a few open questions. A natural question is to close the gaps between our quantum upper and lower bounds. Our quantum techniques on both ends are actually quite different from the state-of-the-art classical ones (e.g., [39]). It is interesting to see whether one can incorporate classical ideas to improve our quantum results. It is also possible to achieve better lower bounds by improving our polynomial method or exploiting the quantum adversary method. Finally, our result motivates the study of the quantum algorithm for the $k$-collision problem with super-constant $k$, which could also be interesting by itself.

2 Master algorithm

Let $p = (p_i)_{i=1}^n$ be a discrete distribution on $[n]$ encoded by the quantum oracle $\hat{O}_p$ defined in (1.5). Inspired by BHH, we develop the following master algorithm to estimate $F(p) := \sum_{i \in [n]} p_i f(p_i)$ for a function $f: (0, 1] \to \mathbb{R}$.

<table>
<thead>
<tr>
<th>Algorithm 1: Estimate $F(p) = \sum_i p_i f(p_i)$ of a discrete distribution $p = (p_i)_{i=1}^n$ on $[n]$.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. <strong>Set</strong> $l, M \in \mathbb{N}$;</td>
</tr>
<tr>
<td>2. <strong>Regard the following subroutine as $A$:</strong></td>
</tr>
<tr>
<td>3. Draw a sample $i \in [n]$ according to $p$;</td>
</tr>
<tr>
<td>4. Use $\text{EstAmp}$ or $\text{EstAmp}'$ with $M$ queries to obtain an estimation $\tilde{p}_i$ of $p_i$;</td>
</tr>
<tr>
<td>5. Output $X = f(\tilde{p}_i)$;</td>
</tr>
<tr>
<td>6. <strong>Use</strong> $A$ for $l$ executions in Theorem 2.1 or Theorem 2.2 and output $\tilde{F}(p)$ to estimate $F(p)$;</td>
</tr>
</tbody>
</table>

Comparing to BHH, we introduce a few new technical ingredients in the design of Algorithm 1 and its analysis, which significantly improves the performance of Algorithm 1 especially for specific $f(\cdot)$s in our case, e.g., $f(p_x) = -\log(p_x)$ (Shannon entropy) and $f(p_x) = p_x^{\alpha-1}$ (Rényi entropy).

The **first** one is a generic quantum speedup of Monte Carlo methods [25], in particular, a quantum algorithm that approximates the output expectation of a subroutine with additive errors that has a quadratic better sample complexity than the one implied by Chebyshev’s inequality.
Theorem 2.1 (Additive error; Theorem 5 of [25]). Let \( A \) be a quantum algorithm with output \( X \) such that \( \text{Var}[X] \leq \sigma^2 \). Then for \( \epsilon \) where \( 0 < \epsilon < 4\sigma \), by using \( O\left((\sigma/\epsilon)^{5/2}(\sigma/\epsilon)\log(\sigma/\epsilon)\right) \) executions of \( A \) and \( A^{-1} \), Algorithm 3 in [25] outputs an estimate \( \tilde{E}[X] \) of \( E[X] \) such that

\[
\Pr \left[ \left| \tilde{E}[X] - E[X] \right| \geq \epsilon \right] \leq 1/5. \tag{2.1}
\]

It is worthwhile mentioning that classically one needs to use \( \Omega(\sigma^2/\epsilon^2) \) executions of \( A \) [14] to estimate \( E(x) \). Theorem 2.1 demonstrates a quadratic improvement on the error dependence. In the case of approximating \( H_p(\epsilon) \), we need to work with multiplicative errors while existing results (e.g. [25]) have a worse error dependence which is insufficient for our purposes. Instead, inspired by [25], we prove the following theorem (our second ingredient) that takes auxiliary information about the range of \( E(X) \) into consideration, which might be of independent interest.

Theorem 2.2 (Multiplicative error; Appendix A). Let \( A \) be a quantum algorithm with output \( X \) such that \( \text{Var}[X] \leq \sigma^2 E[X]^2 \) for a known \( \sigma \). Assume that \( E[X] \in [a,b] \). Then for \( \epsilon \) where \( 0 < \epsilon < 2\sigma \), by using \( A \) and \( A^{-1} \) for \( O((\sigma/\epsilon)\log^{3/2}(\sigma/\epsilon)\log(\sigma/\epsilon)) \) executions, Algorithm 6 (given in Appendix A) outputs an estimate \( \tilde{E}[X] \) of \( E[X] \) such that

\[
\Pr \left[ \left| \tilde{E}[X] - E[X] \right| \geq \epsilon E[X] \right] \leq 1/5. \tag{2.2}
\]

The third ingredient is a fine-tuned error analysis due to the specific \( f(\cdot) \)-s. Similar to BHH, we rely on quantum counting (named EstAmp) [10] to estimate the pre-image size of a Boolean function, which provides another source of quantum speedup. In particular, we approximate any probability \( p_x \) in the query model ((1.5)) by \( \tilde{p}_x \) by estimating the size of the pre-image of a Boolean function \( \chi: [S] \rightarrow \{0,1\} \) with \( \chi(s) = 1 \) if \( O(s) = i \) and \( \chi(s) = 0 \) otherwise. However, for cases in BHH, it suffices to only consider the probability when \( p_x \) and \( \tilde{p}_x \) are close, while in our case, we need to analyze the whole output distribution of quantum counting. Specifically, letting \( t = |\chi^{-1}(1)| \) and \( a = t/S = \sin^2(\omega \pi) \) for some \( \omega \), we have

Theorem 2.3 ([10]). For any \( k, M \in N \), there is a quantum algorithm (named EstAmp) with \( M \) quantum queries to \( \chi \) that outputs \( \tilde{a} = \sin^2 \left( \frac{\pi a}{M} \right) \) for some \( l \in \{0, \ldots, M-1\} \) such that

\[
\Pr \left[ \tilde{a} = \sin^2 \left( \frac{l \pi}{M} \right) \right] = \frac{\sin^2(M \Delta \pi)}{M^2 \sin^2(\Delta \pi)} \leq \frac{1}{(2M \Delta)^2}, \tag{2.3}
\]

where \( \Delta = |\omega - \frac{l}{M}| \). This promises \( |\tilde{a} - a| \leq 2\pi k \sqrt{\frac{a(1-a)}{M}} + k^2 \frac{\pi^2}{M^2} \) with probability at least \( \frac{8}{M^4} \) for \( k = 1 \) and with probability greater than \( 1 - \frac{1}{2(k-1)} \) for \( k \geq 2 \). If \( a = 0 \) then \( \tilde{a} = 0 \) with certainty.

Moreover, we also need to slightly modify EstAmp to avoid outputting \( \tilde{p}_x = 0 \) in estimating Shannon entropy. This is because \( f(\tilde{p}_x) = \log(\tilde{p}_x) \) is not well-defined at \( \tilde{p}_x = 0 \). Let EstAmp' be the modified algorithm. It is required that EstAmp' outputs \( \sin^2 \left( \frac{\pi a}{2M} \right) \) when EstAmp outputs 0 and outputs EstAmp's output otherwise.

By leveraging Theorem 2.1, Theorem 2.2, Theorem 2.3, and carefully setting parameters in Algorithm 1, we have the following corollaries that describe the complexity of estimating any \( F(p) \).

Corollary 2.1 (additive error). Given \( \epsilon > 0 \). If \( l = \Theta \left( \frac{(\sigma^2)}{\epsilon} \log^{3/2} \left( \frac{\pi \sigma^2}{\epsilon} \right) \log \left( \frac{\pi \sigma^2}{\epsilon} \right) \right) \) where \( \text{Var}[X] \leq \sigma^2 \) and \( M \) is large enough such that \( |E[X] - F(p)| \leq \epsilon \), then Algorithm 1 approximates \( F(p) \) with an additive error \( \epsilon \) and success probability \( 2/3 \) using \( O(M \cdot l) \) quantum queries to \( p \).

Corollary 2.2 (multiplicative error). Assume a procedure using \( C_{a,b} \) queries that returns an estimated range \([a, b]\) of \( E[X] \) with probability 0.9. Let \( l = \Theta \left( \frac{(\sigma^2)}{\epsilon} \log^{3/2} \left( \frac{\pi \sigma^2}{\epsilon} \right) \log \left( \frac{\pi \sigma^2}{\epsilon} \right) \right) \) where \( \text{Var}[X]/(E[X])^2 \leq \sigma^2 \) and \( \epsilon > 0 \). For large enough \( M \) such that \( |E[X] - F(p)| \leq \epsilon \), Algorithm 1 estimates \( F(p) \) with a multiplicative error \( \epsilon \) and success probability \( 2/3 \) with \( O(M\cdot l + C_{a,b}) \) queries.
3 Shannon entropy estimation

We develop Algorithm 2 for Shannon entropy estimation with EstAmp' in Line 4, which provides quadratic quantum speedup in $n$.

**Algorithm 2**: Estimate the Shannon entropy of $p = (p_i)_{i=1}^n$ on $[n]

1. Set $l = \Theta\left(\frac{\log n}{\epsilon} \log^{3/2} \left( \frac{\log n}{\epsilon} \right) \log \log \left( \frac{\log n}{\epsilon} \right) \right)$;
2. Regard the following subroutine as $A$
3. Draw a sample $i \in [n]$ according to $p$;
4. Use EstAmp' with $M = 2^\left[\log_2(\sqrt{n}/\epsilon)\right]$ queries to obtain an estimation $\hat{p}_i$ of $p_i$;
5. Output $\hat{x}_i = \log(1/\hat{p}_i)$;
6. Use $A$ for $l$ executions in Theorem 2.1 and output an estimation $\hat{H}(p)$ of $H(p)$;

**Theorem 3.1.** Algorithm 2 approximates $H(p)$ within an additive error $0 < \epsilon \leq O(1)$ with success probability at least $2/3$ using $O\left(\frac{2^m}{\epsilon^2}\right)$ quantum queries to $p$.

*Proof.* We prove this theorem in two steps. The first step is to show that the expectation of the subroutine $A$’s output (denoted $\hat{E} := \sum_{i \in [n]} p_i \cdot \log(1/\hat{p}_i)$) is close to $E := \sum_{i \in [n]} p_i \cdot \log(1/p_i) = H(p)$. To that end, we divide $[n]$ into partitions based on the corresponding probabilities. Let $m = \left[\log_2(\sqrt{n}/\epsilon)\right]$ and $S_0 = \{i : p_i \leq \sin^2(\pi/2^{m+1})\}$, $S_1 = \{i : \sin^2(\pi/2^{m+1}) < p_i \leq \sin^2(\pi/2^m)\}$, $S_2 = \{i : \sin^2(\pi/2^m) < p_i \leq \sin^2(\pi/2^{m-1})\}, \ldots, S_m = \{i : \sin^2(\pi/4) < p_i \leq \sin^2(\pi/2)\}$. For convenience, $s_0 = |S_0|, s_1 = |S_1|, \ldots, s_m = |S_m|$. Then

$$\sum_{j=0}^m s_j = n, \quad \sum_{j=0}^m 2^{2j/s_j} = \Theta(1), \quad (3.1)$$

Our main technical contribution (full proof in Appendix B) is the following upper bound on the expected difference between $\log \hat{p}_i$ and $\log p_i$ in terms of the partition $S_i$, $i = 1, \ldots, n$.

$$\sum_{i \in S_j} p_i \mathbb{E}[|\log \hat{p}_i - \log p_i|] = O\left(\frac{2^j s_j}{2^m}\right) \quad \forall \, j \in \{1, \ldots, m\}, \quad (3.2)$$

By linearity of expectation, we have

$$|\hat{E} - E| \leq \sum_{i \in [n]} p_i \mathbb{E}[|\log \hat{p}_i - \log p_i|] = \sum_{j=0}^m \sum_{i \in S_j} p_i \mathbb{E}[|\log \hat{p}_i - \log p_i|]. \quad (3.3)$$

As a result, by applying (3.1) and Cauchy-Schwartz inequality to (3.3), we have

$$|\hat{E} - E| = \sum_{j=0}^m O\left(\frac{2^j s_j}{2^m}\right) \leq O\left(\sqrt{\sum_{j=0}^m \frac{1}{2^{2m/s_j}} \left( \sum_{j=0}^m \frac{2^j}{2^{2m/s_j}} \right)}\right) = O(\epsilon). \quad (3.4)$$

Because a constant overhead does not influence the query complexity, we may rescale Algorithm 2 by a large enough constant so that $|\hat{E} - E| \leq \epsilon/2$.

The second step is to bound the variance of the random variable, which is

$$\sum_{i \in [n]} p_i (\log \hat{p}_i)^2 - \left(\sum_{i \in [n]} p_i \log \hat{p}_i\right)^2 \leq \sum_{i \in [n]} p_i (\log \hat{p}_i)^2. \quad (3.5)$$
Since for any $i$, $\text{EstAmp}'$ outputs $\tilde{p}_i$ such that $p_i \geq \sin^2(\frac{\pi}{2M}) \geq \frac{1}{M} \geq \frac{2}{4n}$, we have $\sum_{i \in [n]} p_i (\log \tilde{p}_i)^2 \leq \sum_{i \in [n]} p_i \left( \log \frac{4n}{\epsilon} \right)^2 = \left( \log \frac{4n}{\epsilon} \right)^2$. As a result, by Corollary 2.1 we can approximate $\tilde{E}$ up to additive error $\epsilon/2$ with failure probability at most $1/3$ using

$$O \left( \frac{(n/\epsilon)^2 \log 3/2 \left( \frac{\log (n/\epsilon)}{\epsilon} \right) \log \log \left( \frac{(n/\epsilon)^2}{\epsilon} \right)}{\epsilon} \right) \cdot 2^{[\log_2(\sqrt{n}/\epsilon)]} = \tilde{O} \left( \frac{\sqrt{n}}{\epsilon^2} \right)$$

(3.6) quantum queries. Together with $|\tilde{E} - E| \leq \epsilon/2$, Algorithm 2 approximates $E = H(p)$ up to additive error $\epsilon$ with failure probability at most $1/3$.

\section{Non-integer Rényi entropy estimation}

Recall the classical query complexity of non-integer and integer Rényi entropy estimations are different [3]. Quantumly, we also consider them separately; in this section, we consider $\alpha$-Rényi entropy estimation for general non-integer $\alpha > 0$.

Let $P_\alpha(p) := \sum_{i=1}^{n} p_i^\alpha$. Since $H_\alpha(p) = \frac{1}{1-\alpha} \log P_\alpha(p)$, to approximate $H_\alpha(p)$ within an additive error $\epsilon > 0$ it suffices to approximate $P_\alpha(p)$ within a multiplicative error $e^{(\alpha-1)\epsilon} = \Theta(\epsilon)$.

\textbf{Case 1:} $\alpha > 1, \alpha \notin \mathbb{N}$

We develop Algorithm 3 to approximate $P_\alpha(p)$ with a multiplicative error $\epsilon$.

\begin{algorithm}
\textbf{Algorithm 3:} Estimate the $\alpha$-power sum $P_\alpha(p)$ of $p = (p_i)_{i=1}^n$ on $[n], \alpha > 1, \alpha \notin \mathbb{N}$

\begin{enumerate}
\item Draw a sample $i \in [n]$ according to $p$;
\item Use the amplitude estimation procedure $\text{EstAmp}$ with $M = 2^{[\log_2(\sqrt{\epsilon} \log(\frac{2n}{\epsilon}))]+1}$ queries to obtain an estimate $\tilde{p}_i$ of $p_i$;
\item Output $\tilde{x}_i = \tilde{p}_i^{\alpha-1}$;
\end{enumerate}

1 Input parameters $(\alpha, \epsilon, \delta)$, where $\epsilon$ is the multiplicative error and $\delta$ is the failure probability;
2 if $\alpha < 1 + \frac{1}{\log n}$ then
3 \hspace{1em} Take $a = \frac{1}{\epsilon}$ and $b = 1$ as lower and upper bounds on $P_\alpha(p)$, respectively;
4 else
5 \hspace{1em} Recursively call Algorithm 3 with $\alpha' = \alpha(1 + \frac{1}{\log n})^{-1}$, $\epsilon = 1/4$, and $\delta = \frac{1}{12 \log n \log \alpha}$ therein to give an estimate $\tilde{P}_{\alpha'}(p)$ of $P_{\alpha'}(p)$. For simplicity, denote $P := \tilde{P}_{\alpha'}(p)$. Take $a = \frac{(3P/4)^{1+\frac{1}{\log \pi}}}{\epsilon}$ and $b = (\frac{4P}{\pi})^{1+\frac{1}{\log \pi}}$ as lower and upper bounds on $P_{\alpha}(p)$, respectively;
6 Set $l = \Theta \left( \frac{n^{1+\frac{1}{\log \pi}}}{\epsilon} \log^{3/2} \left( \frac{n^{1+\frac{1}{\log \pi}}}{\epsilon} \log \left( \frac{n^{1+\frac{1}{\log \pi}}}{\epsilon} \right) \right) \right)$;
7 Use $A$ for $l$ executions in Theorem 2.2 using $a$ and $b$ as auxiliary information and output an estimation of $P_{\alpha}(p)$;
\end{algorithm}

First, we design a subroutine $\mathcal{A}$ in Algorithm 3 to approximate $P_{\alpha}(p)$ following the same principle as in Algorithm 2. The analysis of the subroutine $\mathcal{A}$ is also similar except that we want a multiplicative error for $P_{\alpha}(p)$, where we need to use Theorem 2.2 that takes a lower and upper bound, $a$ and $b$ respectively, on $\tilde{E}$ as auxiliary information. The extra overhead of Theorem 2.2
comparing to Theorem 2.1 is $b/a$ that could be significant when $a, b$ are loose. Our key observation is to use $P_{\alpha_1}$ to bound $P_{\alpha_2}$ for carefully chosen $0 < \alpha_1 < \alpha_2$. Specifically, by Lemma D.3, we have

$$P_{\alpha_2}(p)^{\alpha_1/\alpha_2} \leq P_{\alpha_1}(p) \leq n^{-\alpha_1/\alpha_2} P_{\alpha_2}(p)^{\alpha_1/\alpha_2}. \quad (4.1)$$

Let $\frac{\alpha_1}{\alpha_2} = 1 + O(\frac{1}{\log n}).$ Because $n^{1/\log n} = e$, we have $P_{\alpha_1}(p) = \Theta(P_{\alpha_2}(p)^{\alpha_1/\alpha_2})$. Therefore, if $\alpha < 1 + \frac{1}{\log n}$ (judged by Line 2), $a = \frac{1}{\epsilon}$ and $b = 1$ in Line 3 are simply lower and upper bounds on $P_{\alpha}(p)$; otherwise, we compute $a$ and $b$ by recursively calling Algorithm 3 to estimate $P_{\alpha'}(p)$ for $\alpha' = \alpha/(1 + 1/\log n)$, which is then used to obtain $a$ and $b$ in Line 5 by (4.1). Because each time $\alpha$ becomes smaller by a multiple of $(1 + 1/\log n)^{-1}$, Algorithm 3 is recursively called by at most $\log n \log \alpha$ times. Furthermore, $b/a < 4e = O(1)$ in both Line 3 and Line 5.

On the other hand, we can show that with high probability the variance of $A_{\alpha}$ becomes smaller by a multiple of $(1 + 1/\log n)$ each time; nevertheless, both recursions end when $\alpha'$ is close enough to 1. On the more technical level, they have different $M$ in $A_{\alpha}$, different upper bounds on the variance of $A_{\alpha}$, and different expressions for $a$ and $b$ in Line 5.

**Theorem 4.1** (Full proof in Appendix D). Algorithm 3 approximates $P_{\alpha}(p)$ within a multiplicative error $0 < \epsilon \leq 1/4$ with success probability $2/3$ using $\tilde{O}(n^{1-1/\alpha})$ quantum queries to $p$.

**Case 2:** $0 < \alpha < 1$

When $0 < \alpha < 1$, our quantum algorithm (Algorithm 8 in Appendix E) follows the same structure as Algorithm 3. The main difference is that, in the case $\alpha > 1$, Algorithm 3 makes $\alpha'$ smaller and smaller by multiplying $(1 + 1/\log n)^{-1}$ each time, whereas in the case $0 < \alpha < 1$, Algorithm 8 makes $\alpha'$ larger and larger by multiplying $(1 - 1/\log n)^{-1}$ each time; nevertheless, both recursions end when $\alpha'$ is close enough to 1. On the more technical level, they have different $M$ in $A_{\alpha}$, different upper bounds on the variance of $A_{\alpha}$, and different expressions for $a$ and $b$ in Line 3 and Line 5.

**Theorem 4.2** (Full proof in Appendix E). Algorithm 8 approximates $P_{\alpha}(p)$ within a multiplicative error $0 < \epsilon \leq O(1)$ with success probability $2/3$ using $\tilde{O}(n^{1-1/\alpha})$ quantum queries to $p$.

### 5 Integer Rényi entropy estimation

Recall the classical query complexity of $\alpha$-Rényi entropy estimation for $\alpha \in \mathbb{N}, \alpha \geq 2$ is $\Theta(n^{1-1/\alpha})$ [3], which is smaller than non-integer cases. Quantumly, we also provide a more significant speedup.

Given the oracle $O_p : [S] \to [n]$ in (1.5), we denote the occurrences of $1, 2, \ldots, n$ among $O_p(1), \ldots, O_p(S)$ as $m_1, \ldots, m_n$, respectively. A key observation is that by (1.4), we have

$$P_{\alpha}(p) = \sum_{i=1}^{n} (m_i/S)^\alpha = S^{-\alpha} \sum_{i=1}^{n} m_i^\alpha. \quad (5.1)$$

Therefore, it suffices to approximate $\sum_{i \in [n]} m_i^\alpha$, which is known as the $\alpha$-frequency moment of $O_p(1), \ldots, O_p(S)$. Based on the quantum algorithm for $\alpha$-distinctness [8], Montanaro [24] proved:
Fact 5.1 ([24], Step 3b-step 3e in Algorithm 2; Lemma 4). Fix \( l \) such that \( 1 \leq l \leq n \). Let \( s_1, \ldots, s_l \in [S] \) be picked uniformly at random, and denote the number of \( \alpha \)-wise collisions in \( \{O_p(s_1), \ldots, O_p(s_l)\} \) as \( C(s_1, \ldots, s_l) \). Then:

- \( C(s_1, \ldots, s_l) \) can be computed using \( O(l^\nu \log(l/\epsilon^2)) \) queries to \( \hat{O}_p \) with failure probability at most \( O(\epsilon^2/l) \), where \( \nu := 1 - 2^{\alpha-2}/(2^\alpha - 1) < \frac{3}{4} \);
- \( \mathbb{E}[C(s_1, \ldots, s_l)] = \binom{l}{\alpha} P_\alpha(p) \) and \( \text{Var}[C(s_1, \ldots, s_l)] = O(1) \).

However, a direct application of [24] will lead to a complexity depending on \( S \) (in particular, \( l \) in Fact 5.1 can be as large as \( S \)) rather than \( n \). Our solution is Algorithm 4 that is almost the same as Algorithm 2 in [24] except Line 1 and Line 2, where we set \( 2^{[\log_2 \alpha n]} \) as an upper bound of \( l \). We claim such choice of \( l \) is valid in case because by the pigeonhole principle, \( \alpha \) elements \( O_p(s_1), \ldots, O_p(s_2) \) in \( [n] \) must have an \( \alpha \)-collision, so the first for-loop must terminate at some \( i \leq [\log_2 \alpha n] \). With this modification, we have Theorem 5.1 for integer R\'enyi entropy estimation.

Algorithm 4: Estimate the \( \alpha \)-power sum \( P_\alpha(p) \) of \( p = (p_i)_{i=1}^n \) on \( [n] \), \( \alpha > 1, \alpha \in \mathbb{N} \).

1. Set \( l = 2^{[\log_2 \alpha n]} \);
2. for \( i = 0, \ldots, [\log_2 \alpha n] \) do
3.     Pick \( s_1, \ldots, s_l \in [S] \) uniformly at random and let \( S \) be the sequence \( O_p(s_1), \ldots, O_p(s_l) \);
4.     Apply the \( \alpha \)-distinctness algorithm in [8] to \( S \) with failure probability \( \frac{1}{100[\log_2 \alpha n]} \);
5.     If it returns a set of \( \alpha \) equal elements, set \( l = 2^i \) and terminate the loop;
6. Set \( M = [K/e^2] \) for some \( K = \Theta(1) \);
7. for \( r = 1, \ldots, M \) do
8.     Pick \( s_1, \ldots, s_l \in [S] \) uniformly at random;
9.     Apply the first bullet in Fact 5.1 to give an estimate \( C^{(r)} \) of the number of \( \alpha \)-wise collisions in \( \{O_p(s_1), \ldots, O_p(s_l)\} \);
10. Output \( \tilde{P}_\alpha(p) = \frac{1}{M^{(l)}} \sum_{r=1}^{M^{(l)}} C^{(r)} \).

Theorem 5.1 (Full proof in Appendix C). Assume \( \alpha > 1, \alpha \in \mathbb{N} \). Algorithm 4 approximates \( P_\alpha(p) \) within a multiplicative error \( 0 < \epsilon < O(1) \) with success probability at least \( \frac{2}{3} \) using \( \tilde{O}\left(\frac{n^{\nu(1-1/\alpha)}}{\epsilon^2}\right) = O\left(\frac{n^{\frac{\alpha}{2}(1-1/\alpha)}}{\epsilon^2}\right) \) quantum queries to \( p \), where \( \nu := 1 - 2^{\alpha-2}/(2^\alpha - 1) < \frac{3}{4} \).

6 Min-entropy estimation

Since the min-entropy of \( p \) is \( H_\infty(p) = -\log \max_{i \in [n]} p_i \) by (1.3), it is equivalent to approximate \( \max_{i \in [n]} p_i \) within multiplicative error \( \epsilon \). We propose Algorithm 5 below to achieve this task.

Theorem 6.1. Algorithm 5 approximates \( \max_{i \in [n]} p_i \) within a multiplicative error \( 0 < \epsilon \leq 1 \) with success probability at least \( \Omega(1) \) using \( \tilde{O}(Q(\lceil \frac{16 \log n}{\epsilon^2} \rceil)-\text{distinctness}) \) quantum queries to \( p \), where \( Q(\lceil \frac{16 \log n}{\epsilon^2} \rceil)-\text{distinctness} \) is the quantum query complexity of \( \lceil \frac{16 \log n}{\epsilon^2} \rceil \)-distinctness problem.

Proof sketch. A key property of the Poisson distribution is that if we take \( M \sim \text{Poi}(\nu) \) samples from \( p \) (as in Line 3), then for each \( j \in [n] \), the number of occurrences of \( j \) in \( \{O_p(s_1), \ldots, O_p(s_M)\} \) follows the Poisson distribution \( M_j \sim \text{Poi}(\nu p_j) \), and \( M_j, M_{j'} \) are independent for all \( j \neq j' \). Furthermore, if a random variable \( X \) follows \( \text{Poi}(\mu) \), then (see Lemma F.1 in Appendix F):
Algorithm 5: Estimate $\max_{i \in \{n\}} p_i$ of a discrete distribution $p = (p_i)_{i=1}^n$ on $[n]$.

1. Set $\lambda = 1$;
2. while $\lambda \leq n$ do
   3. Take $M \sim \text{Poi}(\frac{16\lambda \log n}{\epsilon^2})$. Pick $s_1, \ldots, s_M \in [S]$ uniformly at random and let $\mathcal{S}$ be the sequence $O_p(s_1), \ldots, O_p(s_M)$;
   4. Apply an $\left[\frac{16\log n}{\epsilon^2}\right]$-distinctness quantum algorithm to $\mathcal{S}$ with failure probability at most $\frac{1}{2\lambda \log n}$;
   5. If Line 4 gives a set of $\left[\frac{16\log n}{\epsilon^2}\right]$ equal elements $l \in [n]$, apply quantum counting [10] to $l$ with multiplicative error $\epsilon$ and output its result; if not, set $\lambda \leftarrow \lambda \cdot \sqrt{1+\epsilon}$ and jump to the start of the loop;
6. If $\lambda > n$ and no output has been given, output $1/n$;

- If $\mu < \frac{1}{\sqrt{1+\epsilon}} \cdot \frac{16\log n}{\epsilon^2}$, then $\Pr\left[X \geq \frac{16\log n}{\epsilon^2}\right] \leq \frac{1}{n^2}$;
- If $\mu \geq \frac{16\log n}{\epsilon^2}$, then $\Pr\left[X \geq \frac{16\log n}{\epsilon^2}\right] \geq \frac{1}{2e}$.

Based on these properties, our main strategy is to set $\frac{16\log n}{\epsilon^2}$ as a threshold, take $\nu = \frac{16\lambda \log n}{\epsilon^2}$ as in Line 3, and gradually increase the parameter $\lambda$. For convenience, denote $p_{i^*} = \max_i p_i$. As long as $\nu \cdot p_{i^*} < \frac{16\log n}{\epsilon^2}$, with high probability there is no $\left[\frac{16\log n}{\epsilon^2}\right]$-collision in $\mathcal{S}$, the distinctness quantum algorithm in Line 4 rejects, and $\lambda$ increases by multiplying $\sqrt{1+\epsilon}$ in Line 5; right after the first time when $\nu \cdot p_{i^*} \geq \frac{16\log n}{\epsilon^2}$, with probability at least $1/2e$, $i^*$ has an $\left[\frac{16\log n}{\epsilon^2}\right]$-collision in $\mathcal{S}$, whereas all other entries in $[n]$ do not (with failure probability at most $1/n^2$). In this case, with probability at least $\Omega(1)$, the distinctness quantum algorithm in Line 4 captures $i^*$, and the quantum counting in Line 5 computes $p_{i^*}$ within multiplicative error $\epsilon$.

7 Quantum lower bounds

In this section, we prove Theorem 1.2, which is rewritten below:

**Theorem 7.1.** Assuming $\epsilon = \Theta(1)$, the quantum query complexity of $\alpha$-Rényi entropy estimation is $\Omega(n^{\frac{1}{\alpha} - o(1)})$ if $0 < \alpha < \frac{3}{7}$, $\Omega(n^{\frac{1}{2}})$ if $\frac{3}{7} \leq \alpha \leq 3$, $\Omega(n^{\frac{1}{\alpha} - \frac{1}{\alpha^2}})$ if $3 \leq \alpha < \infty$, and $\Omega(\sqrt{n})$ if $\alpha = \infty$.

We first give a proof using reductions from existing quantum lower bounds.

**Proof.** First, by [3], we know that $\Omega(n^{\frac{1}{\alpha} - o(1)})$ is a lower bound of the classical query complexity of $\alpha$-Rényi entropy. On the other hand, reference [1] shows that for any problem that is invariant under permuting inputs and outputs and that has sufficiently many outputs, the quantum query complexity is at least the seventh root of the classical randomized query complexity. Our query oracle $O_p: [S] \to [n]$ has $n$ outputs with tend to infinity when $n$ is large; the distribution $p$ is invariant under permutations on $[S]$ since $p_i = |\{s \in [S] : O_p(s) = i\}|/S$ is invariant for all $i$; Rényi entropy is invariant under permutations on $[n]$ since it does not depend on the order of $p_i$. Therefore, our problem satisfies the requirements from [1], hence $\Omega(n^{\frac{1}{\alpha} - o(1)})$ is a lower bound of the query complexity of Rényi entropy.

Second, consider the collision problem where you are given either a 2-to-1 function or a 1-to-1 function and decide which is the case. If the cardinality of the domain is $n$, then a 2-to-1 function
corresponds to the distribution \( \left( \frac{2}{n}, \ldots, \frac{2}{n} \right) \) with half \( \frac{2}{n} \) and half 0, whose \( \alpha \)-Rényi entropy is \( \log(n/2) \) for all \( \alpha \) (including Shannon entropy when \( \alpha = 1 \)); a 1-to-1 function corresponds to the uniform distribution \( \left( \frac{1}{n}, \frac{1}{n}, \ldots, \frac{1}{n} \right) \), whose \( \alpha \)-Rényi entropy is \( \log n \) for all \( \alpha \). The difference of these two cases is \( | \log n - \log(n/2) | = \log 2 \), which is a constant. Because references \([2, 22]\) have shown that \( \Omega(n^{1/3}) \) is a lower bound of the collision problem, it is also a lower bound of the quantum query complexity of \( \alpha \)-Rényi entropy.

Third, Theorem 6 of \([24]\) shows that the quantum query complexity of \( \alpha \)-Rényi entropy with \( \alpha \geq 2, \alpha \in \mathbb{N} \) is \( \Omega(n^{1/α - o(1)}) \). The proof is to reduce \( \alpha \)-Rényi entropy to the Hamming weight problem \([27]\), and we observe that the reduction actually works for all \( \alpha > 1 \). For min-entropy, the same reduction gives \( \Omega(\sqrt{n}) \) as a lower bound.

Combining the three points above and taking the maximum, we see that the quantum query complexity of \( \alpha \)-Rényi entropy estimation is \( \Omega(n^{1/α - o(1)}) \) if \( 0 < \alpha < \frac{3}{2} \), \( \Omega(n^{\frac{3}{4}}) \) if \( \frac{4}{7} \leq \alpha \leq 3 \), \( \Omega(n^{2 - \frac{2}{\alpha}}) \) if \( 3 \leq \alpha < \infty \), and \( \Omega(\sqrt{n}) \) if \( \alpha = \infty \).

The lower bounds in Theorem 7.1 assume \( \epsilon = \Theta(1) \). To give better dependence in \( \epsilon \), we use the polynomial method \([7]\) to show quantum lower bounds for entropy estimation. Inspired by the symmetrization technique in \([22]\), we obtain a bivariate polynomial whose degree is at most two times the corresponding quantum query complexity. Next, similar to \([27]\), we apply Paturi’s lemma \([33]\) to give a lower bound of the degree of the polynomial. To be more specific, we prove:

**Proposition 7.1.** The quantum query complexity of estimating min-entropy within error \( \epsilon \) is \( \Omega(\sqrt{n}/\epsilon) \).

**Proposition 7.2.** When \( \alpha > 1 \), the quantum query complexity of estimating \( \alpha \)-Rényi entropy within error \( \epsilon \) is \( \Omega(n^{1/α - \frac{2}{\epsilon}}) \).

Without loss of generality, we assume that the oracle \( O_P \) in (1.5) satisfies \( n|S \), otherwise consider the oracle \( O'_P : [Sn] \rightarrow [n] \) such that \( O'_P(s + Sl) = O_P(s) \) for all \( s \in [S] \) and \( l \in [n] \); this gives an oracle for the same distribution.

We consider the special case where the probabilities \( \{p_i\}_{i=1}^n \) takes at most two different values; to integrate the probabilities, we assume the existence of two integers \( c, d \) where \( c \in \{1, \ldots, n-1\} \), such that \( p_i = \frac{1}{n} - \frac{d}{S} \) for \( n-c \) different \( i \)'s in \( \{1, \ldots, n\} \), and \( p_i = \frac{1}{n} + \frac{(n-c)d}{cS} \) for the other \( c \) \( i \)'s in \( \{1, \ldots, n\} \).

**Proof of Proposition 7.1.** Following the symmetrization technique in \([22]\), we obtain a bivariate polynomial \( Q(c, d) \) where such that the degree of \( Q \) is at most two times the quantum query complexity of min-entropy estimation, and:

1. \( c \in \{1, \ldots, n-1\} \) and \( d \in \{-\left\lfloor \frac{Sc}{n(n-c)} \right\rfloor, \ldots, \left\lfloor \frac{S}{n} \right\rfloor \} \). This is because \( p_i \geq 0 \) for all \( i \in [n] \).

2. \( 0 \leq Q(c, d) \leq 1 \) if \( c \not| nd \). Only if \( c\not| nd \), \( S \cdot \left( \frac{1}{n} + \frac{(n-c)d}{cS} \right) \) is an integer and the distribution \( \{p_i\}_{i=1}^n \) is valid under our model in (1.5).

Furthermore, we consider the property testing problem of determining whether \( \max_i p_i = \frac{1}{n} \) or \( \max_i p_i \geq \frac{1+\epsilon}{n} \), where the success probability should be at most \( 1/3 \) for the former case and at least \( 2/3 \) for the latter case. As a result,

1. \( 0 \leq Q(c, 0) \leq 1/3 \): In this case, \( p_i = \frac{1}{n} \) for all \( i \in [n] \).

2. \( 2/3 \leq Q(c, d) \leq 1 \) if \( c\not| nd \), \( \frac{(n-c)d}{Sc} \geq \frac{\epsilon}{n} \): In this case, \( \exists i \) such that \( p_i = \frac{1}{n} + \frac{(n-c)d}{cS} \geq \frac{1+\epsilon}{n} \).
• $2/3 \leq Q(c, d) \leq 1$ if $c \neq nd$, $d \leq -\epsilon S/n$: In this case, $\exists i$ such that $p_i = 1 - \frac{d}{S} \geq \frac{1 + \epsilon}{n}$.

Therefore, we have

• $0 \leq Q(1, d) \leq 1$ for $d \in \{-\lfloor \frac{S}{n(n-1)} \rfloor, \ldots, \frac{S}{n} \}$;
• $0 \leq Q(1, 0) \leq 1/3$;
• $2/3 \leq Q(1, d) \leq 1$ for $d \in \{-\lfloor \frac{S}{n(n-1)} \rfloor, \ldots, -\lceil \epsilon S \rceil \} \cup \{ \lceil \frac{\epsilon S}{n(n-1)} \rceil, \ldots, \frac{S}{n} \}$.

Using Paturi’s lower bound [33], we have

$$\deg_d Q(1, d) \geq \Omega \left( \frac{\sqrt{\lfloor \frac{S}{n(n-1)} \rfloor \cdot \frac{S}{n}}}{\epsilon S/n(n-1)} \right) = \Omega \left( \frac{\sqrt{n}}{\epsilon} \right).$$

(7.1)

Therefore, $\deg Q(c, d) \geq \deg_d Q(1, d) = \Omega(\sqrt{n}/\epsilon)$.

\[\square\]

Proof of Proposition 7.2. The proof is similar to that of Proposition 7.1. Following the symmetrization technique, we still obtain a bivariate polynomial $Q(c, d)$ where such that the degree of $Q$ is at most two times the query complexity of min-entropy estimation, and $c \in \{1, \ldots, n-1\}$, $d \in \{-\lfloor \frac{S}{n(n-c)} \rfloor, \ldots, \frac{S}{n} \}$, $0 \leq Q(c, d) \leq 1$ if $c \neq nd$. Furthermore, we consider the property testing problem of determining whether $\sum_{i \in [n]} p_i^i \leq \frac{1}{n^\alpha - 1}$ or $\sum_{i \in [n]} p_i^i \geq \frac{2 + 2\epsilon}{n^\alpha - 1}$, where the success probability should be at most $1/3$ for the former case and at least $2/3$ for the latter case. We also assume $c = 1$. On the one hand, when $0 \leq d \leq \lfloor \frac{n^\alpha - 1}{n-1} \cdot \frac{S}{n} \rfloor$, we have $\frac{1}{n} + \frac{(n-1)d}{S} \leq \frac{1}{n^{1-\alpha}}$, and

$$\sum_{i \in [n]} p_i^i \leq \left( \frac{1}{n^{1-\alpha}} \right)^\alpha + (n-1) \left( \frac{1}{n-1} \left( 1 - \frac{1}{n^{1-\alpha}} \right) \right)^\alpha \leq \frac{2}{n^{\alpha-1}}.$$  

(7.2)

On the other hand, because $(1 + m)^\alpha \approx 1 + m\alpha$ when $m = o(1)$, we have

$$\left( \frac{1}{n^{1-\alpha}} + \frac{3\epsilon}{\alpha n^{1-\alpha}} \right)^\alpha + (n-1) \left( \frac{1}{n-1} \left( 1 - \frac{1}{n^{1-\alpha}} - \frac{3\epsilon}{\alpha n^{1-\alpha}} \right) \right)^\alpha = \frac{2 + 2\epsilon}{n^{\alpha-1}}$$

(7.3)

$$\approx \frac{1}{n^{\alpha-1}} \left( 1 + 3\epsilon - \frac{\alpha}{n^{1-\alpha}} \right)^\alpha - \frac{3\epsilon}{\alpha n^{1-\alpha}} - (2 + 2\epsilon)$$

(7.4)

$$= \frac{1}{n^{\alpha-1}} \left( \epsilon - \frac{\alpha + 3\epsilon}{n^{1-\alpha}} \right)$$

(7.5)

$$\geq 0$$

(7.6)

for large enough $n$. As a result, when $d \geq \lfloor \frac{(1 + 3\epsilon/\alpha)n^{1-\alpha} - 1}{n-1} \cdot \frac{S}{n} \rfloor$, we have $\sum_{i \in [n]} p_i^i \geq \frac{2+2\epsilon}{n^{\alpha-1}}$.

Therefore, we have

• $0 \leq Q(1, d) \leq 1$ for $d \in \{0, \ldots, \frac{S}{n} \}$;
• $0 \leq Q(1, d) \leq 1/3$ for $d \in \{0, \ldots, \lfloor \frac{n^{1-\alpha} - 1}{n-1} \cdot \frac{S}{n} \rfloor \}$;
• $2/3 \leq Q(1, d) \leq 1$ for $d \in \{ \lfloor \frac{(1 + 3\epsilon/\alpha)n^{1-\alpha} - 1}{n-1} \cdot \frac{S}{n} \rfloor, \ldots, \frac{S}{n} \}$.
Using Paturi’s lower bound [33], we have
\[
\deg_d Q(1, d) \geq \Omega \left( \frac{1}{n} \sqrt{n^{1/\alpha - 1} \cdot S_n \left( \frac{S_n}{n} - \left\lfloor \frac{n^{1/\alpha - 1} \cdot S_n}{n} \right\rfloor \right)} \right) = \Omega \left( \frac{\alpha n^{1/2 - 1/2}}{\xi} \right).
\] (7.7)

Therefore, \( \deg Q(c, d) \geq \deg_d Q(1, d) = \Omega (\alpha n^{1/2 - 1/2} / \xi) \).

Technically, our proofs only focus on the degree in \( d \) for \( c = 1 \), but in general it is possible to prove a better lower bound when analyzing the degree of the polynomial in \( c \) and \( d \) together. We leave this as an open problem.

Acknowledgements

We thank Andrew M. Childs for discussions that inspired the proof of Theorem 5.1, and general suggestions on our manuscript; we thank Yanjun Han for introducing us classical references related to Shannon and Rényi entropy estimation, in particular his papers [17, 20, 21]. We also thank anonymous reviewers for helpful comments on an earlier version of this paper. TL acknowledges support from NSF CCF-1526380.

Appendices

Throughout the appendices, we consider a discrete distribution \( \{ p_i \}_{i=1}^{n} \) on \([n]\), and \( P_\alpha(p) := \sum_{i=1}^{n} p_i^\alpha \) represents the \( \alpha \)-power sum of \( p \). In the analyses of our algorithms, ‘\( \log \)’ is natural logarithm; ‘\( \approx \)’ omits lower order terms.

A Theorem 2.2: Multiplicative quantum Chebyshev inequality

The main technique that we use is Lemma 4 in [25], which approximates a random variable with an additive error as long as its second-moment is bounded:

Lemma A.1 (Lemma 4 in [25]). Assume \( A \) is a quantum algorithm that outputs a random variable \( X \). Then for \( \epsilon > 0 \), by using \( O((1/\epsilon) \log^{3/2}(1/\epsilon) \log \log(1/\epsilon)) \) executions of \( A \) and \( \mathcal{A}^{-1} \), Algorithm 2 in [25] outputs an estimate \( \tilde{E}[X] \) of \( E[X] \) such that
\[
\Pr \left[ \left| \tilde{E}[X] - E[X] \right| \geq \epsilon (\sqrt{E[X^2]} + 1)^2 \right] \leq 1/40.
\] (A.1)

(The original error probability in (A.1) is 1/5, but it is not hard to be improved to 1/40 by multiplying a constant to the parameters in Lemma 4 in [25].)

Based on Lemma A.1 and inspired by Algorithm 3 and Theorem 5 in [25], we propose Algorithm 6.

Proof of Theorem 2.2. We assume that \( E[X] \in [a, b] \) always holds in the following discussion, which happens with probability at least 0.9. As a result, \( \text{Var}[X] \leq \sigma^2 E[X]^2 \leq \sigma^2 b^2 \). By Chebyshev’s inequality,
\[
\Pr \left[ \left| \bar{m} - E[X]/\sigma b \right| \geq 4 \right] \leq 1/16.
\] (A.2)
Algorithm 6: Estimate $\mathbb{E}[X]$ within multiplicative error $\epsilon$.

1. Run the algorithm that obtains $a, b$ such that $\mathbb{E}[X] \in [a, b]$ with probability at least 0.9;
2. Set $A' = A/\sigma b$;
3. Run $A'$ once and denote $\tilde{m}$ to be the output. Set $B = A' - \tilde{m}$;
4. Let $B_-$ be the algorithm that calls $B$ once; if $B$ outputs $x \geq 0$ then $B_-$ outputs 0, and if $B$ outputs $x < 0$ then $B_-$ outputs $x$. Similarly, let $B_+$ be the algorithm such that if $B$ outputs $x < 0$ then $B_+$ outputs 0, and if $B$ outputs $x \geq 0$ then $B_+$ outputs $x$;
5. Apply Lemma A.1 to $-B_-/6$ and $B_+/6$ with accuracy $\frac{\epsilon a}{48\sigma b}$ and failure probability 1/40, from which we obtain estimates $\bar{\mu}_-$ and $\bar{\mu}_+$, respectively;
6. Output $\mathbb{E}[X] = \sigma b(\tilde{m} - 6\bar{\mu}_- + 6\bar{\mu}_+)$;

Therefore, with probability at least 15/16 we have $|\tilde{m} - \mathbb{E}[X/\sigma b]| \leq 4$. Denote $X_B = \frac{X}{\sigma b} - \tilde{m}$, which is the random variable outputted by $B$; $X_{B,+} := \max\{X_B, 0\}$ is then the output of $B_+$ and $X_{B,-} := \min\{X_B, 0\}$ is the output of $B_-$. Assuming $|\tilde{m} - \mathbb{E}[X/\sigma b]| \leq 4$, we have

\[
\mathbb{E}[X_B^2] = \mathbb{E}\left[\left(\frac{X}{\sigma b} - \mathbb{E}\left[\frac{X}{\sigma b}\right]\right)^2\right] + \mathbb{E}\left[\left(\mathbb{E}\left[\frac{X}{\sigma b}\right] - \tilde{m}\right)^2\right]
\]

\[
\leq 2\mathbb{E}\left[\left(\frac{X}{\sigma b} - \mathbb{E}\left[\frac{X}{\sigma b}\right]\right)^2\right] + 2\mathbb{E}\left[\left(\mathbb{E}\left[\frac{X}{\sigma b}\right] - \tilde{m}\right)^2\right]
\]

\[
\leq 2(1^2 + 4^2) = 34.
\]

Therefore, $\mathbb{E}[X_B^2/6^2] \leq 34/36 < 1$, hence $\mathbb{E}[(X_{B,+}/6)^2] < 1$ and $\mathbb{E}[-(X_{B,-}/6)^2] < 1$. By Lemma A.1, we have

\[
|\bar{\mu}_- - \mathbb{E}[-X_{B,-}/6]| \leq \frac{\epsilon a}{12\sigma b} \quad \text{and} \quad |\bar{\mu}_+ - \mathbb{E}[X_{B,+}/6]| \leq \frac{\epsilon a}{12\sigma b}
\]

both with failure probability at most 1/40. Because

\[
\mathbb{E}[X] = \sigma b(\tilde{m} + \mathbb{E}[X_B]) = \sigma b(\tilde{m} + \mathbb{E}[X_{B,+}] - \mathbb{E}[-X_{B,-}]),
\]

with probability at least $0.9 \cdot 15/16 \cdot (1 - 1/40)^2 > 4/5$, we have

\[
|\tilde{m} - \bar{\mu}_- - \mathbb{E}[-X_{B,-}/6]| + 6|\bar{\mu}_+ - \mathbb{E}[X_{B,+}/6]| \leq \sigma b \cdot 2 \cdot 6 \cdot \frac{\epsilon a}{12\sigma b} = \epsilon a \leq \epsilon \mathbb{E}(X).
\]

\[
\square
\]

B Complete proof of Theorem 3.1: Shannon entropy estimation

To complete the proof of Theorem 3.1, we prove:

\[
\sum_{t \in S_0} p_t \mathbb{E}[| \log \tilde{p}_t - \log p_t |] = O\left(\frac{s_0}{2m}\right).
\]

To achieve this, we need the following lemma:

Lemma B.1. For $0 < t \leq 1$ and $x \in (0, t]$,

\[
h(x) := x(\log t - \log x) \leq t/e.
\]

Furthermore, $h(x)$ reaches its maximum $t/e$ at $x = t/e$, is an increasing function when $x \in (0, t/e)$, and is a decreasing function when $x \in (t/e, t)$.
Proof. We have $h'(x) = \log t - \log x - 1$. Therefore, when $x \in (0, t/e)$, $h'(x) > 0$ hence $h(x)$ is an increasing function; when $x \in (t/e, t)$, $h'(x) < 0$ hence $h(x)$ is a decreasing function; when $x = t/e$, $h'(x) = 0$ and $h$ reaches its maximum $t/e$.

Proof of (B.1). Since $i \in S_0$, we can write $p_i = \sin^2(\theta_i \pi)$ where $0 < \theta_i \leq 1/2^{n+1}$. By Theorem 2.3, for any $l \in \{1, 2, \ldots, 2^m - 1\}$ the output of EstAmp taking $2^m$ queries satisfies

$$\Pr \left[ \tilde{p}_i = \sin^2 \left( \frac{\pi}{2^{m+1}} \right) \right] = \frac{\sin^2 \left( \frac{2^m \theta}{2^{m+1}} \right)}{2^{2m} \sin^2 \left( \frac{\theta}{2^{m+1}} \right)} \leq 1; \quad \text{(B.3)}$$

$$\Pr \left[ \tilde{p}_i = \sin^2 \left( \frac{l\pi}{2^{m+1}} \right) \right] = \frac{\sin^2 \left( \frac{2^m \left( \frac{l}{2^{m+1}} - \theta_i \right) \pi}{2^{m+1} \sin^2 \left( \frac{l}{2^{m+1}} - \theta_i \right) \pi} \right)}{(2^{m+1} \left( \frac{l}{2^{m+1}} - \theta_i \right) \pi)^2} \leq \frac{1}{(2^{m+1} \left( \frac{l}{2^{m+1}} - \theta_i \right) \pi)^2}. \quad \text{(B.4)}$$

Combining (B.3), (B.4), and Lemma B.1, for any $i \in S_0$ we have

$$p_i \mathbb{E} \left[ \left| \log \tilde{p}_i - \log p_i \right| \right] \leq 1 \cdot p_i \left( \log \sin^2 \left( \frac{\pi}{2^{m+1}} \right) - \log p_i \right) + \sum_{l=1}^{2^m-1} p_i \frac{\log \sin^2 \left( \frac{l\pi}{2^{m+1}} \right)}{2^{2m} \left( \frac{l}{2^{m+1}} - \theta_i \right)^2} \leq \frac{\sin^2 \left( \frac{\pi}{2^{m+1}} \right)}{e} + \sum_{l=1}^{2^m-1} \frac{1}{(2l-1)^2} \cdot \sin^2 \left( \frac{l\pi}{2^{m+1}} \right) \left( \log \sin^2 \left( \frac{l\pi}{2^{m+1}} \right) - \log \sin^2 \left( \frac{\pi}{2^{m+1}} \right) \right), \quad \text{(B.5)}$$

$$\leq \frac{\pi^2}{4e} \cdot \frac{1}{2^{2m}} + \frac{\pi^2}{2^{2m}} \cdot \frac{1}{4} \sum_{l=1}^{2^m-1} \frac{1}{(2l-1)^2} \log \left( \frac{\sin \left( \frac{l\pi}{2^{m+1}} \right)}{\sin \left( \frac{\pi}{2^{m+1}} \right)} \right)^2 \leq \frac{\pi^2}{4e} \cdot \frac{1}{2^{2m}} + \frac{\pi^2}{2^{2m}} \cdot \frac{\log 2l}{(2l-1)^2} \leq O \left( \frac{1}{2^{2m}} \right), \quad \text{(B.6)}$$

where (B.5) comes from (B.3) and (B.4), (B.6) and (B.7) comes from Lemma B.1, (B.8) holds because $\sin^2 \left( \frac{l\pi}{2^{m+1}} \right) \leq 4l^2 \sin^2 \left( \frac{\pi}{2^{m+1}} \right)$, and (B.9) holds because $\sum_{l=1}^{\infty} \frac{\log l}{(2l-1)^2} = O(1)$. Consequently,

$$\sum_{i \in S_0} p_i \mathbb{E} \left[ \left| \log \tilde{p}_i - \log p_i \right| \right] = O \left( \frac{1}{2^{2m}} \right) \cdot \frac{s_0}{2^{2m}} = O \left( \frac{s_0}{2^{2m}} \right). \quad \text{(B.10)}$$

For $j \in \{1, 2, \ldots, m\}$, the dominating term is the case where the angles of $\tilde{p}_i$ and $p_i$ fall into the same interval of length $\frac{1}{2^m}$, and as a result $\left| \log \tilde{p}_i - \log p_i \right| = O \left( \frac{1}{2^j} \right)$. Using similar techniques to the proof of above to bound the tail, (3.2) follows.

## C Theorem 5.1: Integer Rényi entropy estimation

Our proof of Theorem 5.1 is inspired by the proof of Theorem 5 in [24].

Proof. Because $O_p$ takes values in $[n]$, by pigeonhole principle, for any $s_1, \ldots, s_{\alpha n} \in [S]$ there exists a $\alpha$-wise collision among $O_p(s_1), \ldots, O_p(s_{\alpha n})$. Therefore, Line 5 terminates the first loop with some $l \leq 2^{\log_2 \alpha n}$ with probability at least $(1 - 1/10 |\log_2 \alpha n|) |\log_2 \alpha n| \geq e^{-1/10} > 0.9$. 

16
Moreover, tighter bounds on \( l \) are established next. On the one hand, by Chebyshev’s inequality and Fact 5.1, the probability that the first for-loop fails to terminate when \( l \leq \frac{A}{P_{\alpha}(p)} \) for some constant \( A > 0 \) is at most

\[
\Pr \left[ C(s_1, \ldots, s_l) = 0 \right] \leq \frac{\Var[C(s_1, \ldots, s_l)]}{\E[C(s_1, \ldots, s_l)]^2} = O \left( \frac{1}{l^{2\alpha}P_{\alpha}(p)^2} \right) = O \left( \frac{1}{B^{2\alpha}} \right). \tag{C.1}
\]

Therefore, taking a large enough \( B \) ensures that \( l = \Omega \left( \frac{1}{P_{\alpha}(p)^{1/\alpha}} \right) \) with failure probability at most \( 1/20 \). On the other hand, by Markov’s inequality and Fact 5.1, we have

\[
\Pr \left[ C(s_1, \ldots, s_l) \geq 1 \right] \leq \E[C(s_1, \ldots, s_l)] = O \left( l^{\alpha}P_{\alpha}(p) \right). \tag{C.2}
\]

As a result, the probability that the first for-loop terminates when \( l \leq \frac{A}{P_{\alpha}(p)^{1/\alpha}} \) for some constant \( A > 0 \) is at most

\[
O \left( P_{\alpha}(p) \right) \cdot \sum_{i=0}^{\left\lfloor \log_2 \left( \frac{A}{P_{\alpha}(p)^{1/\alpha}} \right) \right\rfloor} 2^{i\alpha} = O \left( A^\alpha \right). \tag{C.3}
\]

Therefore, taking a small enough \( A > 0 \) ensures that \( l = \Omega \left( \frac{1}{P_{\alpha}(p)^{1/\alpha}} \right) \) with failure probability at most \( 1/20 \). In all, we have \( l = \Theta \left( \frac{1}{P_{\alpha}(p)^{1/\alpha}} \right) \) with probability at least 0.9.

By Fact 5.1, the output \( \E[\tilde{P}_{\alpha}(p)] \) in Line 10 of Algorithm 4 satisfies

\[
\E[\tilde{P}_{\alpha}(p)] = \frac{1}{M(\binom{\nu}{i})} \sum_{r=1}^{M} \E[C^{(r)}] = P_{\alpha}(p), \quad \Var[\tilde{P}_{\alpha}(p)] = \frac{1}{(M(\binom{\nu}{i}))^2} \sum_{r=1}^{M} \Var[C^{(r)}] = O \left( \frac{1}{M^{2\alpha}} \right). \tag{C.4}
\]

Therefore, by Chebyshev’s inequality and recall \( l = \Theta \left( \frac{1}{P_{\alpha}(p)^{1/\alpha}} \right) \), we have

\[
\Pr \left[ |\tilde{P}_{\alpha}(p) - P_{\alpha}(p)| \geq \epsilon P_{\alpha}(p) \right] \leq O \left( \frac{1}{M^{2\alpha} \epsilon^2 P_{\alpha}(p)^2} \right) = O \left( \frac{1}{K} \right). \tag{C.5}
\]

Taking a large enough constant \( K \) in Line 6 of Algorithm 4, we have \( \Pr \left[ |\tilde{P}_{\alpha}(p) - P_{\alpha}(p)| \leq \epsilon P_{\alpha}(p) \right] \geq 0.9 \). In all, with probability at least \( 0.9 \times 0.9 \times 0.9 > 2/3 \), \( \tilde{P}_{\alpha}(p) \) approximates \( P_{\alpha}(p) \) within multiplicative error \( \epsilon \).

For the rest of the proof, it suffices to compute the query complexity of Algorithm 4. Because the \( \alpha \)-distinctness algorithm on \( m \) elements in [8] takes \( \tilde{O}(m^\nu \log(1/\delta)) \) quantum queries when the success probability is \( 1 - \delta \), the first for-loop in Algorithm 4 takes \( \sum_{i=0}^{\log_2 l} O(2^\alpha \log \log_2 \alpha n) = \tilde{O}(l^\nu) = \tilde{O}(n^{\nu(1-1/\alpha)}) \) quantum queries because

\[
l = \Theta \left( \frac{1}{P_{\alpha}(p)^{1/\alpha}} \right) = O \left( n^{1-1/\alpha} \right), \tag{C.6}
\]

following from \( P_{\alpha}(p) \geq n^{1-\alpha} \). The second for-loop takes \( \lceil K/\epsilon^2 \rceil \cdot O(l^{\nu} \log(l/\epsilon^2)) = \tilde{O} \left( \frac{n^{\nu(1-1/\alpha)}}{\epsilon^2} \right) \) quantum queries by Fact 5.1 and (C.6). In total, the number of quantum queries is \( \tilde{O} \left( \frac{n^{\nu(1-1/\alpha)}}{\epsilon^2} \right) \) as claimed.

**Remark C.1.** In Theorem 5.1, we regard \( \alpha \) as a constant, i.e., the query complexity \( \tilde{O} \left( \frac{n^{\nu(1-1/\alpha)}}{\epsilon^2} \right) \) hides the multiple in \( \alpha \). In fact, by analyzing the dependence in \( \alpha \) carefully in the above proof, the query complexity of Algorithm 4 is actually

\[
\tilde{O} \left( \alpha^{\nu^2} \cdot \frac{n^{\nu(1-1/\alpha)}}{\epsilon^2} \right). \tag{C.7}
\]
The dependence in $\alpha$ is super-exponential; therefore, Algorithm 4 is not good enough to approximate min-entropy (i.e., $\alpha = \infty$). As a result, we give the quantum algorithm for estimating min-entropy separately (see Section 6 and Appendix F).

## D Theorem 4.1: Non-integer Rényi entropy estimation, $\alpha > 1$

In this Section, we give the complete proof of Theorem 4.1, which is rewritten below:

**Theorem D.1** (Rewriting Theorem 4.1). The output of Algorithm 3 approximates $P_\alpha(p)$ within a multiplicative error $0 < \epsilon \leq 1/4$ with success probability at least $1 - \delta$ for some $\delta > 0$ using $\tilde{O}(n^{1-\frac{1}{2\alpha}})$ quantum queries to $p$, where $\tilde{O}$ hides polynomials terms of $\log n$, $\log 1/\epsilon$, and $\log 1/\delta$.

For convenience, Algorithm 3 is also rewritten here:

---

**Algorithm 7: Rewriting Algorithm 3**

Regard the following subroutine as $\mathcal{A}$:

- Draw a sample $i \in [n]$ according to $p$;
- Use the amplitude estimation procedure $\text{EstAmp}$ with $M = 2^{[\log_2(\frac{\alpha}{e} \log(\frac{e}{\epsilon}))]+1}$ queries to obtain an estimate $\tilde{p}_i$ of $p_i$;
- Output $\tilde{x}_i = \tilde{p}_i^{-\alpha}$.

1. **Input** $\alpha$, multiplicative error $\epsilon$, and failure probability $\delta$;
2. **if** $\alpha < 1 + \frac{1}{\log n}$ **then**
   3. Take $a = \frac{1}{\epsilon}$ and $b = 1$ as lower and upper bounds on $P_\alpha(p)$, respectively;
4. **else**
   5. Recursively call Algorithm 3 with $\alpha' = \alpha(1 + \frac{1}{\log n})^{-1}$, $\epsilon = 1/4$, and $\delta = \frac{1}{12 \log n \log \alpha}$ therein to give an estimate $\tilde{P}_{\alpha'}(p)$ of $P_{\alpha'}(p)$. For simplicity, denote $P := \tilde{P}_{\alpha'}(p)$. Take $a = (\frac{3P/4}{\frac{1}{\epsilon} - \frac{1}{2\alpha}})$ and $b = (\frac{4P}{3})^{1 + \frac{1}{\log n}}$ as lower and upper bounds on $P_\alpha(p)$, respectively;
6. **Set** $l = \Theta(\frac{1}{\epsilon} \log(\frac{1}{\epsilon}) \log(\frac{1}{\epsilon}))$;
7. Use $\mathcal{A}$ for $l$ executions in Theorem 2.2 using $a$ and $b$ as auxiliary information and output an estimation of $P_\alpha(p)$;
8. **Run** Line 1 to Line 7 for $\lceil 48 \log \frac{1}{\delta} \rceil$ executions and take the median of all outputs in Line 7, denoted as $\tilde{P}_\alpha(p)$. Output $\tilde{P}_\alpha(p)$.

---

**Proof of Theorem 4.1.** First, if the estimate $\tilde{p}_i$ in the subroutine $\mathcal{A}$ of Algorithm 3 were precisely accurate, the expectation of the subroutine's output would be $E := \sum_{i \in [n]} p_i \cdot p_i^{-\alpha} = P_\alpha(p)$. To be precise, we bound how far the actual expectation of the subroutine’s output $\tilde{E}$ is from the exact value $P_\alpha(p)$. In Lemma D.1, we show that when taking $M = 2^{[\log_2(\frac{\alpha}{e} \log(\frac{e}{\epsilon}))]+1}$ queries in $\text{EstAmp}$, we have $|\tilde{E} - E| = O(\epsilon E)$.

As a result, to approximate $P_\alpha(p)$ within multiplicative error $\Theta(\epsilon)$, it is equivalent to approximate $\tilde{E}$ within multiplicative error $\Theta(\epsilon)$. Recall Theorem 2.2 (multiplicative quantum Chebyshev inequality) showed that if the variance of the random variable outputted by $\mathcal{A}$ is at most $\sigma^2 \tilde{E}^2$ for a known $\sigma$, and if we can obtain two values $a, b$ such that $\tilde{E} \in [a, b]$ with probability at least 0.9, then $\tilde{O}(\sigma b/\epsilon a)$ executions of $\mathcal{A}$ suffices to approximate $\tilde{E}$ within multiplicative error $\epsilon$ with success
probability at least 0.8. In the main body of the algorithm (Line 1 to Line 8), we use Theorem 2.2 to approximate $\tilde{E}$.

On the one hand, in Lemma D.2, we show that for $\alpha > 1$ and large enough $n$, the variance is at most $5n^{1-1/\alpha}E^2$ with probability at least $\frac{8}{\alpha^2}$. This gives $\sigma = \sqrt{5n^{1-1/\alpha}} = O(n^{1/2-1/2\alpha})$.

On the other hand, we need to compute the lower bound $a$ and upper bound $b$. A key observation (Lemma D.3) is that for any $0 < \alpha_1 < \alpha_2$, we have

$$\left(\sum_{i \in [n]} p_i^{\alpha_2}\right)^{\alpha_2} \leq \sum_{i \in [n]} p_i^{\alpha_1} \leq n^{1-\alpha_1/\alpha_2}\left(\sum_{i \in [n]} p_i^{\alpha_2}\right)^{\alpha_2}. \quad (D.1)$$

Because $n^{1/\log n} = e$, if $\frac{\alpha_2}{\alpha_1} = 1 + O\left(\frac{1}{\log n}\right)$, then

$$\sum_{i \in [n]} p_i^{\alpha_1} = \Theta\left(\left(\sum_{i \in [n]} p_i^{\alpha_2}\right)^{\alpha_2}\right). \quad (D.2)$$

As a result, we compute $a$ and $b$ by recursively calling Algorithm 3 to estimate $P_\alpha(p)$ for $\alpha' = \alpha/(1 + 1/\log n)$, which is used to compute the lower bound $a$ and upper bound $b$ in Line 5; the recursive call keeps until $\alpha < 1 + \frac{1}{\log n}$, when $a = \frac{1}{e}$ and $b = 1$ (as in Line 3) are simply lower and upper bounds on $P_\alpha(p)$ by (D.1).

To be precise, in Lemma D.4, we prove that $b/a < 4e = O(1)$, and with probability at least $1/e^{1/12} > 0.92$, $a$ and $b$ are indeed lower and upper bounds on $P_\alpha(p)$, respectively; furthermore, in Line 5, Algorithm 3 is recursively called at most $\log n \log \alpha$ times, and each recursive call takes at most $\tilde{O}(n^{1-\pi_e})$ queries. This promises that when we apply Corollary 2.2, the cost $C_{a,b}$ is dominated by the query cost from Algorithm 6.

Combining all points above, Corollary 2.2 approximates $\tilde{E}$ up to multiplicative error $\Theta(\epsilon)$ with success probability at least $\frac{3}{\pi^2} \cdot 0.92 \cdot 4/5 > 2/3$ using

$$\log n \log \alpha \cdot \tilde{O}\left(\frac{4e \cdot \sqrt{5n^{1-1/\alpha}}}{\epsilon}\right) \cdot 2^{\lceil \log_2 (\frac{\sqrt{\pi \log(\sqrt{\pi \log(\sqrt{\pi})))}}{\epsilon^2})\rceil} + 1 = \tilde{O}\left(\frac{n^{1-1/2\alpha}}{\epsilon^2}\right). \quad (D.3)$$

quantum queries. Together with $|\tilde{E} - E| = O(\epsilon E)$ and rescale $l, M$ by a large enough constant, Line 1 to Line 7 in Algorithm 3 approximates $E = P_\alpha(p)$ up to multiplicative error $\epsilon$ with success probability at least 2/3.

Finally, in Lemma D.5, we show that after repeating the procedure for $\lceil 48 \log \frac{1}{\delta} \rceil$ executions and taking the median $\tilde{P}_\alpha(p)$ (as in Line 8), the success probability that $\tilde{P}_\alpha(p)$ approximates $P_\alpha(p)$ within multiplicative error $\epsilon$ is boosted to $1 - \delta$. □

In the coming Subsections, we give proofs for the lemmas mentioned above.

### D.1 Expectation of $A$ is $\epsilon$-close to $P_\alpha(p)$

**Lemma D.1.** $|\tilde{E} - E| = O(\epsilon E)$.

**Proof of Lemma D.1.** For convenience, denote $m = \lceil \log_2(\sqrt{n}/\epsilon \log(\sqrt{n}/\epsilon)) \rceil + 1$, and $S_0, S_1, \ldots, S_m$ the same as in Section 3. We still have (3.1). By linearity of expectation,

$$|\tilde{E} - E| \leq \sum_{i \in [n]} p_i E\left[|p_i^{\alpha - 1} - p_i^{\alpha - 1}|\right] = \sum_{j=0}^{m} \sum_{i \in S_j} p_i E\left[|p_i^{\alpha - 1} - p_i^{\alpha - 1}|\right]. \quad (D.4)$$
Therefore, to prove $|\tilde{E} - E| = O(\epsilon E)$ it suffices to show
\[
\sum_{j=0}^{m} \sum_{i \in S_j} p_i \mathbb{E}[|p_i^{\alpha - 1} - p_i^{\alpha - 1}|] = O\left( \epsilon \sum_{i \in [n]} p_i^{\alpha} \right). \tag{D.5}
\]

For each $i \in [n]$ we write $p_i = \sin^2(\theta_i \pi)$. Assume $k \in \mathbb{Z}$ such that $k \leq 2^m \theta < k + 1$. By Theorem 2.3, for any $l \in \{1, 2, \ldots, \max\{k - 1, 2^m - k - 1\}\}$ the output of $\text{EstAmp}$ taking $2^n$ queries satisfies
\[
\Pr \left[ \tilde{p}_i = \sin^2 \left( \frac{(k + l + 1)\pi}{2m} \right) \right], \Pr \left[ \tilde{p}_i = \sin^2 \left( \frac{(k - l - 1)\pi}{2m} \right) \right] \leq \frac{1}{4l^2}. \tag{D.6}
\]
Furthermore, because $\sin \theta = \theta - O(\theta^3)$, $\cos \theta = 1 - O(\theta^2)$, and $(1 + \theta)2^{\alpha - 1} = 1 + (2\alpha - 1)\theta + o(\theta)$ for $\theta$ close to 0, we have
\[
(\sin((\theta_i + \frac{l}{2m})\pi))^2(2^{\alpha - 1}) - (\sin((\theta_i\pi))^2(2^{\alpha - 1}) = O\left( \frac{l}{2m} (\theta_i \pi)^{2\alpha - 3} \right). \tag{D.7}
\]
Combining (D.6), (D.7), and the fact that $\sum_{i=1}^{2^m} \frac{1}{l} = \Theta(m)$, we have
\[
\sum_{j=0}^{m} \sum_{i \in S_j} p_i \mathbb{E}[|p_i^{\alpha - 1} - p_i^{\alpha - 1}|] = O\left( \sum_{j=0}^{m} s_j \cdot \left( \frac{2^j}{2m \pi} \right)^2 \cdot 2 \sum_{l=1}^{2^m} \frac{1}{4l^2} \cdot \frac{l}{2m} \left( \frac{2^j}{2m \pi} \right)^{2\alpha - 3} \right) \tag{D.8}
\]
\[
= O\left( \frac{\pi^{2\alpha - 1} m}{2^{2\alpha m}} \cdot \sum_{j=0}^{m} s_j 2^{(2\alpha - 1)j} \right). \tag{D.9}
\]
On the other side,
\[
\epsilon \sum_{i \in [n]} p_i^{\alpha} = \Theta\left( \epsilon \sum_{j=0}^{m} s_j \cdot \left( \frac{2^j}{2m \pi} \right)^{2\alpha} \right) = \Theta\left( \frac{\epsilon \cdot \pi^{2\alpha}}{2^{2\alpha m}} \sum_{j=0}^{m} s_j 2^{2\alpha j} \right). \tag{D.10}
\]
Therefore, to prove equation (D.5), by (D.9) and (D.10) it suffices to prove
\[
\sum_{j=0}^{m} s_j 2^{(2\alpha - 1)j} = O\left( \frac{\epsilon}{m} \sum_{j=0}^{m} s_j 2^{2\alpha j} \right). \tag{D.11}
\]
Because $m = \lfloor \log_2(\frac{\sqrt{n}}{\epsilon} \log(\frac{\sqrt{n}}{\epsilon})) \rfloor + 1$, we have $\frac{2^m}{m} \geq \sqrt{n}$, hence $\frac{2^m}{\sqrt{n}} \geq \frac{m}{\epsilon}$. As a result, it suffices to prove
\[
\sum_{j=0}^{m} s_j 2^{(2\alpha - 1)j} = O\left( \frac{\sqrt{n}}{2m} \sum_{j=0}^{m} s_j 2^{2\alpha j} \right). \tag{D.12}
\]
If $\alpha \geq 3/2$, by Hölder’s inequality we have
\[
\left( \sum_{j=0}^{m} s_j \right)^{\frac{1}{2\alpha}} \left( \sum_{j=0}^{m} s_j 2^{2\alpha j} \right)^{\frac{2\alpha - 1}{2\alpha}} \geq \sum_{j=0}^{m} s_j 2^{(2\alpha - 1)j}. \tag{D.13}
\]
By equation (3.1), this gives
\[
\sqrt{n}^{\frac{1}{2\alpha - 1}} \left( \sum_{j=0}^{m} s_j 2^{2\alpha j} \right) \geq \left( \sum_{j=0}^{m} s_j 2^{(2\alpha - 1)j} \right) \left( \sum_{j=0}^{m} s_j 2^{(2\alpha - 1)j} \right)^{\frac{1}{2\alpha - 1}}. \tag{D.14}
\]

20
By Hölder’s inequality and also equation (3.1), we have
\[
\left(\sum_{j=0}^{m} s_j 2^{2\alpha j}\right)^{\frac{2\alpha-3}{2\alpha-1}} \left(\sum_{j=0}^{m} s_j 2^{(2\alpha-1)j}\right)^{\frac{2}{2\alpha-1}} \geq 2^{\sum_{j=0}^{m} s_j 2^{2j}} = \Theta(2^{2m}).
\] (D.15)

This is equivalent to
\[
\frac{1}{n}^{\frac{2\alpha-3}{2\alpha-1}} \left(\sum_{j=0}^{m} s_j 2^{(2\alpha-1)j}\right)^{\frac{1}{2\alpha-1}} \geq \Theta(2^{m}).
\] (D.16)

Combining (D.14) and (D.16), we get exactly (D.12).

If \(1 < \alpha < 3/2\), by Hölder’s inequality we have
\[
\left(\sum_{j=0}^{m} s_j 2^{2\alpha j}\right)^{\frac{1}{2\alpha}} = \left(\sum_{j=0}^{m} s_j \right)^{\frac{1}{2\alpha}} \geq \sum_{j=0}^{m} s_j 2^{2j};
\] (D.17)
\[
\left(\sum_{j=0}^{m} s_j 2^{2j}\right)^{\frac{2\alpha-1}{2\alpha}} \left(\sum_{j=0}^{m} s_j \right)^{\frac{\alpha-1}{2\alpha}} \geq \sum_{j=0}^{m} s_j 2^{(2\alpha-1)j}.
\] (D.18)

By equation (3.1), the two inequalities above give
\[
\sum_{j=0}^{m} s_j 2^{2\alpha j} \geq n^{1-\alpha} 2^{\alpha m} \quad \text{and} \quad \sum_{j=0}^{m} s_j 2^{(2\alpha-1)j} \leq n^{1.5-\alpha} 2^{(2\alpha-1)m},
\] (D.19)
which give (D.12). □

**D.2 Bound the variance of \(A\) by the square of its expectation**

**Lemma D.2.** With probability at least \(\frac{8}{\pi^2}\), the variance of the random variable outputted by \(A\) is at most \(5n^{1-1/\alpha}\tilde{E}^2\).

**Proof of Lemma D.2.** The expectation and variance of the output by \(A\) are \(\tilde{E} = \sum_{i=1}^{n} p_i \cdot \tilde{p}_i^{\alpha-1}\) and \(\sum_{i \in [n]} p_i \cdot (\tilde{p}_i^{\alpha-1})^2 - (\sum_{i \in [n]} p_i \cdot \tilde{p}_i^{\alpha-1})^2\), respectively. Therefore, it suffices to show that with probability at least \(\frac{8}{\pi^2}\),
\[
\sum_{i \in [n]} p_i \cdot (\tilde{p}_i^{\alpha-1})^2 \leq 5n^{1-1/\alpha} \left(\sum_{i=1}^{n} p_i \cdot \tilde{p}_i^{\alpha-1}\right)^2.
\] (D.20)

By Theorem 2.3, with probability at least \(\frac{8}{\pi^2}\), we have that for any \(i \in [n]\),
\[
|\tilde{p}_i - p_i| \leq \frac{2\pi \sqrt{p_i}}{2m} \leq \frac{\epsilon \pi \sqrt{p_i}}{\sqrt{n}}.
\] (D.21)

For convenience, denote \(p := p_{i^*}\) to be the maximal one among \(p_1, \ldots, p_n\), i.e., \(p = \max_{i \in \{1, \ldots, n\}} p_i\). We also denote \(\tilde{p} := \tilde{p}_i\). Then we have
\[
\frac{(\sum_{i=1}^{n} p_i \cdot \tilde{p}_i^{\alpha-1})^2}{\sum_{i \in [n]} p_i \cdot (\tilde{p}_i^{\alpha-1})^2} \geq \frac{(\sum_{i=1}^{n} p_i \cdot \tilde{p}_i^{\alpha-1})^2}{\tilde{p}^{\alpha-1} \cdot \sum_{i \in [n]} p_i \cdot \tilde{p}_i^{\alpha-1}} = \frac{\sum_{i=1}^{n} p_i \cdot \tilde{p}_i^{\alpha-1}}{\tilde{p}^{\alpha-1}}.
\] (D.22)
Furthermore, because \( x^\alpha \) is a convex function in \([0,1]\), by (D.21) and Jensen’s inequality we have

\[
\sum_{i=1}^{n} p_i \cdot \frac{x_i^{\alpha-1}}{\tilde{p}^{\alpha-1}} = \frac{p \cdot \tilde{p}^{\alpha-1} + \sum_{i \neq j} p_i \cdot \tilde{p}_i^{\alpha-1}}{\tilde{p}^{\alpha-1}} \geq \frac{p \left( p + \frac{\epsilon \sqrt{p}}{\sqrt{n}} \right)^{\alpha-1} + (n-1) \cdot \frac{1-p}{n-1} \cdot \frac{1-p}{n-1} - \frac{\epsilon \sqrt{1-p}}{\sqrt{n}}}{\tilde{p}^{\alpha-1}} \approx p + (1-p) \left( \frac{1-p - \epsilon \sqrt{1-p}}{np} \right)^{\alpha-1}.
\]

(D.23)

(D.24)

(D.25)

Therefore, it suffices to show that for large enough \( n \),

\[
p + (1-p) \left( \frac{1-p - \epsilon \sqrt{1-p}}{np} \right)^{\alpha-1} \geq 0.2n^{-1(1/\alpha)}.
\]

(D.26)

If \( p \geq 0.2n^{-1(1/\alpha)} \), equation (D.26) directly follows. If \( p < 0.2n^{-1(1/\alpha)} \),

\[
\lim_{n \to \infty} n^{1-1/\alpha} \cdot (1-p) \left( \frac{1-p - \epsilon \sqrt{1-p}}{np} \right)^{\alpha-1} \geq \lim_{n \to \infty} n^{1-1/\alpha} \left( 1 - 0.2n^{-1(1/\alpha)} \right) \left( \frac{1 - 0.2n^{-1(1/\alpha)} - \epsilon \sqrt{1-p}}{0.2n^{1/\alpha}} \right)^{\alpha-1} = \lim_{n \to \infty} \left( 1 - 0.2n^{-1(1/\alpha)} \right) \left( \frac{1 - 0.2n^{-1(1/\alpha)} - \epsilon \sqrt{1-p}}{0.2n^{1/\alpha}} \right)^{\alpha-1} = 1 \cdot \left( \frac{1 - \epsilon \sqrt{1-p}}{0.2} \right)^{\alpha-1} > 1 > 0.2,
\]

(D.27)

(D.28)

(D.29)

where (D.29) is true because \( \frac{1 - \epsilon \sqrt{1-p}}{0.2} > \frac{1 - 3.2/4}{0.2} = 1 \). Because (D.25) only omits lower order terms and the limit in (D.29) is a constant larger than 0.2, Lemma D.2 follows.

\[\square\]

### D.3 Give tight bounds on \( P_\alpha(p) \) by \( P_{\alpha'}(p) \)

**Lemma D.3.** For any distribution \((p_i)_{i=1}^{n}\) and \( 0 < \alpha_1 < \alpha_2 \), we have

\[
\left( \sum_{i \in [n]} p_i^{\alpha_2} \right)^{\frac{\alpha_1}{\alpha_2}} \leq \sum_{i \in [n]} p_i^{\alpha_1} \leq n^{1-\frac{\alpha_1}{\alpha_2}} \left( \sum_{i \in [n]} p_i^{\alpha_2} \right)^{\frac{\alpha_1}{\alpha_2}}.
\]

(D.30)

**Proof of Lemma D.3.** On the one hand, by the generalized mean inequality, we have

\[
\left( \frac{\sum_{i \in [n]} p_i^{\alpha_2}}{n} \right)^{\frac{1}{\alpha_2}} \geq \left( \frac{\sum_{i \in [n]} p_i^{\alpha_1}}{n} \right)^{\frac{1}{\alpha_1}},
\]

which gives the second inequality in (D.30).

On the other hand, since \( \frac{\alpha_1}{\alpha_2} \leq 1 \) and

\[
0 \leq \frac{p_i^{\alpha_2}}{\sum_{j \in [n]} p_j^{\alpha_2}} \leq 1 \quad \forall i \in [n],
\]

we have

\[
\sum_{i \in [n]} p_i^{\alpha_1} = \sum_{i \in [n]} \left( \frac{\sum_{j \in [n]} p_j^{\alpha_2}}{\sum_{i \in [n]} p_i^{\alpha_2}} \right)^{\frac{\alpha_1}{\alpha_2}} \geq \sum_{i \in [n]} \frac{p_i^{\alpha_2}}{\sum_{j \in [n]} p_j^{\alpha_2}} = 1,
\]

(D.33)

which is equivalent to the first inequality in (D.30). \[\square\]
D.4 Analyze the recursive calls

Lemma D.4. With probability at least 0.92, the \( a \) and \( b \) in Line 3 or Line 5 of Algorithm 3 are indeed lower and upper bounds on \( P_\alpha(p) \), respectively, and \( b/a = O(1) \); furthermore, in Line 5, Algorithm 3 is recursively called for at most \( \log n \log \alpha \) executions, and each recursive call takes at most \( \tilde{O}(n^{1 - \frac{1}{2\alpha}}) \) queries.

Proof of Lemma D.4. We decompose the proof into two parts:

- **In Line 5, Algorithm 3 is recursively called for at most \( \log n \log \alpha \) executions, and each recursive call takes at most \( \tilde{O}(n^{1 - \frac{1}{2\alpha}}) \) queries:**

  Because each recursive call of Algorithm 3 reduces \( \alpha \) by multiplying \( (1 + \frac{1}{\log n})^{-1} \), and the recursion ends when \( \alpha < 1 + \frac{1}{\log n} \), the total number of recursive calls is at most \( \log(\frac{1 + \frac{1}{\log n}}{\alpha}) \leq \log n \log \alpha \).

  When \( \alpha < 1 + \frac{1}{\log n} \), \( a \) and \( b \) are set in Line 3 and no extra queries are needed; when Line 5 calls \( \alpha(1 + \frac{1}{\log n})^{-k} \)-power sum estimation for some \( k \in \mathbb{N} \), by induction on \( k \), we see that this call takes at most \( \tilde{O}(n^{1 - \frac{1}{2\alpha}}) \) queries. As a result, when we apply Corollary 2.2, the cost \( C_{a,b} \) is dominated by the query cost from Algorithm 6.

- **With probability at least 0.92, \( a \) and \( b \) are lower and upper bounds on \( P_\alpha(p) \) respectively, and \( b/a = O(1) \):**

  When \( 1 < \alpha < 1 + \frac{1}{\log n} \), on the one hand we have \( \sum_{i=1}^{n} p_i^\alpha \leq \sum_{i=1}^{n} p_i = 1 \); on the other hand, because \( n^{\frac{1}{\log n}} = e \), by Lemma D.3 we have

  \[
  \sum_{i=1}^{n} p_i^\alpha \geq \left(\frac{\sum_{i=1}^{n} p_i}{n^{\alpha-1}}\right)^\alpha \geq \frac{1}{e}.
  \]  

  (D.34)

  Therefore, \( a = 1/e \) and \( b = 1 \) in Line 3 are lower and upper bounds on \( P_\alpha(p) \) respectively, and \( b/a = e = O(1) \).

  When \( \alpha > 1 + \frac{1}{\log n} \), for convenience denote \( \alpha' = \alpha(1 + \frac{1}{\log n})^{-1} \). As justified above, the total number of recursive calls in Line 5 is at most \( \log n \log \alpha \). Because we take \( \delta = \frac{1}{12 \log n \log \alpha} \) in Line 5, with probability at least

  \[
  \left(1 - \frac{1}{12 \log n \log \alpha}\right)^{\log n \log \alpha} \geq \frac{1}{e^{1/12}} > 0.92,
  \]  

  (D.35)

  the output of every recursive call is within \( 1/4 \)-multiplicative error. As a result, the \( P \) in Line 5 satisfies \( 3P/4 \leq \sum_{i=1}^{n} p_i^{\alpha'} \leq 4P/3 \). Combining this with Lemma D.3 and using \( n^{\frac{1}{\log n}} = e \), we have

  \[
  \frac{(3P/4)^{1 + \frac{1}{\log n}}}{e} \leq \sum_{i=1}^{n} p_i^{\alpha} \leq (4P/3)^{1 + \frac{1}{\log n}}.
  \]  

(D.36)
In other words, $a$ and $b$ are indeed lower and upper bounds on $P_\alpha(p)$, respectively. Furthermore, $b/a = O(1)$ because
\[
\frac{b}{a} = e \cdot \left(\frac{16}{9}\right)^{1 + \frac{1}{\log 3}} < 4e = O(1).
\] (D.37)

**D.5 Boost the success probability**

**Lemma D.5.** By repeating Line 1 to Line 7 in Algorithm 3 for $\lceil 48 \log \frac{1}{\delta} \rceil$ executions and taking the median $\tilde{P}_\alpha(p)$, the success probability is boosted to $1 - \delta$.

**Proof of Lemma D.5.** Denote the outputs after running Line 1 to Line 7 for the median $\tilde{P}_\alpha(p)$, for each $i \in \{1, \ldots, \lceil 48 \log \frac{1}{\delta} \rceil\}$, with probability at least $2/3$ we have
\[
|\tilde{P}_\alpha(p)^{(i)} - P_\alpha(p)| \leq \epsilon P_\alpha(p). \tag{D.38}
\]

For each $i \in \{1, \ldots, \lceil 48 \log \frac{1}{\delta} \rceil\}$, denote $X_i$ to be a Boolean random variable such that $X_i = 1$ if (D.38) holds, and $X_i = 0$ otherwise. Then $\Pr[X_i = 1] \geq 2/3$. Because in Line 8 the output $\tilde{P}_\alpha(p)$ is the median of all $\tilde{P}_\alpha(p)^{(1)}, \ldots, \tilde{P}_\alpha(p)^{(\lceil 48 \log \frac{1}{\delta} \rceil)}$, $|\tilde{P}_\alpha(p) - P_\alpha(p)| > \epsilon P_\alpha(p)$ leads to $\sum_{i=1}^{\lceil 48 \log \frac{1}{\delta} \rceil} X_i < \lceil 48 \log \frac{1}{\delta} \rceil/2$. On the other hand, by Chernoff bound we have
\[
\Pr \left[ \sum_{i=1}^{\lceil 48 \log \frac{1}{\delta} \rceil} X_i < \frac{\lceil 48 \log \frac{1}{\delta} \rceil}{2} \right] \leq \exp \left( -\frac{2/3 \lceil 48 \log \frac{1}{\delta} \rceil \cdot (1/4)^2}{2} \right) \leq \delta. \tag{D.39}
\]
Therefore, with probability at least $1 - \delta$, we have $|\tilde{P}_\alpha(p) - P_\alpha(p)| \leq \epsilon P_\alpha(p)$.

**E Theorem 4.2: Non-integer Rényi entropy estimation, $0 < \alpha < 1$**

In this Section, we prove Theorem 4.2, which is rewritten below:

**Theorem E.1 (Rewriting Theorem 4.2).** The output of Algorithm 8 approximates $P_\alpha(p)$ within a multiplicative error $0 < \epsilon \leq O(1)$ with success probability at least $1 - \delta$ for some $\delta > 0$ using $\tilde{O}\left(\frac{n^{\frac{1}{\alpha-1}}}{\epsilon} \right)$ quantum queries to $p$, where $\tilde{O}$ hides polynomials terms of $\log n$, $\log 1/\epsilon$, and $\log 1/\delta$.

Before we give the formal proof of Theorem 4.2, we compare the similarities and differences between Algorithm 3 and Algorithm 8, listed below:

- In both algorithms, the subroutine $A$ has the same structure, and is designed to estimate $P_\alpha(p)$. However, to make the expectation of $A$ close to $P_\alpha(p)$, the *EstAmp* in Algorithm 3 suffices to take $M = 2^{\lceil \log_2 (\frac{\sqrt{n} \pi}{\log (\sqrt{n})}) \rceil} + 1$ queries (see Lemma D.1), whereas the *EstAmp* in Algorithm 8 needs to take $M = 2^{\lceil \log_2 (\frac{1/2\alpha}{\epsilon} \log (\frac{1/2\alpha}{\epsilon})) \rceil} + 1$ queries (see Lemma E.1);

- In both algorithms, we use Theorem 2.2 to approximate the expectation of $A$ (denoted $\tilde{E}$), hence they both need to upper-bound the variance of $A$ by a multiple of $\tilde{E}^2$. However, technically the proofs are different, and we obtain different upper bounds in Lemma D.2 and Lemma E.2, respectively;
Algorithm 8: Estimate the $\alpha$-power sum $P_\alpha(p)$ of $p = (p_i)_{i=1}^n$ on $[n], 0 < \alpha < 1$

Regard the following subroutine as $\mathcal{A}$:

<table>
<thead>
<tr>
<th>Draw a sample $i \in [n]$ according to $p$;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use the amplitude estimation procedure $\text{EstAmp}$ with $M = 2^{\lceil \log_2 \left(\frac{\alpha^{1/2}\epsilon}{\alpha \log(2^{1/2\alpha})}\right)\rceil + 1}$</td>
</tr>
<tr>
<td>queries to obtain an estimate $\tilde{p}_i$ of $p_i$;</td>
</tr>
<tr>
<td>Output $\tilde{x}_i = \tilde{p}_i^{-\alpha-1}$;</td>
</tr>
</tbody>
</table>

1. Input parameters $(\alpha, \epsilon, \delta)$, where $\epsilon$ is the multiplicative error and $\delta$ is the failure probability;
2. if $\alpha > 1 - \frac{1}{\log n}$ then
   3. Take $a = 1$ and $b = \epsilon$ as lower and upper bounds on $P_\alpha(p)$, respectively;
4. else
   5. Recursively call Algorithm 8 with $\alpha' = \alpha(1 - \frac{1}{\log n})^{-1}$, $\epsilon = 1/2$, and $\delta = \frac{1}{12 \log n \log 1/\alpha}$ therein to give an estimate $\tilde{P}_\alpha(p)$ of $P_\alpha(p)$.
   For simplicity, denote $P := \tilde{P}_\alpha(p)$. Take $a = (P/2)^{\epsilon^{-1} \log n}$ and $b = e(2P)^{\epsilon^{-1} \log n}$ as lower and upper bounds on $P_\alpha(p)$, respectively;
6. Set $l = \Theta \left( \frac{\log^{3/2}}{\epsilon} \log \log \left( \frac{\log^{\frac{1}{\alpha}}}{\epsilon} \right) \right)$;
7. Use $\mathcal{A}$ for $l$ executions in Theorem 2.2 using $a$ and $b$ as auxiliary information and output an estimation of $P_\alpha(p)$;
8. Run Line 1 to Line 7 for $\left[ 48 \log \frac{1}{\delta} \right]$ executions and take the median of all outputs in Line 7, denoted as $\tilde{P}_\alpha(p)$. Output $\tilde{P}_\alpha(p)$;

- Since both algorithms use Theorem 2.2, they both need to compute a lower bound $a$ and upper bound $b$ on $P_\alpha(p)$. Both algorithms achieve this by observing Lemma D.3, and they both compute $a$ and $b$ by recursively call the estimation of $P_{\alpha'}(p)$ for some $\alpha'$ closer to 1. However, in the case $\alpha > 1$, Algorithm 3 makes $\alpha'$ smaller and smaller by multiplying $(1 + \frac{1}{\log n})^{-1}$ each time, and ends the recursion when $\alpha' < 1 + \frac{1}{\log n}$; in the case $0 < \alpha < 1$, Algorithm 8 makes $\alpha'$ larger and larger by multiplying $(1 - \frac{1}{\log n})^{-1}$ each time, and ends the recursion when $\alpha' > 1 - \frac{1}{\log n}$. This leads to different expressions for $a$ and $b$ in Line 3 and Line 5 of both algorithms, and technically the proofs for Lemma D.4 and Lemma E.3 is different;

- Both algorithms boost the success probability to $1 - \delta$ by repeating the algorithm for $\left[ 48 \log \frac{1}{\delta} \right]$ executions and taking the median, and the correctness is both promised by Lemma D.5.

**Proof of Theorem 4.2.** First, if the estimate $\tilde{p}_i$ in the subroutine $\mathcal{A}$ of Algorithm 8 were precisely accurate, the expectation of the subroutine’s output would be $E := \sum_{i \in [n]} p_i \cdot p_i^{-\alpha-1} = P_\alpha(p)$. To be precise, we bound how far the actual expectation of the subroutine’s output $\tilde{E}$ is from the exact value $P_\alpha(p)$. In Lemma E.1, we show that when taking $M = 2^{\lceil \log_2 \left(\frac{\alpha^{1/2}\epsilon}{\alpha \log(2^{1/2\alpha})}\right)\rceil + 1}$ queries in $\text{EstAmp}$, we have $|\tilde{E} - E| = O(\epsilon E)$.

As a result, to approximate $P_\alpha(p)$ within multiplicative error $\Theta(\epsilon)$, it is equivalent to approximate $\tilde{E}$ within multiplicative error $\Theta(\epsilon)$. Recall Theorem 2.2 (multiplicative quantum Chebyshev inequality) showed that if the variance of the random variable outputted by $\mathcal{A}$ is at most $\sigma^2 \tilde{E}^2$ for a known $\sigma$, and if we can obtain two values $a, b$ such that $\tilde{E} \in [a, b]$ with probability at least 0.9, then $O((\sigma b/\epsilon a)$ executions of $\mathcal{A}$ suffices to approximate $\tilde{E}$ within multiplicative error $\epsilon$ with success probability at least 0.8. In the main body of the algorithm (Line 1 to Line 8), we use Theorem 2.2 to approximate $\tilde{E}$.
On the one hand, in Lemma E.2, we show that for any $0 < \alpha < 1$, the variance is at most $2n^{1/\alpha-1}\tilde{E}^2$ with probability at least $\frac{\alpha}{\pi}$. This gives $\sigma = \sqrt{2n^{1/\alpha-1}} = O(n^{1/2\alpha-1/2})$.

On the other hand, we need to compute the lower bound $a$ and upper bound $b$. As stated in the proof of Theorem 4.1, for any $0 < \alpha_1 < \alpha_2$ with $\frac{\alpha_2}{\alpha_1} = 1 + O\left(\frac{1}{\log n}\right)$,\[
\sum_{i \in [n]} p_i^{\alpha_1} = \Theta\left(\left(\sum_{i \in [n]} p_i^{\alpha_2}\right)^{\frac{\alpha_1}{\alpha_2}}\right). \tag{E.1}
\]

As a result, we compute $a$ and $b$ by recursively calling Algorithm 8 to estimate $P_{\alpha'}(p)$ for $\alpha' = \frac{\alpha}{(1 - 1/\log n)}$, which is used to compute the lower bound $a$ and upper bound $b$ in Line 5; the recursive call keeps until $\alpha > 1 - \frac{1}{\log n}$, when $a = 1$ and $b = e$ (as in Line 3) are simply lower and upper bounds on $P_{\alpha}(p)$.

To be precise, in Lemma E.3, we prove that $b/a \leq 4e = O(1)$, and with probability at least $1/e^{1/12} > 0.92$, $a$ and $b$ are indeed lower and upper bounds on $P_{\alpha}(p)$, respectively; furthermore, in Line 5, Algorithm 8 is recursively called by at most $\log n \log \frac{1}{\alpha}$ times, and each recursive call takes at most $\tilde{O}(n^{1/2-1/2})$ queries. This promises that when we apply Corollary 2.2, the cost $C_{a,b}$ is dominated by the query cost from Algorithm 6.

Combining all points above, Corollary 2.2 approximates $\tilde{E}$ up to multiplicative error $\Theta(\epsilon)$ with success probability at least $\frac{8}{\pi^2} \cdot 0.92 \cdot 4/5 > 2/3$ using\[
\log n \log \frac{1}{\alpha} \cdot \tilde{O}\left(\frac{4e \cdot \sqrt{2n^{1/\alpha-1}}}{\epsilon}\right) \cdot 2^{\log_2\left(\frac{2^{1/2\alpha}}{\epsilon} \log\left(2^{1/2\alpha}\right)\right) + 1} = \tilde{O}\left(n^{\frac{1}{2}-\frac{1}{2}}\right) \tag{E.2}
\]
quantum queries. Together with $|	ilde{E} - E| = O(\epsilon E)$ and rescale $t, M$ by a large enough constant, Line 1 to Line 7 in Algorithm 8 approximates $E = P_{\alpha}(p)$ up to multiplicative error $\epsilon$ with success probability at least $2/3$.

Finally, following from Lemma D.5, after repeating the procedure for $48 \log \frac{1}{\delta}$ executions and taking the median $\tilde{P}_{\alpha}(p)$ (as in Line 8), the success probability that $\tilde{P}_{\alpha}(p)$ approximates $P_{\alpha}(p)$ within multiplicative error $\epsilon$ is boosted to $1 - \delta$.

In the coming Subsections, we give proofs for the lemmas mentioned above.

### E.1 Expectation of $A$ is $\epsilon$-close to $P_{\alpha}(p)$

**Lemma E.1.** $|\tilde{E} - E| = O(\epsilon E)$.

**Proof of Lemma E.1.** For convenience, denote $m = \lceil \log_2\left(\frac{2^{1/2\alpha}}{\epsilon} \log\left(2^{1/2\alpha}\right)\right)\rceil + 1$, and $S_0, S_1, \ldots, S_m$ the same as previous definitions. We still have (3.1). By linearity of expectation,\[
|\tilde{E} - E| \leq \sum_{i \in [n]} p_i E\left[\frac{1}{p_i^{1-\alpha}} - \frac{1}{p_i^{-\alpha}}\right] = \sum_{j=0}^{m} \sum_{i \in S_j} p_i E\left[\frac{1}{p_i^{1-\alpha}} - \frac{1}{p_i^{-\alpha}}\right]. \tag{E.3}
\]

Therefore, to prove $|\tilde{E} - E| = O(\epsilon E)$ it suffices to show\[
\sum_{j=0}^{m} \sum_{i \in S_j} p_i E\left[\frac{\alpha - 1}{p_i^{\alpha-1}} - p_i^{\alpha-1}\right] = O\left(\epsilon \sum_{i \in [n]} p_i^\alpha\right). \tag{E.4}
\]
For each $i \in [n]$ we write $p_i = \sin^2(\theta_i \pi)$. Assume $k \in \mathbb{Z}$ such that $k \leq 2^m \theta < k + 1$. Same as Appendix D, we have (D.6). Furthermore, similar to (D.7) we have

\[
(\sin((\theta_i + \frac{l}{2^m})\pi))^2(\alpha-1) - (\sin(\theta_i \pi))^{2(\alpha-1)} = O(\frac{l}{2^m} (\theta_i \pi)^{2\alpha-3}).
\] (E.5)

Combining (D.6) and (E.5), we have

\[
\sum_{j=0}^{m} \sum_{i \in S_j} p_i \mathbb{E}\left[\frac{1}{p_i^{1-\alpha}} - \frac{1}{p_i^{1-\alpha}}\right] = O\left(\frac{\pi^{2\alpha-1} m}{2^{2am}} \cdot \sum_{j=0}^{m} s_j 2^{(2\alpha-1)j}\right). \tag{E.6}
\]

On the other side,

\[
\epsilon \sum_{i \in [n]} p_i^{\alpha} = \Theta\left(\epsilon \sum_{j=0}^{m} s_j \cdot \left(\frac{2^j}{2m \pi}\right)^{2\alpha}\right) = \Theta\left(\epsilon \cdot \frac{\pi^{2\alpha}}{2^{2am}} \sum_{j=0}^{m} s_j 2^{2\alpha j}\right). \tag{E.7}
\]

Therefore, to prove equation (E.4), by (E.6) and (E.7) it suffices to prove

\[
\sum_{j=0}^{m} s_j 2^{(2\alpha-1)j} = O\left(\frac{\epsilon}{m} \sum_{j=0}^{m} s_j 2^{2\alpha j}\right). \tag{E.8}
\]

Because $m = \lceil \log_2(\frac{n^{1/2\alpha}}{\epsilon} \log(\frac{n^{1/2\alpha}}{\epsilon})) \rceil + 1$, we have $\frac{m}{m} \geq \frac{n^{1/2\alpha}}{\epsilon}$, hence $\frac{m}{n^{1/2\alpha}} \geq \frac{m}{\epsilon}$. As a result, it suffices to prove

\[
\sum_{j=0}^{m} s_j 2^{(2\alpha-1)j} = O\left(\frac{n^{1/2\alpha}}{2m} \sum_{j=0}^{m} s_j 2^{2\alpha j}\right). \tag{E.9}
\]

Since $s_j \in \mathbb{N}$, $s_j \leq s_j^{1/\alpha}$; as a result,

\[
\frac{\sum_{j=0}^{m} s_j 2^{2j}}{(\sum_{j=0}^{m} s_j 2^{2\alpha j})^{1/\alpha}} \leq \sum_{j=0}^{m} (s_j 2^{2\alpha j})^{1/\alpha} \leq \sum_{j=0}^{m} s_j 2^{2\alpha j} = 1. \tag{E.10}
\]

Plugging (3.1) into the inequality above, we have

\[
\left(\sum_{j=0}^{m} s_j 2^{2\alpha j}\right)^{\frac{1}{2\alpha}} = \Omega(2^m). \tag{E.11}
\]

On the other side, by Hölder’s inequality we have

\[
\left(\sum_{j=0}^{m} s_j\right)^{\frac{1}{2\alpha}} \left(\sum_{j=0}^{m} s_j 2^{2\alpha j}\right)^{\frac{2\alpha-1}{2\alpha}} \geq \sum_{j=0}^{m} s_j 2^{(2\alpha-1)j}. \tag{E.12}
\]

Combining (3.1), (E.11), and (E.12), we get exactly (E.9). □
E.2 Bound the variance of $\mathcal{A}$ by the square of its expectation

**Lemma E.2.** With probability at least $\frac{8}{\pi^2}$, the variance of the random variable outputted by $\mathcal{A}$ is at most $2n^{1/\alpha - 1}\tilde{E}^2$.

**Proof of Lemma E.2.** Because $\tilde{E} = \sum_{i=1}^{n} p_i \tilde{p}_i^{\alpha - 1}$ and the variance is $\sum_{i=1}^{n} p_i \cdot (\tilde{p}_i^{\alpha - 1})^2 - \left(\sum_{i=1}^{n} p_i \cdot \tilde{p}_i^{\alpha - 1}\right)^2 \leq \sum_{i=1}^{n} p_i \cdot (\tilde{p}_i^{\alpha - 1})^2$, it suffices to show that

$$\sum_{i=1}^{n} p_i \cdot (\tilde{p}_i^{\alpha - 1})^2 \leq 2n^{1/\alpha - 1} \left(\sum_{i=1}^{n} p_i \cdot \tilde{p}_i^{\alpha - 1}\right)^2. \tag{E.13}$$

By Theorem 2.3, with probability at least $\frac{8}{\pi^2}$, we have that for any $i \in [n]$,

$$|\tilde{p}_i - p_i| \leq \frac{2\pi \sqrt{p_i}}{2m} \leq \frac{\epsilon \pi \sqrt{p_i}}{n^{1/2\alpha}}. \tag{E.14}$$

As a result,

$$\sum_{i=1}^{n} p_i (\tilde{p}_i^{\alpha - 1})^2 \leq \sum_{i=1}^{n} p_i \left(p_i - \frac{\epsilon \pi \sqrt{p_i}}{n^{1/2\alpha}}\right)^{-2(1-\alpha)} \tag{E.15}$$

$$= \sum_{i=1}^{n} p_i^{2\alpha - 1} \left(1 - \frac{\epsilon \pi}{n^{1/2\alpha} \sqrt{p_i}}\right)^{-2(1-\alpha)} \tag{E.16}$$

$$\approx \sum_{i=1}^{n} p_i^{2\alpha - 1} \left(1 + 2(1-\alpha)\frac{\epsilon \pi}{n^{1/2\alpha} \sqrt{p_i}}\right) \tag{E.17}$$

$$= \sum_{i=1}^{n} p_i^{2\alpha - 1} + \frac{2(1-\alpha)\epsilon \pi}{n^{1/2\alpha}} \sum_{i=1}^{n} p_i^{2\alpha - 0.5}. \tag{E.18}$$

Furthermore,

$$\sqrt{n} \left(\sum_{i=1}^{n} p_i^{2\alpha - 1}\right) \geq \left(\sum_{j=1}^{n} \sqrt{p_j}\right) \left(\sum_{i=1}^{n} p_i^{2\alpha - 1}\right) \geq \sum_{i=j=1}^{n} \sqrt{p_j p_i^{2\alpha - 1}} = \sum_{i=1}^{n} p_i^{2\alpha - 0.5}. \tag{E.19}$$

Plugging this into (E.18), we have

$$\sum_{i=1}^{n} p_i (\tilde{p}_i^{\alpha - 1})^2 \leq \left(1 + \frac{2(1-\alpha)\epsilon \pi}{n^{1/2\alpha - 1/2}}\right) \sum_{i=1}^{n} p_i^{2\alpha - 1}. \tag{E.20}$$

Using similar techniques, we can show

$$\left(\sum_{i=1}^{n} p_i \cdot \tilde{p}_i^{\alpha - 1}\right)^2 \geq \left(1 - \frac{2(1-\alpha)\epsilon \pi}{n^{1/\alpha - 1}}\right) \left(\sum_{i=1}^{n} p_i^{\alpha}\right)^2. \tag{E.21}$$

Since $0 < \alpha < 1$,

$$\lim_{n \to \infty} 1 + \frac{2(1-\alpha)\epsilon \pi}{n^{1/2\alpha - 1/2}} = 1, \quad \lim_{n \to \infty} 1 - \frac{2(1-\alpha)\epsilon \pi}{n^{1/\alpha - 1}} = 1. \tag{E.22}$$
Because (E.17) only omits lower order terms and the limits in (E.22) are both 1, to prove (E.13) it suffices to prove that for large enough \( n \),
\[
\sum_{i=1}^{n} p_i^{2\alpha-1} \leq n^{1/\alpha-1} \left( \sum_{i=1}^{n} p_i^{\alpha} \right)^{2}.
\]  
(E.23)

By generalized mean inequality, we have
\[
\left( \frac{1}{n} \sum_{i=1}^{n} p_i^{2\alpha-1} \right)^{\frac{1}{2\alpha-1}} \leq \left( \frac{1}{n} \sum_{i=1}^{n} p_i^{\alpha} \right)^{\frac{1}{\alpha}}.
\]  
(E.24)

Therefore,
\[
\sum_{i=1}^{n} p_i^{2\alpha-1} \leq n^{1-\frac{2\alpha-1}{\alpha}} \left( \sum_{i=1}^{n} p_i^{\alpha} \right)^{\frac{2\alpha-1}{\alpha}} \leq n^{1/\alpha-1} \left( \sum_{i=1}^{n} p_i^{\alpha} \right)^{2}. 
\]  
(E.25)

Hence the result follows.

\[ \square \]

E.3 Analyze the recursive calls

**Lemma E.3.** With probability at least 0.92, the \( a \) and \( b \) in Line 3 or Line 5 of Algorithm 8 are indeed lower and upper bounds on \( P_\alpha(p) \), respectively, and \( b/a = O(1) \); furthermore, in Line 5, Algorithm 8 is recursively called for at most \( \log n \log \frac{1}{\alpha} \) executions, and each recursive call takes at most \( \tilde{O}(n^{\frac{1}{\alpha}-\frac{1}{2}}) \) queries.

**Proof of Lemma E.3.** Similar to Lemma D.4, we decompose the proof into two parts:

- **In Line 5, Algorithm 8 is recursively called for at most \( \log n \log \frac{1}{\alpha} \) executions, and each recursive call takes at most \( \tilde{O}(n^{\frac{1}{\alpha}-\frac{1}{2}}) \) queries:**

  Because each recursive call of Algorithm 8 increases \( \alpha \) by multiplying \( (1 - \frac{1}{\log n})^{-1} \) and the recursion ends when \( \alpha > 1 - \frac{1}{\log n} \), the total number of recursive calls is at most
  \[
  \frac{\log \alpha}{\log(1 - \frac{1}{\log n})} \leq \log n \log \frac{1}{\alpha}.
  \]

  When \( \alpha > 1 - \frac{1}{\log n} \), \( a \) and \( b \) are set in Line 3 and no extra queries are needed; when Line 5 calls \( \alpha(1 - \frac{1}{\log n})^{-k} \)-power sum estimation for some \( k \in \mathbb{N} \), by induction on \( k \), we see that this call takes at most \( \tilde{O}(n^{(1-1/\log n)k}) \) \( \leq \tilde{O}(n^{\frac{1}{\alpha}-\frac{1}{2}}) \) queries. As a result, when we apply Corollary 2.2, the cost \( C_{a,b} \) is dominated by the query cost from Algorithm 6.

- **With probability at least 0.92, \( a \) and \( b \) are lower and upper bounds on \( P_\alpha(p) \) respectively, and \( b/a = O(1) \):**

  When \( 1 - \frac{1}{\log n} < \alpha < 1 \), on the one hand we have \( \sum_{i=1}^{n} p_i^{\alpha} \geq \sum_{i=1}^{n} p_i = 1 \); on the other hand, because \( n^{\frac{1}{\log n}} = e \), by Lemma D.3 we have
  \[
  \sum_{i=1}^{n} p_i^{\alpha} \leq n^{1-\alpha} \left( \sum_{i=1}^{n} p_i \right)^{\alpha} \leq e.
  \]  
(E.26)
Therefore, \( a = 1 \) and \( b = e \) in Line 3 are lower and upper bounds on \( P_\alpha(p) \) respectively, and \( b/a = e = O(1) \).

When \( \alpha < 1 - \frac{1}{\log n} \), for convenience denote \( \alpha' = \alpha(1 - \frac{1}{\log n})^{-1} \). As justified above, the total number of recursive calls in Line 5 is at most \( \log n \log \frac{1}{\alpha} \). Because we take \( \delta = \frac{1}{12 \log n \log 1/\alpha} \) in Line 5, with probability at least \( 1 - \frac{1}{12 \log n} \), the output of every recursive call is within \( 1/2 \)-multiplicative error. As a result, the \( P \) in Line 5 satisfies \( P/2 \leq \sum_{i=1}^{n} p_i^{\alpha'} \leq 2P \). Combining this with Lemma D.3 and using \( n^{1/\log n} = e \), we have

\[
(P/2)^{1 - \frac{1}{\log n}} \leq \sum_{i=1}^{n} p_i^\alpha \leq e(2P)^{1 - \frac{1}{\log n}}.
\]  

(E.28)

In other words, \( a \) and \( b \) are indeed lower and upper bounds on \( P_\alpha(p) \), respectively. Furthermore, \( b/a = O(1) \) because

\[
\frac{b}{a} = e \cdot 4^{1 - \frac{1}{\log n}} \leq 4e = O(1).
\]

(E.29)

\[\square\]

### F  Theorem 6.1: Min-entropy estimation

We first establish the following lemma:

**Lemma F.1.** Let \( X \sim \text{Poi}(\mu) \). Then, if \( \mu < \frac{1}{\sqrt{1+\epsilon}} \cdot \frac{16 \log n}{\epsilon^2} \), we have

\[
\Pr \left[ X \geq \frac{16 \log n}{\epsilon^2} \right] \leq \frac{1}{n^2};
\]  

(F.1)

If \( \mu \geq \frac{16 \log n}{\epsilon^2} \), then

\[
\Pr \left[ X \geq \frac{16 \log n}{\epsilon^2} \right] \geq \frac{1}{2e}.
\]  

(F.2)

**Proof of Lemma F.1.** We first prove (F.1). In [15], it is shown that if \( \lambda > 0 \) and \( X \sim \text{Poi}(\lambda) \), then for any \( \nu > 1 \) we have

\[
\Pr[X \geq \nu \lambda] \leq \frac{e^{-\lambda} \lambda^{\nu \lambda}}{\left( \nu \lambda \right)!(1-1/\nu)}.
\]  

(F.3)

Taking \( \lambda = \frac{1}{\sqrt{1+\epsilon}} \cdot \frac{16 \log n}{\epsilon^2} \) and \( \nu = \sqrt{1+\epsilon} \), by Sterling’s formula we have

\[
\Pr \left[ X \geq \frac{16 \log n}{\epsilon^2} \right] \leq \frac{e^{-\lambda} \lambda^{\nu \lambda}}{(\nu \lambda)!(1-1/\nu)} \approx \frac{2}{\epsilon} \frac{e^{-\lambda} \lambda^{\nu \lambda}}{2 \pi \nu} \approx \sqrt{\frac{2}{\pi} \frac{1}{e^{\epsilon}}} \left( \frac{\epsilon^{\epsilon/2}}{(1+\epsilon)^{1+\epsilon/2}} \right)^\lambda.
\]  

(F.4)

The tail bound of Poisson distributions is also studied elsewhere, for example, in [23, Exercise 4.7].
Because

\[
\lim_{\epsilon \to 0} \left( \frac{e^{\epsilon/2}}{(1 + \frac{\epsilon}{2})^{1+\epsilon/2}} \right)^{8/\epsilon^2} = \lim_{\epsilon \to 0} \exp \left[ \frac{4}{\epsilon} - \left( \frac{8}{\epsilon^2} + \frac{4}{\epsilon} \right) \ln \left( 1 + \frac{\epsilon}{2} \right) \right] = \lim_{\epsilon \to 0} \exp \left[ -1 + O(\epsilon) \right] = e^{-1},
\]

we have

\[
\left( \frac{e^{\epsilon/2}}{(1 + \frac{\epsilon}{2})^{1+\epsilon/2}} \right)^{\lambda} \approx e^{-\frac{\epsilon^2}{8}} \approx n^2.
\]

Plugging (F.8) into (F.4), we have

\[
\Pr \left[ X \geq \frac{16 \log n}{\epsilon^2} \right] \leq \sqrt{\frac{2}{\pi}} \frac{1}{\sqrt{16 \log n}} \frac{1}{n^2} \leq \frac{1}{n^2}.
\]

Now we prove (F.2). A theorem of Ramanujan [18, Question 294] states that for any positive integer \( M \),

\[
\frac{1}{2} e^M = \sum_{m=0}^{M-1} \frac{M^m}{m!} + \theta(M) \cdot \frac{M!}{M!},
\]

where \( \frac{1}{3} \leq \theta(M) \leq \frac{1}{2} \forall M \). As a result,

\[
\sum_{m=M}^{\infty} \frac{M^m}{m!} \geq \frac{1}{2} e^M.
\]

We take \( M = \lceil \frac{16 \log n}{\epsilon^2} \rceil \). By (F.11), we have

\[
\Pr \left[ X \geq \frac{16 \log n}{\epsilon^2} \right] = e^{-\frac{16 \log n}{\epsilon^2}} \sum_{m=M}^{\infty} \frac{(\frac{16 \log n}{\epsilon^2})^m}{m!} \geq e^{-M-1} \sum_{m=M}^{\infty} \frac{M^m}{m!} \geq \frac{1}{2e}.
\]

Proof of Theorem 6.1. Because \( H_\infty(p) = -\log \max_{i \in [n]} p_i \), to approximate \( H_\infty(p) \) within additive error \( \epsilon \), it is equivalent to approximate \( \max_{i \in [n]} p_i \) within multiplicative error \( \epsilon \).

Denote \( \sigma \) to be the permutation on \([n]\) such that \( p_{\sigma(1)} \geq p_{\sigma(2)} \geq \cdots \geq p_{\sigma(n)} \). Without loss of generality, we assume that \( p_{\sigma(2)} \leq \frac{p_{\sigma(1)}}{1+\epsilon} \); otherwise, \( p_{\sigma(2)} \) is close enough to \( p_{\sigma(1)} \) in the sense that applying quantum counting to \( p_{\sigma(2)} \) within multiplicative error \( \epsilon \) gives an approximation to \( p_{\sigma(1)} \) within multiplicative error \( 2\epsilon \). We may assume that every call of the \( \lceil \frac{16 \log n}{\epsilon^2} \rceil \)-distinctness quantum algorithm in Line 4 of Algorithm 5 succeeds if and only if an \( \lceil \frac{16 \log n}{\epsilon^2} \rceil \)-collision exists, because this happens with probability at least \( \left( 1 - \frac{\epsilon}{2 \log n} \right)^{\log_2 \sqrt{\epsilon}} \geq e^{-1} = \Omega(1) \); for convenience, this is always assumed in the result of the proof.

On the one hand, when

\[
\frac{p_{\sigma(1)} \cdot 16 \lambda \log n}{\epsilon^2} < \frac{1}{\sqrt{1+\epsilon}} \cdot \frac{16 \log n}{\epsilon^2},
\]

31
by Lemma F.1 we have \( \Pr \left[ M_i \geq \frac{16 \log n}{\epsilon^2} \right] \leq \frac{1}{n^2} \forall i \in [n] \). Therefore, with probability at least \( 1 - \frac{1}{n} \), there is no \( \frac{16 \log n}{\epsilon^2} \)-collision in \( S \). Since the while loop only has at most \( \log_{1+\epsilon} n = O\left( \frac{\log n}{\epsilon} \right) \) rounds and \((1 - \frac{1}{n})^{\log n/\epsilon} = 1 - o(1)\), we may assume that as long as (F.13) holds, Line 4 of Algorithm 5 always has a negative output and Line 5 enforces \( \lambda \leftarrow \lambda \cdot \sqrt{1+\epsilon} \) and jumps to the start of the while loop.

The while loop keeps iterating until (F.13) is violated. In the second iteration after (F.13) is violated, we have

\[
\frac{16 \log n}{\epsilon^2} \leq \frac{p_{\sigma(1)} \cdot 16 \lambda \log n}{\epsilon^2} < \sqrt{1+\epsilon} \cdot \frac{16 \log n}{\epsilon^2},
\]

since \( p_{\sigma(2)} \leq \frac{p_{\sigma(1)}}{1+\epsilon} \), we have

\[
\frac{p_{\sigma(2)} \cdot 16 \lambda \log n}{\epsilon^2} \leq \frac{1}{\sqrt{1+\epsilon}} \frac{16 \log n}{\epsilon^2}.
\]

As a result, by Lemma F.1 we have

\[
\Pr \left[ M_{\sigma(1)} \geq \frac{16 \log n}{\epsilon^2} \right] \geq \frac{1}{2e}; \quad \Pr \left[ M_i \geq \frac{16 \log n}{\epsilon^2} \right] \leq \frac{1}{n^2} \forall i \in [n]/\{\sigma(1)\}.
\]

Therefore, \( \Pr \left[ \text{Line 4 outputs } \sigma(1) \text{ in the second iteration after (F.13) is violated} \right] \geq \frac{1}{2e} \left( 1 - \frac{n-1}{n^2} \right)^{n-1} \). In the first iteration after (F.13) is violated, we still have \( \Pr \left[ M_i \geq \frac{16 \log n}{\epsilon^2} \right] \leq \frac{1}{n^2} \forall i \in [n]/\{\sigma(1)\} \).

Therefore,

\[
\Pr \left[ \text{Line 4 outputs } \sigma(1) \text{ in the first or second iteration after (F.13) is violated} \right] \geq \frac{1}{2e} \left( 1 - \frac{n-1}{n^2} \right)^{n-1} \cdot \left( 1 - \frac{n-1}{n^2} \right)^{n-1} \geq \frac{1}{2e^3} = \Omega(1).
\]

In all, with probability \( \Omega(1) \), Line 4 of Algorithm 5 outputs \( \sigma(1) \) correctly in the first or second iteration after (F.13) is violated; after that, the quantum counting in Line 5 approximates \( p_{\sigma(1)} = \max_{i \in [n]} p_i \) within multiplicative error \( \epsilon \). This establishes the correctness of Algorithm 5.

It remains to show that the quantum query complexity of Algorithm 5 is \( \widetilde{O}\left( Q\left( \left\lceil \frac{16 \log n}{\epsilon^2} \right\rceil \right) \right) \). Because there are at most \( \log_{1+\epsilon} n = O\left( \frac{\log n}{\epsilon} \right) \) iterations in the while loop, the \( \left\lceil \frac{16 \log n}{\epsilon^2} \right\rceil \)-distinctness algorithm in Line 4 is called at most \( O\left( \frac{\log n}{\epsilon^2} \right) \) executions; if it gives an \( \left\lceil \frac{16 \log n}{\epsilon^2} \right\rceil \)-collision, because \( \max_{i \in [n]} p_i \geq 1/n \), the quantum query complexity of the quantum counting in Line 5 is at most \( O\left( \frac{\sqrt{n}}{\epsilon} \right) \), which is smaller than the \( \Omega(n^{2/3}) \) quantum lower bound on distinctness problem [2]. As a result, the query complexity of Algorithm 5 in total is at most

\[
O\left( \frac{\log n}{\epsilon} \right) \cdot Q\left( \left\lceil \frac{16 \log n}{\epsilon^2} \right\rceil \text{-distinctness} \right) + O\left( \frac{\sqrt{n}}{\epsilon} \right) = \widetilde{O}\left( \left\lceil \frac{16 \log n}{\epsilon^2} \right\rceil \text{-distinctness} \right).
\]

\[ \square \]

Remark F.1. In some special cases, Algorithm 5 already demonstrates provable quantum speedup. Recall the state-of-the-art quantum algorithm for \( k \)-distinctness is [8] by Belovs, which has query complexity \( O(2^{k^2} n^{1-2k^{-2}/(2^{k-1})}) \); however, this is superlinear when \( k = \Theta(\log n) \). Nevertheless, if we are promised that \( H_\infty(p) \leq f(n) \) for some \( f(n) = o(\sqrt{\log n}) \), then we can replace the \( n \) in Line 2 of Algorithm 5 by \( e^{f(n)} \) and replace every \( \left\lceil \frac{16 \log n}{\epsilon^2} \right\rceil \) by \( \left\lceil \frac{16f(n)}{\epsilon^2} \right\rceil \), and it can be shown that the quantum query complexity of min-entropy estimation is \( \widetilde{O}\left( e^{\left( \frac{1}{2} + o(1) \right) f(n)} \right) \), whereas the best classical algorithm takes \( \tilde{O}(e^{f(n)}) \) queries. In this case, we obtain a \( \left( \frac{3}{2} + o(1) \right) \)-quantum speedup, but the classical query complexity is already small \( (e^{\sqrt{\log n}} = n^{1/\sqrt{\log n}} = o(n^c) \text{ for any } c > 0) \).
References


