Mean estimation when you have the source code; or, quantum Monte Carlo methods*

Robin Kothari† Ryan O'Donnell‡

Abstract

Suppose $\boldsymbol{y}$ is a real random variable, and one is given access to "the code" that generates it (for example, a randomized or quantum circuit whose output is $\boldsymbol{y}$). We give a quantum procedure that runs the code $O(n)$ times and returns an estimate $\widehat{\boldsymbol{\mu}}$ for $\mu = \mathbf{E}[\boldsymbol{y}]$ that with high probability satisfies $|\widehat{\boldsymbol{\mu}} - \mu| \le \sigma/n$, where $\sigma = \mathbf{stddev}[\boldsymbol{y}]$. This dependence on n is optimal for quantum algorithms. One may compare with classical algorithms, which can only achieve the quadratically worse $|\widehat{\boldsymbol{\mu}} - \mu| \le \sigma/\sqrt{n}$. Our method improves upon previous works, which either made additional assumptions about $\boldsymbol{y}$, and/or assumed the algorithm knew an a priori bound on σ, and/or used additional logarithmic factors beyond $O(n)$. The central subroutine for our result is essentially Grover's algorithm but with complex phases.

1 Introduction

Let $\boldsymbol{y}$ be a real random variable.[1] One may wish to estimate its mean $\mu = \mathbf{E}[\boldsymbol{y}]$ from independent samples $\boldsymbol{y}_1, \boldsymbol{y}_2, \dots, \boldsymbol{y}_n$. A natural strategy is to output the sample mean $\widehat{\boldsymbol{\mu}} = (\boldsymbol{y}_1 + \cdots + \boldsymbol{y}_n)/n$, an unbiased estimator with standard deviation $\sigma/\sqrt{n}$, where $\sigma = \mathbf{stddev}[\boldsymbol{y}] = \sqrt{\mathbf{E}[(\boldsymbol{y} - \mu)^2]}$. Then Chebyshev's inequality implies, say,

$$\mathbf{Pr}[|\widehat{\boldsymbol{\mu}} - \mu| \ge 10\sigma/\sqrt{n}] \le 1\%. \tag{1.1}$$

As familiar special cases: if $\boldsymbol{y}$ is bounded in $[0,1]$, then $\sigma \le 1$ and we get that $n = O(1/\epsilon^2)$ samples suffice to ensure $|\widehat{\boldsymbol{\mu}} - \mu| \le \epsilon$ with high probability; if $\boldsymbol{y} \in \{0,1\}$, then $\sigma = \sqrt{\mu(1-\mu)} \le \sqrt{\mu}$, and we get that $n = O(1/\epsilon)$ samples suffice to distinguish $\mu \ge \epsilon$ from $\mu \le \epsilon/2$ with high probability. Up to constant factors, these guarantees cannot be improved upon if the samples $\boldsymbol{y}_1, \dots, \boldsymbol{y}_n$ are coming "from nature".

But what if we have "the code" for $\boldsymbol{y}$? By this we mean we have access to, say, a randomized circuit C whose output is $\boldsymbol{y}$. In a certain sense this means we don't need any samples at all to estimate μ: By enumerating all possible random paths for C, we can compute μ perfectly. But this could be enormously expensive; if running C to produce a single sample takes effort S, then the brute-force enumeration analysis might take $\exp(S)$ effort. It is much more practical to treat C as a "black box" and apply Inequality (1.1), expending just $O(S/\epsilon^2)$ effort to get a high-confidence estimate of μ with "error bar" $\epsilon\sigma$. This idea is the essence of the *Monte Carlo Method* [HH64]. Very surprisingly (at least, circa the mid-'90s), one can do quadratically better using a quantum computer! As we show in this work, only $O(S/\epsilon)$ effort is needed to get the same guarantee.

To state our main result, let $\boldsymbol{y}$ be a discrete real random variable (whose values are encodable by bits on a digital computer). We will formally discuss "having the code" for $\boldsymbol{y}$ in Section 2, but for now suffice it to say it includes the following scenarios:

Scenarios for "having the code":

1. Access to a classical randomized circuit (with no input) whose output is a draw from $\boldsymbol{y}$.
2. More generally, access to a unitary quantum circuit (with some fixed input $|0^k\rangle$) such that, upon measuring its output and discarding some bits, we get a draw from $\boldsymbol{y}$. (Note that a quantum circuit with intermediate measurements can be transformed to this form.)

*The full version of the paper can be accessed at https://arxiv.org/abs/2208.07544

†Microsoft Quantum. robin@robinkothari.com

‡Carnegie Mellon University Computer Science Department. Part of this research was performed while the author was at Microsoft Quantum. This work was partially supported by ARO grant W911NF2110001. odonnell@cs.cmu.edu

[1]Throughout we use **boldface** to denote random quantities.

Model	Assumption on y	Additive error	Samples	Reference
Uniform	Bernoulli, $\mu = 0$ or $\mu = 1/n^2$	(distinguishes*)	$O(n)$	[Gro96]
Uniform	$[0,1]$-bounded	$1/n$	$O(n \cdot \mathrm{polylog}\, n)$	[Gro98]
General	Bernoulli	σ/n †	$O(n)$	[BHT98]
General‡	$[0,1]$-bounded	$\sqrt{\mu}/n$	$O(n)$	[Ter99]
Uniform	$\max\{\sigma, \lvert\mu\rvert\} \le \sigma_{\mathrm{bound}}$ known	$\sigma_{\mathrm{bound}}/n$	$O(n \log^{3/2} n \log\log n)$	[Hei02]
General	$\sigma \le \sigma_{\mathrm{bound}}$ known	$\sigma_{\mathrm{bound}}/n$	$O(n \log^{3/2} n \log\log n)$	[Mon15]
General	(none)	σ/n	$O(n \log^{3/2} n \log\log n)$	[Ham21]
General	(none)	$\boldsymbol{\sigma/n}$	$\boldsymbol{O(n)}$	**Us**

Table 1: Prior quantum algorithms for mean estimation.

* Usually equivalently stated as using $O(\sqrt{N})$ queries to distinguish $\mu = 0$ from $\mu = 1/N$.

† Rather than just $\sigma/n = \sqrt{\mu(1-\mu)}/n$, this result is usually stated with an extra additive $1/n^2$ error (cf. [BHMT02]). But note that $1/n^2 \le O(\sigma/n)$ unless $\sigma \ll 1/n$. Supposing $\sigma \ll 1/n$, we have $\mu \ll 1/n^2$ (or $1 - \mu \ll 1/n^2$, but the reasoning will be similar), and inspecting the algorithm shows that it will output the estimate 0 (with high probability). But an estimate of 0 *is* within additive error σ/n of $\mu \sim \sigma^2$ when $\mu \ll 1/n^2$.

‡ Terhal stated her result for the Uniform model, but it is easy to see it also works in the General model, as it is a direct reduction to the General Bernoulli result of [BHT98]. The additive $1/n^2$ appearing in her statement can be deleted for this reason, too.

3. Less generally, access to a unitary quantum circuit that produces $\frac{1}{\sqrt{N}} \sum_{j=1}^{N} |j\rangle |y_j\rangle$, and $\boldsymbol{y}$ is defined to be the uniform distribution on the multiset of reals $\{y_1, y_2, \ldots, y_N\}$. Grover's algorithm works in this model.

In this work, we show the following theorem:

Theorem 1.1. *There is a computationally efficient quantum algorithm with the following properties: Given "the code" for a random variable $\boldsymbol{y}$, the algorithm uses $O(n)$ samples[2] and outputs an estimate $\widehat{\boldsymbol{\mu}}$ such that*

$$\mathbf{Pr}[|\widehat{\boldsymbol{\mu}} - \mu| > \sigma/n] \le 1/3, \tag{1.2}$$

where $\mu = \mathbf{E}[\boldsymbol{y}]$ and $\sigma = \mathbf{stddev}[\boldsymbol{y}]$. (By repeating the algorithm $O(\log 1/\delta)$ times and taking the median, one can reduce the "1/3" to any $\delta > 0$.)

Remark 1.2. Regarding computational efficiency, in Appendix A we show that if the code for $\boldsymbol{y}$ is a circuit of gate complexity S, then the gate complexity of our algorithm in Theorem 1.1 is $O(nS)$.[3]

Theorem 1.1 is known to be optimal (up to constant factors, see e.g. [Ham21, Thm. 4.6.2]), and it caps a long sequence of works that obtain similar results but with more assumptions and/or weaker parameters; see Table 1. (The "Uniform" model in the table's first column refers to the model in Item 3 above.)

1.1 Methods

The centerpiece of Theorem 1.1 is Theorem 1.3 below:

Theorem 1.3. *There is a computationally efficient quantum algorithm that solves the following task:*

Main Task. *Given a parameter $\epsilon > 0$ and "the code" for a random variable $\boldsymbol{y}$ promised to satisfy $\mathbf{E}[\boldsymbol{y}^2] \le 1$, use $O(1/\epsilon)$ samples and distinguish (with confidence at least 2/3) between the cases (i) $|\mu| \le \epsilon/2$ and (ii) $\epsilon \le |\mu| \le 2\epsilon$, where $\mu = \mathbf{E}[\boldsymbol{y}]$.*

One thing to notice is that Theorem 1.3 directly implies Grover's algorithm [Gro96] (in its distinguishing form):[4] Given N equally likely items with either zero or one of them being "marked", we can form the random

[2] In this introduction, we will say that an algorithm uses q "samples" from $\boldsymbol{y}$ to mean that it uses the code for $\boldsymbol{y}$ at most q times.

[3] Except in the rather specific and unlikely case of $\Omega(\log n) \le S < o(\log(n) \cdot (\log\log n)^2)$, in which case there is an extra factor of at most $(\log\log n)^2$.

[4] It's also not hard to show our routine can be used to *find* a unique marked item.

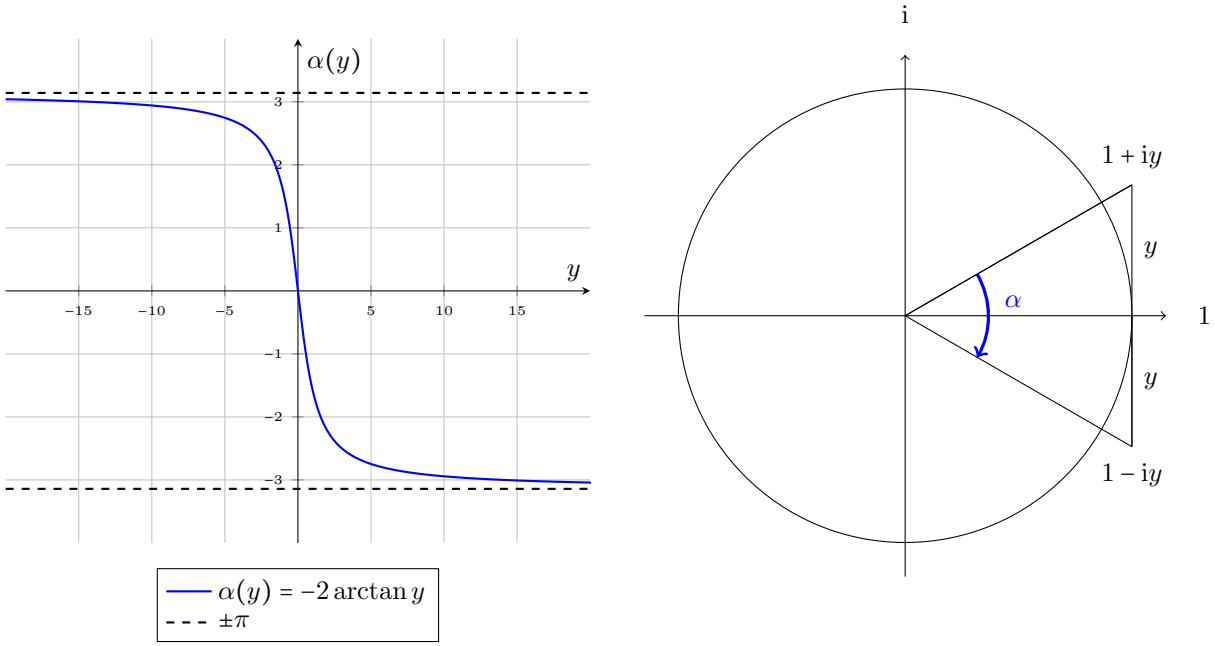

Figure 1: Representations of the function $\alpha(y) = -2\arctan y$.

variable $\boldsymbol{y}$ that is 0 on unmarked items and $\sqrt{N}$ on a marked item. Then we always have $\mathbf{E}[\boldsymbol{y}^2] \leq 1$, and either $|\mu| = 0$ or $|\mu| = 1/\sqrt{N}$ depending on whether there is a marked item. Thus we can use $\epsilon = 1/\sqrt{N}$ in Theorem 1.3.

In fact, our algorithm for Theorem 1.3 essentially *is* Grover's algorithm — but with complex phases! Recall that Grover's algorithm is composed of a product of two unitaries. The first unitary in Grover's algorithm, the "diffusion operator", is a reflection about the uniform superposition over all N items. One can think of this state as encoding the uniform distribution over N equally likely items (but not which are marked/unmarked); if $N = 2^k$, it is the output of $H^{\otimes n}|0^k\rangle$. In our algorithm, the diffusion operator will be similar; in the Item 2 scenario from above where the code is a quantum circuit $\mathcal{C}$, it would be reflection through the code's output $\mathcal{C}|0^k\rangle$.

The second unitary in Grover's algorithm, the "phase oracle", is a diagonal unitary that encodes which items are marked: since in the Grover setup the random variable $\boldsymbol{y}$ takes only two distinct values (marked or unmarked), these are mapped to the two phases -1 and $+1$. In our algorithm, the phase oracle will remain a diagonal unitary, inserting a phase based on the outcome y of the random variable. But since y may now be *any* value from the real line, we need a map from an arbitrary real number to a phase. We map these values to general complex phases by associating $y \in \mathbb{R}$ to $e^{i\alpha} \in \mathbb{C}$, where $\alpha = -2\arctan y$, which maps $\mathbb{R}$ to $(-\pi, +\pi)$ (conceptually, this is the phase that rotates $1 + iy$ to $1 - iy$). Notice that if $\boldsymbol{y}$ just takes on the two values 0 and $\sqrt{N}$, the associated phases are $+1$ and "almost -1".

Grover's algorithm repeatedly alternates the diffusion and phase operators. Our algorithm for the Main Task does the same, but the resulting intermediate states will contain general complex amplitudes. (Nevertheless, the diffusion operator still acts by reflects a list of *complex* amplitudes through their mean.) This is not the first time using complex phases in Grover's has been suggested [Gro05], but our analysis would seem to be new. The main challenge is to analyze the eigenvectors and eigenvalues of the unitary $\mathcal{U}$ obtained by composing the diffusion operator and the phased implementation of $\boldsymbol{y}$. Unlike in the Grover case, it is the composition of a reflection and a *general* unitary operator, so $\mathcal{U}$ is no longer essentially two-dimensional; it is fundamentally high-dimensional.

Nevertheless, we are able to show that when $\mathbf{E}[\boldsymbol{y}^2] \leq 1$, the natural "starting state" (namely, the output of the code) is mostly supported on eigenvectors of $\mathcal{U}$ with eigenvalue near $e^{i\cdot 2|\mu|}$. Given this analysis, we can solve the Main Task in Theorem 1.3 rather easily. One way to finish is an immediate appeal to Quantum Phase Estimation [Kit95]. Alternatively, it's not too hard to show that more elementary strategies can work (provided one adjusts the constant factors in the Main Task's statement): Our analysis implies that form $T = \Theta(1/\epsilon)$, the starting state is close to an eigenvector of $\mathcal{U}^T$ with eigenvalue either close to $+1$ (when $|\mu| \ll \epsilon$) or close to -1

(when $|\mu| \approx \epsilon$). Then these cases can be distinguished with the simple Hadamard test.

Going from Theorem 1.3 to Theorem 1.1. A significant utility of our Theorem 1.3 is that it applies to *any* random variable with $\mathbf{E}[\boldsymbol{y}^2] \le 1$, not just $\{0,1\}$-valued or even $[0,1]$-valued random variables. As we will show in Section 4, this makes it very easy to compose with simple classical reductions. For example, classical binary search lets us upgrade Theorem 1.3 to *estimate* μ to additive $\pm\epsilon$ with $O(1/\epsilon)$ samples. Then, with a standard classical halving trick we can achieve the Approximate Counting / Amplitude Estimation results of [BHT98, BHMT02], as well as the [BHT98, Ter99] results from Table 1. Indeed, the only extra quantum technique we use to reach our final Theorem 1.3 is the recent Quantile Finding algorithm of Hamoudi [Ham21] (which itself is essentially Grover's algorithm together with classical reductions). Thus all of these algorithms, up to and including Theorem 1.1, can be obtained using nothing more than classical reductions and our elementary new quantum routine for Theorem 1.3.

1.2 Applications of mean estimation Mean estimation is used throughout algorithmic theory, and the quadratic speedup afforded by quantum computers (shown precisely herein) has many applications; see, e.g., [Ham21] for an excellent survey. We mention here some basic examples/applications.

Bernoulli random variables, and Circuit-SAT. Recall that a Bernoulli (i.e., $\{0,1\}$-valued) random variable $\boldsymbol{y}$ which is 1 with probability μ has mean μ and standard deviation $\sigma = \sqrt{\mu(1-\mu)} \le \sqrt{\mu}$. Classically (or "without the code"), $O(n)$ samples lets one estimate μ to within additive error $\sqrt{\mu}/\sqrt{n}$; so $O(1/\epsilon^2)$ samples always suffice for additive error ϵ, but also $O(1/\epsilon)$ samples suffice to distinguish $\mu \le \epsilon/2$ from $\mu \ge \epsilon$. Our Theorem 1.1 (and also the much earlier work on amplitude estimation [BHT98]) implies that "with the code", a quantum algorithm can improve these bounds to $n = O(1/\epsilon)$ and $n = O(1/\sqrt{\epsilon})$, respectively.

The simplest case of this corresponds to Grover's algorithm [Gro96] for the Unique-Circuit-SAT problem. Suppose C is a classical m-input, 1-output Boolean circuit with S gates, promised to be either unsatisfiable, or with a unique satisfying assignment. Replacing its inputs by coin-flip gates, we thereby obtain classical "code" (in the model from Item 1 above) for a Bernoulli random variable $\boldsymbol{y}$ that either has mean $\mu = 0$ or mean $\mu = 2^{-m}$. Grover's special case of Theorem 1.1 then shows that $n = O(1/\sqrt{2^{-m}}) = O(\sqrt{2}^m)$ samples — and hence $O(\sqrt{2}^m \cdot S)$ quantum circuit complexity — suffices to decide Unique-Circuit-SAT.

Distinguishing classical probability distributions. This application illustrates the importance of considering *non*-bounded random variables, and achieving an additive guarantee that involves the standard deviation σ.

Suppose q and r are fixed, known probability distributions on $[D]$, and that an algorithm has access to samples from an unknown probability distribution p on $[D]$, promised to be either q or r. It is well known that the sample complexity needed to distinguish $p = q$ from $p = r$ (with error probability at most 1/3, say) is $\Theta(1/H(q,r)^2)$, where $H(q,r)^2 = \sum_{i=1}^{D} (\sqrt{q_i} - \sqrt{r_i})^2$ denotes the squared *Hellinger distance* between q and r.[5] A recent work of Belovs [Bel19] shows that if one has "the code" for p, there is a quantum algorithm that can distinguish $p = q$ from $p = r$ using just $\Theta(1/H(q,r))$ samples.

Here we show how Belovs's result is simply recovered from our Theorem 1.1. Writing $H = H(q,r)$, one way to distinguish q and r is through mean estimation on the random variable $y : [D] \to \mathbb{R}$ defined by[6]

$$y(i) = \frac{\sqrt{q_i} - \sqrt{r_i}}{\sqrt{q_i} + \sqrt{r_i}} = \frac{(\sqrt{q_i} - \sqrt{r_i})^2}{q_i - r_i}. \tag{1.3}$$

Writing $\mu_q = \mathbf{E}_q[\boldsymbol{y}]$, $\sigma_q^2 = \mathbf{Var}_q[\boldsymbol{y}]$ and μ_r, σ_r analogously, observe that we have

$$\mu_q - \mu_r = \sum_{i=1}^{D} (q_i - r_i) y(i) = \sum_{i=1}^{D} (\sqrt{q_i} - \sqrt{r_i})^2 = H^2 \tag{1.4}$$

and

$$\sigma_q^2 + \sigma_r^2 \le \mathbf{E}_q[\boldsymbol{y}^2] + \mathbf{E}_r[\boldsymbol{y}^2] = \sum_{i=1}^{D} (q_i + r_i) \frac{(\sqrt{q_i} - \sqrt{r_i})^2}{(\sqrt{q_i} + \sqrt{r_i})^2} \le H^2, \tag{1.5}$$

[5] Recall that $\frac{1}{2} H(q,r)^2 = 1 - BC(q,r)$, where $BC(q,r) = \sum_{i=1}^{D} \sqrt{q_i}\sqrt{r_i}$ is the *Bhattacharyya coefficient*, which clearly tensorizes: $BC(q^{\otimes n}, r^{\otimes n}) = BC(q,r)^n$. Then the lower bound follows from the known total variation lower bound $TV(q,r) \ge \frac{1}{2} H(q,r)^2$. For the upper bound, read on.

[6] The reader may verify that defining $0/0 = 0$ takes care of edge cases in what follows.

where the inequality here uses $(q_i+r_i)/(\sqrt{q_i}+\sqrt{r_i})^2 \le 1$. From this we see that $p=q$ vs. $p=r$ can be distinguished by estimating the mean $\mathbf{E}_p[\boldsymbol{y}]$ to additive accuracy $H^2/2$; and moreover, that $\sigma_p \le H$. Classically (or without "the code"), we need to ensure $\sigma_p/\sqrt{n} \le H/\sqrt{n} < H^2/2$, and can only say that $n = O(1/H^2)$ samples suffice. But *with* the code, our quantum algorithm from Theorem 1.1 shows that only $H/n < H^2/2$ is needed; i.e., $n = O(1/H)$ samples suffice, matching Belovs's result. (Indeed, careful inspection of Belovs's work shows that his algorithm can be thought of as performing mean estimation/distinguishing on $\boldsymbol{y}$, additionally relying on the fact that H is a *known* upper bound for σ.)

Instance-dependent algorithms for multiplicative mean estimation. Consider the [BHT98] entry of Table 1; it says that with a fixed budget of $O(n)$ samples, one can estimate the mean μ of a Bernoulli random variable $\boldsymbol{y}$ to additive error $\sigma/n \le \sqrt{\mu}/n$. With this guarantee, one cannot even distinguish mean μ from mean 0 unless $n \ge 1/\sqrt{\mu}$. On the other hand, provided $n \ge 2/\sqrt{\mu}$, say, the estimate is accurate to within a *multiplicative* factor of 2. If this is one's only goal, one might wish for an algorithm that *stops early* — after only $O(1/\sqrt{\mu})$ samples — obtaining a factor-2 approximation of μ *despite not knowing* μ *a priori*. This is the idea of *sequential analysis*, from statistics. Note that once μ is known to a factor of 2, one can use the nonadaptive [BHT98] result to get a refined factor-$(1+\epsilon)$ approximation using $n = O(1/(\epsilon\sqrt{\mu}))$ samples.

This sort of "instance-dependent" guarantee was provided even for $[0,1]$-bounded random variables in the work [BHMT02]. Precisely, their algorithm takes a parameter $\epsilon > 0$ and estimates the mean μ of a $[0,1]$-bounded random variable to a factor of $1+\epsilon$ using $O(1/(\epsilon\sqrt{\mu}))$ samples.

Following the theme of Table 1, one might wish to improve this result to take into account the standard deviation σ. Factor-$(1+\epsilon)$ approximation corresponds to additive error $\epsilon\mu$, and equating this with σ/n suggests that an improved instance-dependent sample complexity of $O(\sigma/(\epsilon\mu))$ might be possible. Indeed, if a constant-factor upper bound on the "coefficient of variation" $|\sigma/\mu|$ happens to be known, this is immediate. However if no prior assumptions are made, Hamoudi [Ham21] observes that a lower bound of Nayak [Nay99] shows that up to constant factors, no bound better than $O(\max\{\sigma/(\epsilon\mu), 1/\sqrt{\epsilon\mu}\})$ is possible, even for Bernoulli random variables. On the other hand, Hamoudi also uses the classical sequential analysis methods of [DKLR00] to show that the preceding bound *can* be obtained, up to polylog factors, for $[0,1]$-bounded random variables. The technique is a direct reduction to his instance-*independent* result from Table 1. We may apply the same reduction using our improved result, thereby obtaining:

Theorem 1.4. *In the setting of Theorem 1.1, if* $\boldsymbol{y}$ *is* $[0,1]$*-bounded, there is an algorithm that, given* $\epsilon > 0$*, has the following behavior except with probability at most* $1/3$*: It obtains* $O(\max\{\sigma/(\epsilon\mu), 1/\sqrt{\epsilon\mu}\})$ *samples (dependent on the unknown* μ*), and outputs an estimate* $\widehat{\boldsymbol{\mu}}$ *such that*

$$(1-\epsilon)\mu \le \widehat{\boldsymbol{\mu}} \le (1+\epsilon)\mu. \tag{1.6}$$

Algorithms for finance. As another application, we briefly describe some real-world use cases of the Monte Carlo method in finance. We give only an overview of the financial terms used; for more information, we refer readers to a survey on quantum algorithms in finance [OML19, BvDJ+20, HGL+22] or a textbook on the mathematics of finance [Lue14]. Using quantum computers to solve the financial problems described below has been studied in some detail in prior work; see, e.g., [RGB18, WE19, SES+20, CKM+21].

In finance, a derivative is a contract that derives its value from some underlying variable, such as the price of a specific stock. A "European call option" is a simple example of a derivative. Let us fix an underlying stock whose price on day i is denoted y_i. A European call option with a strike price of K and a maturity date of T days is a contract that on day T rewards the contract holder with $\max\{0, y_T - K\}$. In words, if the stock price on day T is above K, the contract holder is rewarded with the difference, and otherwise receives nothing. Now if we have a probabilistic model for the daily price movement of the stock, we can infer a probability distribution over the possible values y_T. To determine a fair price for this call option (under our model), we need to compute the expected value of y_T under this probability distribution. In practice the probability distribution is efficiently sampleable, and the computational bottleneck is to compute this expected value, which is clearly a Monte Carlo mean estimation task. For example, a very simple model might be that the stock price increases by a factor of f_0 with probability p and decreases by a factor of f_1 with probability $1-p$. A more commonly used model is to assume the stock price follows geometric Brownian motion, as in the Black–Scholes–Merton model [BS73, Mer73]. Under this model, European call options can actually be priced analytically, and we don't need to use Monte Carlo methods. But more generally, an option may depend on more than one underlying asset, and the payoff

function can be be a complicated function of the entire history of stock prices $y_0, \ldots, y_T$. In such cases analytical solutions may not exist, but the Monte Carlo method works just fine as long as the distribution is efficiently sampleable and the payoff function is efficiently computable.

Note also that there is no generic reason why the derivative price should be in a known bounded interval or have known standard deviation bound. Indeed, in the analysis from [CKM+21] on pricing autocallable and TARF derivatives, significant gate complexity arose due to the errors incurred by artificially truncating prices to bounded intervals. This suggests that our methods, which don't require any such bounds, might be helpful.

Other examples from finance where Monte Carlo methods are used in practice include the computation of Value at Risk (VaR) and Conditional Value at Risk (CVaR); these give more examples where the techniques of this paper can be used to give a quadratic speedup using a quantum computer.

More applications in TCS. There are innumerable additional applications of Monte Carlo mean estimation throughout theoretical computer science — simulated annealing algorithms, approximation of partition functions, MCMC approximate counting algorithms, subgraph count estimation algorithms, data stream estimation algorithms, etc.; see, e.g., [Mon15, Ham21] for some illustrations.

2 Preliminaries

In this section we formally define random variables, and what it means to "have the code" for them. In short, we use the same model as Montanaro [Mon15] — essentially, Item 2 in the scenarios from Section 1. (The reader may also refer to the thesis of Hamoudi [Ham21], where the model is discussed in careful detail.)

2.1 Probability distributions

Before defining random variables, we discuss probability distributions. As our random variables will be implemented by finite circuits, it suffices to discuss finite probability distributions.

Definition 2.1 (Finite probability space)**.** A finite probability space is a pair (Ω, p) where Ω is a finite set of "outcomes" and $p : \Omega \to \mathbb{R}$ is a probability distribution, satisfying $p(\omega) \geq 0$ for all $\omega \in \Omega$, and $\sum_{\omega \in \Omega} p(\omega) = 1$.

As suggested in the scenarios of Section 1, there are several ways a probability distribution may be implemented with a classical or quantum circuit. (See [Bel19, Sec. 3] for a somewhat detailed discussion/comparison.) We will prefer the most general one there, Item 2, where a draw from the distribution is obtained by measuring the output of a quantum circuit $\mathcal{P}$ and possibly discarding some of the result. This is a very natural model, though it does not seem to have a common and succinct name; since $\mathcal{P}$ can be used to produce "synthetic data", we will term it a "synthesizer":

Definition 2.2 (Synthesizer)**.** Let p be a probability distribution on Ω. A synthesizer for p is any unitary circuit $\mathcal{P}$ that performs the map

$$\mathcal{P}|\vec{0}\rangle = \sum_{\omega \in \Omega} \sqrt{p(\omega)}\,|\omega\rangle\,|\text{garbage}_\omega\rangle\,, \tag{2.7}$$

where: $|\vec{0}\rangle$ is any easy-to-prepare fixed state (for definiteness, say $|\vec{0}\rangle = |0^k\rangle$ where k is the number of input qubits to $\mathcal{P}$); $|\omega\rangle$ is a normalized vector representing ω; and, $|\text{garbage}_\omega\rangle$ is any normalized "garbage vector". In a typical implementation we would have the space $\Omega = [D]$ (for D a power of 2), with $|\omega\rangle$ being the $\log_2 D$-qubit representation of ω in the computational basis.

Observe that in the typical scenario, if we produce $\mathcal{P}|\vec{0}\rangle$ and measure the first register (ignoring/discarding the second), we indeed obtain the outcome ω with probability $p(\omega)$.

Example 2.3. For $\Omega = \{0, 1, 2, \ldots, 2^n - 1\}$, the uniform distribution p on Ω has, as a synthesizer, the circuit that consists of applying a Hadamard gate to each qubit of $|\vec{0}\rangle = |0^n\rangle$.

Example 2.4. Continuing the previous example, suppose that as in Grover's algorithm, an additional circuit "marks" the outcomes $M \subseteq \Omega$ by producing

$$\sum_{\ell=0}^{2^n-1} |y_\ell\rangle\,|\ell\rangle\,, \quad \text{where } y_\ell = \begin{cases} 1 & \text{if } \ell \in M, \\ 0 & \text{if } \ell \notin M. \end{cases} \tag{2.8}$$

(Here contrary to common convention, we put the "flag register" on the left rather than the right.) The above state may be rewritten as

$$\sqrt{\tfrac{2^n-|M|}{2^n}}\,|0\rangle\,|\text{garbage}_0\rangle + \sqrt{\tfrac{|M|}{2^n}}\,|1\rangle\,|\text{garbage}_1\rangle \tag{2.9}$$

with "garbage vectors"

$$|\text{garbage}_0\rangle = \tfrac{1}{\sqrt{2^n-|M|}} \sum_{\ell \notin M} |\ell\rangle\,, \qquad |\text{garbage}_1\rangle = \tfrac{1}{\sqrt{|M|}} \sum_{\ell \in M} |\ell\rangle\,. \tag{2.10}$$

In this way, the composite algorithm producing the state in Equation (2.9) may be seen as a synthesizer for the two-outcome Bernoulli probability with parameter $p = \frac{|M|}{2^n}$.

Definition 2.5 (Having the code for a distribution)**.** When we use the phrase "having the code" for a distribution p, we refer only to having black-box access to controlled-$\mathcal{P}$ and controlled-$\mathcal{P}^\dagger$, where $\mathcal{P}$ is a synthesizer for p.[7] Of course, if we have white-box access to the circuit $\mathcal{P}$, we can easily produce circuits for controlled-$\mathcal{P}$ and controlled-$\mathcal{P}^\dagger$.

Remark 2.6. Allowing "garbage vectors" in Equation (2.7) is crucial to obtain an acceptable level of generality. Insisting that a synthesizer produce a "coherent" (garbage-free) version of Equation (2.7) (as in the "uniform" model Item 3 of Section 1) would be very limiting. Indeed, efficient coherent synthesizers are typically impossible even when there is an efficient *classical* sampling algorithm for the distribution.[8]

Classical algorithms. The above synthesizer definition also covers the case of having a classical circuit with coin-flip gates that generates draws from p (as in the Item 1 scenario). Such a circuit can be converted into a deterministic circuit that accepts r random bits as input. Then the deterministic circuit can be made reversible (with constant-factor overhead) using classical Toffoli and NOT gates [Ben73]. The resulting circuit will now accept some a ancillary input bits set to 0, and output the original output, along with some additional garbage bits. Next, we make a quantum circuit with the same behavior on classical basis states by replacing all the classical Toffoli and NOT gates with quantum Toffoli and quantum NOT gates. Finally, the original r random input bits can be replaced by r Hadamard gates with with $|0\rangle$ inputs. The result is a synthesizer circuit $\mathcal{P}$ for the probability distribution, with gate complexity only a constant factor larger than that of the original classical circuit.[9]

2.2 Random variables

Now we come to the main object of study in this paper, real random variables. Formally, a (discrete, real) random variable is just a real-valued function on a probability space:

Definition 2.7 (Random variable)**.** Given a finite probability space (Ω, p), a random variable $\boldsymbol{y}$ is defined by a function $y : \Omega \to \mathbb{R}$.

Definition 2.8 (Moments of a random variable)**.** For a random variable $\boldsymbol{y}$, we define the following:

(mean or expected value) $\quad \mu \coloneqq \mathbf{E}[\boldsymbol{y}] = \sum_{\omega\in\Omega} p(\omega)y(\omega)$

(second moment) $\quad s^2 \coloneqq \mathbf{E}[\boldsymbol{y}^2] = \sum_{\omega\in\Omega} p(\omega)y(\omega)^2$

(variance) $\quad \sigma^2 \coloneqq \mathbf{Var}[\boldsymbol{y}] = \mathbf{E}[(\boldsymbol{y}-\mu)^2] = \mathbf{E}[\boldsymbol{y}^2] - \mu^2 = s^2 - \mu^2$

(standard deviation) $\quad \sigma = \mathbf{stddev}[\boldsymbol{y}]$

[7] Grover-type algorithms use $\mathcal{P}^\dagger$, but we point out it is also very reasonable to require controlled-$\mathcal{P}$. Without it, there would be no way to implement a synthesizer for a simple modification of p such as "with probability 1/2 draw from p, with probability 1/2 output 0".

[8] For example, given a graph G, it is easy to uniformly sample from all automorphisms (vertex-labelings) of the graph. But if we could efficiently synthesize a quantum state that is the uniform superposition over all automorphisms, then we would be able to solve Graph Isomorphism efficiently: We would simply create the state for G, the state for H, and check if they are the same or orthogonal.

[9] To see this explicitly worked out, see Appendix A of the arXiv version of [WSK+21]

Remark 2.9. Our notation for the raw second moment, s^2, is not standard (as opposed to the standard notation σ^2 for the variance, aka "central second moment").

Regarding implementation of a random variable $y : \Omega \to \mathbb{R}$, we assume a standard quantum oracle:

Definition 2.10 (Having the code for a random variable)**.** When we use the phrase "having the code" for a random variable $\boldsymbol{y}$ on probability space (Ω, p), this refers to having a synthesizer for p, as well as having access to controlled-$\mathcal{Y}$ and controlled-$\mathcal{Y}^\dagger$, where $\mathcal{Y}$ is any unitary circuit with the behavior

$$\mathcal{Y} |\omega\rangle |0^b\rangle |0^c\rangle = |\omega\rangle |y(\omega)\rangle |0^c\rangle \tag{2.11}$$

for all $\omega \in \Omega$. Here it is assumed the real range of $\boldsymbol{y}$ is encoded using b bits (e.g., in fixed-point representation); the extra c bits are for ancillas.

Remark 2.11. As with classical implementations of probability distributions, given a classical circuit computing $y : \Omega \to \mathbb{R}$ (with appropriate input/output encoding), one can efficiently convert it to a quantum circuit $\mathcal{Y}$ as above (taking care to uncompute garbage).

Remark 2.12. Some readers may find it overly fussy that we have insisted on the mathematical definition of random variables as functions on probability spaces. However, it will be very convenient in our work to think of them in this way.[10]

Consider also the unfussy notion of a random variable $\boldsymbol{y}$ being implemented by a quantum (or classical randomized) circuit $\mathcal{C}$, wherein measuring $\mathcal{C} |\vec{0}\rangle$ (and discarding garbage) directly yields a draw from $\boldsymbol{y}$. In this case, we can formally define $\Omega = \text{range}(\boldsymbol{y})$, define $p(\omega) = \mathbf{Pr}[\boldsymbol{y} = \omega]$, treat $\mathcal{C}$ as a synthesizer for p, and formally take $y : \Omega \to \mathbb{R}$ to be the identity map (so that $\mathcal{Y} = \mathbb{1}$ and $b = c = 0$ in Equation (2.11)).

3 Establishing Theorem 1.3 — Grover with complex phases

In this section we establish Theorem 1.3. First, in Section 3.1 we fully describe the algorithm, which involves setting up a unitary $\mathcal{U}$ and performing phase estimation with an initial state $|\mathbf{1}\rangle$. The next Section 3.2 gives some generic preliminaries on phase estimation. Subsequently, we need to analyze the eigenvalues and eigenvectors of our particular $\mathcal{U}$ — or at least the eigenspaces in which $|\mathbf{1}\rangle$ mostly resides. We introduce some notation in Section 3.3; then in Section 3.4 we derive the key eigenvalue inequalities for our analysis and do two things:

- Show the inequalities easily imply that Quantum Phase Estimation achieves Theorem 1.3, except with worse constants (which nevertheless would suffice to solve our overall Mean Estimation task).
- Show that a sharper analysis of the eigenvalue inequalities would lead to Theorem 1.3 with its constants as stated.

Subsequently in Section 3.5, we give the sharper analysis of the eigenvalue inequalities. In Section 3.6, we observe that using Quantum Phase Estimation as a black box is arguably overkill for our problem (though it makes the analysis succinct); we illustrate how one can instead complete the algorithm via measuring $|\mathbf{1}\rangle$ against $\mathcal{U}^T |\mathbf{1}\rangle$ for $T = \Theta(1/\epsilon)$. Finally, in Section 3.7 we observe that, in a certain sense, all of the eigenvalues and eigenvectors of $\mathcal{U}$ can be described geometrically and somewhat simply. We found that this description did not seem to simplify any of our preceding analysis, though it may aid in intuition.

3.1 Algorithm description

Since our algorithm is essentially just Grover's algorithm with complex phases, it's easy to fully describe the algorithm and its complexity. Proving correctness of the algorithm will then be the goal of the subsequent subsections.

Let $([D], p)$ be a probability space implemented by synthesizer $\mathcal{P}$, so

$$\mathcal{P} |\vec{0}\rangle = \sum_{\ell=1}^{D} \sqrt{p(\ell)} |\ell\rangle |\text{garbage}_\ell\rangle \tag{3.12}$$

[10] Such readers may also recall, e.g., how much simpler it is to prove Linearity of Expectation from the definition $\mathbf{E}[\boldsymbol{y}] = \sum_\omega p(\omega) y(\omega)$ than from $\mathbf{E}[\boldsymbol{y}] = \sum_y \mathbf{Pr}[\boldsymbol{y} = y] x$.

as in Equation (2.7). Here we have written $\ell \in [D]$ instead of $\omega \in \Omega$ to make the notation less laborious (but note that the $|\ell\rangle$ above, really $|\omega\rangle$, need not literally denote the ℓth standard basis vector). Let $\boldsymbol{y}$ be a real random variable defined by $y : [D] \to \mathbb{R}$ and computed by circuit $\mathcal{Y}$ as in Equation (2.11). In this section we will write

$$y_\ell \text{ instead of } y(\ell). \tag{3.13}$$

We may now define the key unitary used by our algorithm that accomplishes the Main Task from Theorem 1.3. As in Grover's algorithm, it is composed of two parts:

$$\mathcal{U} = \mathrm{REFL}_p \cdot \mathrm{ROT}_y. \tag{3.14}$$

Definition 3.1. The operator REFL_p (essentially the "Grover diffusion operator" vis-a-vis p) is defined by

$$\mathrm{REFL}_p = \mathcal{P}(2|\vec{0}\rangle\langle\vec{0}| - \mathbb{1})\mathcal{P}^\dagger. \tag{3.15}$$

Definition 3.2. The operator ROT_y (the "phase oracle") is defined by

$$\mathrm{ROT}_y |\ell\rangle |\mathrm{garbage}_\ell\rangle = e^{\mathrm{i}\alpha_\ell} |\ell\rangle |\mathrm{garbage}_\ell\rangle \quad \forall \ell \in [D], \tag{3.16}$$

where the angles $\alpha_\ell := -2 \arctan y_\ell$ are defined so that

$$e^{\mathrm{i}\alpha_\ell}(1 + \mathrm{i}y_\ell) = 1 - \mathrm{i}y_\ell. \tag{3.17}$$

Remark 3.3. The operator REFL_p is evidently efficiently computable using two applications of "the code": one application of $\mathcal{P}$ and one application $\mathcal{P}^\dagger$. Additionally, as we will later use quantum phase estimation (or, at least, the Hadamard test), we will in fact require controlled-$\mathcal{U}$, not just $\mathcal{U}$ itself, and hence will really need applications of controlled-$\mathcal{P}$ and controlled-$\mathcal{P}^\dagger$.

The operator ROT_y is also efficiently computable using two applications of "the code": Given $\mathcal{Y}$ as in Equation (2.11), we adjoin $|0^b\rangle$ and apply $\mathcal{Y}$ to get $|\ell\rangle |\mathrm{garbage}_\ell\rangle |y_\ell\rangle$.[11] We may then employ a classical routine (the computational efficiency and precision of which are discussed in Appendix A) to compute $|\alpha_\ell\rangle$ from $|y_\ell\rangle$, multiply by the phase $e^{\mathrm{i}\alpha_\ell}$, uncompute with the help of $\mathcal{Y}^\dagger$, and thus finally reach $e^{\mathrm{i}\alpha_\ell} |\ell\rangle |\mathrm{garbage}_\ell\rangle$. Recall again that we will eventually use controlled-$\mathcal{U}$, not just $\mathcal{U}$, and hence again we will really need one application each of controlled-$\mathcal{Y}$ and controlled-$\mathcal{Y}^\dagger$.

Thus overall (controlled-)$\mathcal{U}$ can be efficiently implemented with four uses of "the code" for $\boldsymbol{y}$.

The main claim in the later analysis is that the state $\mathcal{P}|\vec{0}\rangle$ has high overlap with the eigenvectors of $\mathcal{U}$ of eigenphase approximately $2|\mu|$. Then employing phase estimation with precision $\epsilon/6$, which requires $O(1/\epsilon)$ uses of $\mathcal{U}$, will allow us to distinguish the two ranges of $|\mu|$ and complete the proof of Theorem 1.3.

3.2 Generic phase estimation setup In this subsection, let $\mathcal{U}$ denote *any* generic unitary operator on $\mathbb{C}^D$. Suppose we perform phase estimation (or a simpler, Grover-like algorithm) with $\mathcal{U}$ and "starting state" $|\sigma\rangle$. Then we will need to know about the eigenvalue(s) of $\mathcal{U}$ corresponding to the eigenvector(s) that $|\sigma\rangle$ is close to. Let us introduce some notation to facilitate this:

Notation 3.4. Fix an eigendecomposition of $\mathcal{U}$,

$$\mathcal{U} = \sum_{j=1}^{D} e^{\mathrm{i}\theta_j} |u_j\rangle\langle u_j|, \tag{3.18}$$

with $-\pi < \theta_j \le \pi$. Given some $|\sigma\rangle$, we express it in $\mathcal{U}$'s eigenbasis as

$$|\sigma\rangle = \sum_{j=1}^{D} \hat{\sigma}_j |u_j\rangle, \qquad \hat{\sigma}_j := \langle u_j|\sigma\rangle. \tag{3.19}$$

When $|\sigma\rangle$ is a *unit* vector, Pythagorus tells us the squared coefficients $|\hat{\sigma}_1|^2, \ldots, |\hat{\sigma}_D|^2$ form a probability distribution on $[D]$. In this case we will write $\boldsymbol{j} \sim J_{\mathcal{U}}(|\sigma\rangle)$ to denote that $\boldsymbol{j}$ is drawn according to this probability distribution; we will also write $\boldsymbol{\theta} \sim \Theta_{\mathcal{U}}(|\sigma\rangle)$ to denote that $\boldsymbol{\theta}$ is the random angle formed by drawing $\boldsymbol{j} \sim J_{\mathcal{U}}(|\sigma\rangle)$ and then setting $\boldsymbol{\theta} = \theta_{\boldsymbol{j}}$.

[11] Formally, we will also need to adjoin ancillas, but these will always be set to all-$|0\rangle$'s and restored to all-$|0\rangle$'s, and thus may be safely ignored. As is conventional, we will avoid further mention of them.

Remark 3.5. One could say the random variable $\boldsymbol{\theta} \sim \Theta_{\mathcal{U}}(|\sigma\rangle)$ is the output of "Idealized Phase Estimation"; that is, phase estimation making no error. Indeed, the *actual* behavior of Quantum Phase Estimation [Kit95, CEMM98] when run with $\mathcal{U}$ and $|\sigma\rangle$ is that, after $O(\log(1/\delta)/\epsilon)$ applications of controlled-$\mathcal{U}$, the output is a random variable $\boldsymbol{\theta}'$ with the following property:

There is a probabilistic coupling between $\boldsymbol{\theta}$ and $\boldsymbol{\theta}'$ under which $\mathbf{Pr}[|\boldsymbol{\theta} - \boldsymbol{\theta}'| > \epsilon] \le \delta$.

On the topic of closeness between $J_{\mathcal{U}}(\cdot)$ distributions, the following fact relates the fidelity between two different starting states and the Hellinger distance between their associated $J_{\mathcal{U}}(\cdot)$'s:

Proposition 3.6. *Given $\mathcal{U}$ as in Notation 3.4, suppose $|\sigma\rangle, |\tau\rangle \in \mathbb{C}^d$ are unit vectors. Write $q_1, \dots, q_d$ (respectively, $r_1, \dots, r_d$) for the probabilities of $J_{\mathcal{U}}(|\sigma\rangle)$ (respectively $J_{\mathcal{U}}(|\tau\rangle)$). Then we have the following Bhattacharyya coefficient / Hellinger-squared bound:*

$$BC(q,r) \ge |\langle\sigma|\tau\rangle|; \qquad \textit{in other words,} \qquad H(q,r)^2 \le 2(1 - |\langle\sigma|\tau\rangle|). \tag{3.20}$$

Proof. Writing $\lambda_j = \sqrt{r_j/q_j}$ (and taking $\lambda_j = 0$ if $q_j = 0$), we have

$$\begin{aligned} H(q,r)^2 = \sum_j q_j(1-\lambda_j)^2 = 2\left(1 - \sum_j \sqrt{q_j}\sqrt{r_j}\right) &= 2\left(1 - \sum_j |\langle u_j|\sigma\rangle| \cdot |\langle u_j|\tau\rangle|\right) \\ &\le 2\left(1 - \left|\sum_j \langle\sigma|u_j\rangle \cdot \langle u_j|\tau\rangle\right|\right) = 2(1 - |\langle\sigma|\tau\rangle|). \end{aligned} \tag{3.21}$$

□

When it comes to analyzing eigenvalues $e^{\mathrm{i}\theta}$ of $\mathcal{U}$, we will use the following quantity — quaintly called the *haversine* of angle θ — to measure how "nontrivial" rotation-by-θ is:

Notation 3.7. For any $\theta \in \mathbb{R}$ we may write $\operatorname{hav}\theta := \left|\frac{1-e^{\mathrm{i}\theta}}{2}\right|^2 = \frac{1-\cos\theta}{2} = \left(\sin\frac{\theta}{2}\right)^2 \in [0,1]$.

From Equation (3.19) we have $\left(\frac{\mathbb{1}-\mathcal{U}}{2}\right)^{\pm 1}|\sigma\rangle = \sum_j \hat{\sigma}_j \left(\frac{1-e^{\mathrm{i}\theta_j}}{2}\right)^{\pm 1}|u_j\rangle$, and thus the above fact implies:

Proposition 3.8. *Let $|\sigma\rangle$ be a unit vector, and assume for $\boldsymbol{\theta} \sim \Theta_{\mathcal{U}}(|\sigma\rangle)$ that $\boldsymbol{\theta}$ is never 0. Then*

$$\left\| \left(\frac{\mathbb{1}-\mathcal{U}}{2}\right)^{\pm 1} |\sigma\rangle \right\|^2 = \mathop{\mathbf{E}}_{\boldsymbol{\theta}\sim\Theta_{\mathcal{U}}(|\sigma\rangle)} \left[(\operatorname{hav}\boldsymbol{\theta})^{\pm 1}\right]. \tag{3.22}$$

3.3 Quantum states corresponding to complex random variables We now return to our particular $\mathcal{U} = \mathrm{REFL}_p \cdot \mathrm{ROT}_y$ as described in Equation (3.14). Let us introduce some notation that allows us to conveniently talk about states on which this unitary acts.

Notation 3.9. Let $z_1, \dots, z_D$ be any complex numbers. We may think of this list as defining a *complex-valued* random variable $\boldsymbol{z}$ on $([D], p)$. (To draw from $\boldsymbol{z}$, first choose $\boldsymbol{\ell} \in [D]$ according to p and then set $\boldsymbol{z} = z_{\boldsymbol{\ell}}$.) Then we will also define the (not necessarily unit) vector

$$|\boldsymbol{z}\rangle = \sum_{\ell=1}^{D} z_\ell \sqrt{p(\ell)}\, |\ell\rangle\, |\mathrm{garbage}_\ell\rangle\,. \tag{3.23}$$

In particular, referring to Equation (3.12) we have

$$\mathcal{P}\,|\vec{0}\rangle = |\mathbf{1}\rangle\,, \tag{3.24}$$

where $\mathbf{1}$ denotes the random variable on $([D], p)$ that is constantly 1.

It is easy to compute the following:

Fact 3.10. For complex-valued random variables $\boldsymbol{w}, \boldsymbol{z}$ on $([D], p)$ we have

$$\langle \boldsymbol{w}|\boldsymbol{z}\rangle = \mathbf{E}_p[\overline{\boldsymbol{w}}\boldsymbol{z}] = \sum_{\ell=1}^{D} p(\ell)\overline{w}_\ell z_\ell. \tag{3.25}$$

In particular, $\langle \mathbf{1}|\boldsymbol{z}\rangle = \mathbf{E}_p[\boldsymbol{z}]$.

Remark 3.11. We will often consider *non*-unit vectors $|\boldsymbol{z}\rangle$. The vector $|\boldsymbol{z}\rangle$ is only a properly normalized quantum state if $\mathbf{E}_p[|\boldsymbol{z}|^2] = 1$. (For example, $|\mathbf{1}\rangle$ *is* a valid quantum state.)

Remark 3.12. Even if $|\boldsymbol{w}\rangle, |\boldsymbol{z}\rangle$ are unit vectors, and thus may be considered quantum states, one should *not* consider them to be identical if they are equal up to a global phase. The reason is our algorithm will eventually introduce a control qubit (for phase estimation purposes), which will make global phases into relative phases.

With this notation for states, we can now examine what the reflection operator does to a state. Recalling our new notation (particularly Equation (3.24)), our reflection operator defined in Definition 3.1 is

$$\mathrm{REFL}_p = \mathcal{P}(2|\vec{0}\rangle\langle\vec{0}| - \mathbb{1})\mathcal{P}^\dagger = 2\,|\mathbf{1}\rangle\langle\mathbf{1}| - \mathbb{1}. \tag{3.26}$$

From Fact 3.10 we see that $\mathrm{REFL}_p\,|\boldsymbol{z}\rangle = |\boldsymbol{z}_{\mathrm{refl}}\rangle$, where

$$\boldsymbol{z}_{\mathrm{refl}} = 2\mathbf{E}_p[\boldsymbol{z}] - \boldsymbol{z} \tag{3.27}$$

is the random variable in which each z_ℓ is replaced with its reflection through the "barycenter" $\mathbf{E}_p[\boldsymbol{z}]$.

3.4 Eigenvalue analysis In this section we do the eigenvector/eigenvalue analysis of our operator $\mathcal{U} = \mathrm{REFL}_p \cdot \mathrm{ROT}_y$, as a function of the random variable $\boldsymbol{y}$. We will use the notation

$$\mu = \mathbf{E}_p[\boldsymbol{y}], \qquad s^2 = \mathbf{E}_p[\boldsymbol{y}^2]. \tag{3.28}$$

The key vectors for our analysis are the "starting state" $|\mathbf{1}\rangle$, and the following vector:

$$|\mathbf{1} + \mathbf{i}\boldsymbol{y}\rangle \coloneqq \sum_{\ell=1}^{D}(1 + \mathrm{i}y_\ell)\sqrt{p(\ell)}\,|\ell\rangle\,|\mathrm{garbage}_\ell\rangle\,. \tag{3.29}$$

Using Fact 3.10, we have:

Fact 3.13. $\langle \mathbf{1} + \mathbf{i}\boldsymbol{y}|\mathbf{1} + \mathbf{i}\boldsymbol{y}\rangle = \mathbf{E}_p[\mathbf{1}^2 + \boldsymbol{y}^2] = 1 + s^2$; and, $\langle \mathbf{1}|\mathbf{1} + \mathbf{i}\boldsymbol{y}\rangle = 1 + \mathrm{i}\mu$, so $|\langle \mathbf{1}|\mathbf{1} + \mathbf{i}\boldsymbol{y}\rangle| = \sqrt{1+\mu^2} \geq 1$.

From this we see that if s is small (as we will assume), then $|\mathbf{1} + \mathbf{i}\boldsymbol{y}\rangle$ is close to being a unit vector, and this unit vector is close to $|\mathbf{1}\rangle$. Let us introduce a normalized version of the vector:

Notation 3.14. We write $|\widetilde{\mathbf{1}}_y\rangle = \frac{1}{\sqrt{1+s^2}}\,|\mathbf{1} + \mathbf{i}\boldsymbol{y}\rangle$, a unit vector.

Perhaps the key intuition behind the analysis is that if $\mu = 0$, then the vector $|\mathbf{1} + \mathbf{i}\boldsymbol{y}\rangle$ is *fixed* by $\mathcal{U}$ (i.e., it is an eigenvector of eigenvalue 1). The following proposition generalizes this fact:

Proposition 3.15. $\dfrac{\mathbb{1} - \mathcal{U}}{2}\,|\mathbf{1} + \mathbf{i}\boldsymbol{y}\rangle = \mathrm{i}\mu\,|\mathbf{1}\rangle\,.$

Proof. This follows from

$$\begin{aligned}
\mathcal{U}\,|\mathbf{1} + \mathbf{i}\boldsymbol{y}\rangle &= \mathrm{REFL}_p \cdot \mathrm{ROT}_y\,|\mathbf{1} + \mathbf{i}\boldsymbol{y}\rangle && (3.30)\\
&= \mathrm{REFL}_p\,|\mathbf{1} - \mathbf{i}\boldsymbol{y}\rangle \qquad (\text{using Equation (3.17)}) && (3.31)\\
&= |\mathbf{1} + \mathbf{i}(\boldsymbol{y} - 2\boldsymbol{\mu})\rangle \qquad (\text{using Equation (3.27)}) && (3.32)\\
&= |\mathbf{1} + \mathbf{i}\boldsymbol{y}\rangle - 2i\mu\,|\mathbf{1}\rangle\,.
\end{aligned}$$

□

Proposition 3.16. *Writing* $\widetilde{\boldsymbol{\theta}} \sim \Theta_{\mathcal{U}}(|\widetilde{\mathbf{1}}_y\rangle)$, *we have* $\mathbf{E}[\mathrm{hav}\,\widetilde{\boldsymbol{\theta}}] = \mu^2/(1+s^2)$.

Proof. This is immediate by taking the squared-length of both sides in Proposition 3.15 and then applying Proposition 3.8 with exponent +1. □

Somewhat peculiarly, for $\boldsymbol{\theta} \sim \Theta_{\mathcal{U}}(|\mathbf{1}\rangle)$ we can also determine the expected *reciprocal* of $\operatorname{hav}\boldsymbol{\theta}$:

Proposition 3.17. *Writing $\boldsymbol{\theta} \sim \Theta_{\mathcal{U}}(|\mathbf{1}\rangle)$, we have $\mathbf{E}[(\operatorname{hav}\boldsymbol{\theta})^{-1}] = (1+s^2)/\mu^2$.*
(Technically, we must assume that $\boldsymbol{\theta}$ is never 0 and that $\mu \neq 0$.)

Proof. Rearranging the statement of Proposition 3.15 gives

$$\left(\frac{\mathbb{1}-\mathcal{U}}{2}\right)^{-1}|\mathbf{1}\rangle = \frac{1}{\mathrm{i}\mu}|\mathbf{1}+\mathbf{i}\boldsymbol{y}\rangle\,. \tag{3.33}$$

The proof is completed by taking the squared-length on both sides and then applying Proposition 3.8 with exponent -1. □

Recall that, assuming s is small, we have that $|\mathbf{1}\rangle$ is close to $|\widetilde{\mathbf{1}}_y\rangle$, and hence $\boldsymbol{\theta}$ should be similar in distribution to $\widetilde{\boldsymbol{\theta}}$. The preceding two propositions therefore suggest that $\operatorname{hav}\boldsymbol{\theta} = \sin^2(\boldsymbol{\theta}/2)$ ought to concentrate around $\mu^2/(1+s^2)$; i.e., $|\boldsymbol{\theta}|$ ought to concentrate around $2|\mu|$ (again, when assuming s is small). Indeed, we can use them to establish the following:

Theorem 3.18. *For certain constants $s_0, c_0, 1/C_0, \delta_0 > 0$, the following holds: Provided $s \leq s_0$, for $\boldsymbol{\theta} \sim \Theta_{\mathcal{U}}(|\mathbf{1}\rangle)$ we have*

$$\mathbf{Pr}\Big[c_0 \cdot 2|\mu| \leq |\boldsymbol{\theta}| \leq C_0 \cdot 2|\mu|\Big] \geq 1-\delta_0. \tag{3.34}$$

In particular (see Corollary 3.23), we may take $s_0 = \frac{1}{16}$, $c_0 = \frac{4}{5}$, $C_0 = \frac{5}{4}$, $\delta_0 = \frac{2}{9}$.

With these specific "in particular" constants, we can complete the proof of Theorem 1.3 almost immediately by using phase estimation. On the other hand, achieving these constants is slightly fiddly; hence, we defer this to Section 3.5. For now, we illustrate how Theorem 3.18 can be proven in a very simple way, allowing for worse constants. (As we note in Remarks 3.20 and 4.1 and Section 3.6, these worse constants are still sufficient for giving elementary proofs of our main results Theorems 1.1 and 1.3.)

Proof of Theorem 3.18 with worse constants. We establish the theorem with

$$s_0 = .001, \quad c_0 = .05, \quad C_0 = 50, \quad \delta_0 = .005. \tag{3.35}$$

Beginning with a technicality, note that the conclusion of our theorem is continuous with respect to infinitesimally perturbing $\mathcal{U}$; thus we may assume without loss of generality that $\mu, \boldsymbol{\theta} \neq 0$ always holds. Now applying Markov's inequality to Proposition 3.17 we get

$$\mathbf{Pr}[(\operatorname{hav}\boldsymbol{\theta})^{-1} > 350(1+s^2)/\mu^2] \leq \frac{1}{350} < .003. \tag{3.36}$$

Assuming $s \leq .001$, we conclude that except with probability less than .003 we have

$$(\operatorname{hav}\boldsymbol{\theta})^{-1} \leq 350(1+s^2)/\mu^2 \implies \sin^{-2}(\boldsymbol{\theta}/2) \leq 400/\mu^2 \implies (|\mu|/20)^2 \leq \sin^2(\boldsymbol{\theta}/2) \leq (\boldsymbol{\theta}/2)^2. \tag{3.37}$$

Hence

$$\mathbf{Pr}[|\boldsymbol{\theta}| < .05 \cdot 2|\mu|] \leq .003. \tag{3.38}$$

On the other hand, applying Markov's inequality to Proposition 3.16 gives

$$\mathbf{Pr}[\operatorname{hav}\widetilde{\boldsymbol{\theta}} > 1000\mu^2/(1+s^2)] \leq .001, \tag{3.39}$$

and we conclude that except with probability at most .001 we have

$$\operatorname{hav}\widetilde{\boldsymbol{\theta}} \leq 1000\frac{\mu^2}{1+s^2} \leq 1000\mu^2 \implies |\sin(\boldsymbol{\theta}/2)| \leq \sqrt{1000}|\mu| \implies |\boldsymbol{\theta}/2| \leq \tfrac{\pi}{2}\sqrt{1000}|\mu| < 50 \cdot 2|\mu|. \tag{3.40}$$

Hence

$$\mathbf{Pr}[|\widetilde{\boldsymbol{\theta}}| > 50 \cdot 2|\mu|] \le .001. \tag{3.41}$$

Finally, Fact 3.13 implies the fidelity bound $|\langle \mathbf{1}|\widetilde{\mathbf{1}}_y\rangle|^2 \ge (1+\mu^2)/(1+s^2) \ge 1/(1+s^2)$, from which it is not hard to deduce

$$\mathbf{Pr}[|\boldsymbol{\theta}| > 50 \cdot 2|\mu|] \le .001 + s \le .002 \tag{3.42}$$

under the assumption $s \le .001$. (One can, e.g., use Helstrom's theorem [Hel76] for this.) Combining Inequalities (3.38) and (3.42) completes the proof. □

As mentioned, by using the version of Theorem 3.18 with good constants (proved in Section 3.5), we can easily complete the proof of Theorem 1.3 by appealing to phase estimation. In fact, with this method we do not even need the "$|\mu| \le 2\epsilon$" part of "case (ii)" in the theorem statement; just $|\mu| \ge \epsilon$ is sufficient. Thus we have the following slightly stronger form of Theorem 1.3:

Theorem 3.19 (A stronger form of Theorem 1.3)**.** *There is a computationally efficient quantum algorithm with the following properties: Given a parameter $\epsilon > 0$ and the code for a random variable $\boldsymbol{y}$, promised to satisfy $s = \sqrt{\mathbf{E}[\boldsymbol{y}^2]} \le \frac{1}{16}$, the algorithm uses $O(1/\epsilon)$ samples and distinguishes (with confidence at least 2/3) between the cases (i) $|\mu| \le \epsilon/2$ and (ii) $|\mu| \ge \epsilon$, where $\mu = \mathbf{E}[\boldsymbol{y}]$.*

Remark 3.20. The reader will notice that we have taken $s \le \frac{1}{16}$ as a hypothesis here, whereas Theorem 1.3 has $s \le 1$. However one may observe that the theorem's statement is insensitive to multiplying both $\boldsymbol{y}$ and ϵ by any fixed constant $0 < s_0 < 1$ (such as $s_0 = \frac{1}{16}$[12]); this only affects the sample complexity by a constant factor.

Proof of Theorem 3.19. We perform Quantum Phase Estimation on the unitary $\mathcal{U}$ and starting state $|\mathbf{1}\rangle$, with accuracy parameter $\epsilon/6$ and confidence parameter $\delta = 1/9$, producing output $\boldsymbol{\theta}'$. As described in Remark 3.5, this can be done with $O(1/\epsilon)$ uses of controlled-$\mathcal{U}$, which implies $O(1/\epsilon)$ uses of the code for $\boldsymbol{y}$ (Remark 3.3) The result is that, for $\boldsymbol{\theta} \sim \Theta_{\mathcal{U}}(|\mathbf{1}\rangle)$ being the "Idealized Phase Estimation" output, there is a coupling such that $|\boldsymbol{\theta} - \boldsymbol{\theta}'| \le \epsilon/6$ except with probability at most 1/9. Thus from Theorem 3.18, we get that

$$\tfrac{4}{5}|\mu| - \epsilon/12 \le |\boldsymbol{\theta}'/2| \le \tfrac{5}{4}|\mu| + \epsilon/12 \tag{3.43}$$

except with probability at most $\frac{1}{9} + \frac{2}{9} = 1/3$. Now on one hand, in case (i) we have

$$|\mu| \le \epsilon/2 \implies |\boldsymbol{\theta}'/2| \le (5/8)\epsilon + \epsilon/12 < .71\epsilon. \tag{3.44}$$

On the other hand, in case (ii) we have

$$|\mu| \ge \epsilon \implies |\boldsymbol{\theta}'/2| \ge (4/5)\epsilon - \epsilon/12 > .71\epsilon. \tag{3.45}$$

Thus we can distinguish the two cases with confidence at least 2/3 by deciding whether $\boldsymbol{\theta}' \gtrless 1.42\epsilon$. □

3.5 Sharper eigenvalue analysis In this section, we establish Theorem 3.18 with the explicit good constants. The intuition behind the analysis is the following observation: Suppose for a moment that $\widetilde{\boldsymbol{\theta}}$ and $\boldsymbol{\theta}$ from Propositions 3.16 and 3.17 were identically distributed. Then, writing $\boldsymbol{h} = \frac{\sqrt{1+s^2}}{|\mu|}\sqrt{\mathrm{hav}\,\boldsymbol{\theta}}$, these would imply

$$\mathbf{E}[(\boldsymbol{h} - \boldsymbol{h}^{-1})^2] = \frac{1+s^2}{\mu^2}\mathbf{E}[\boldsymbol{h}] + \frac{\mu^2}{1+s^2}\mathbf{E}[\boldsymbol{h}^{-1}] - 2\,\mathbf{E}[1] = 1 + 1 - 2 = 0, \tag{3.46}$$

and hence

$$\boldsymbol{h} - \boldsymbol{h}^{-1} \equiv 0 \quad\Longrightarrow\quad \boldsymbol{h} \equiv 1 \quad\Longrightarrow\quad \sqrt{\mathrm{hav}\,\boldsymbol{\theta}} = |\sin(\boldsymbol{\theta}/2)| \equiv \frac{|\mu|}{\sqrt{1+s^2}}. \tag{3.47}$$

The next theorem (of which Theorem 3.18 is a corollary) makes this idea rigorous by taking care of the fact that we don't quite have $\widetilde{\boldsymbol{\theta}} \not\equiv \boldsymbol{\theta}$:

[12] For the sake of implementation it is nicer if the constant is a power of 2 so that adjusting the oracle $\mathcal{Y}$ is simple.

Theorem 3.21. *Fix any $C \geq 1$ and assume $s \leq \frac{1}{C}$. Then for $\boldsymbol{\theta} \sim \Theta_{\mathcal{U}}(|\mathbf{1}\rangle)$,*

$$\mathbf{Pr}\left[|\sin(\boldsymbol{\theta}/2)| \notin \frac{|\mu|}{\sqrt{1+s^2}} \cdot \left[\frac{1}{1+Cs}, \frac{1}{1-Cs}\right]\right] \leq \frac{2}{C^2}. \tag{3.48}$$

The proof will use a numerical lemma:

Lemma 3.22. *For real numbers h, λ with $h > 0$, it holds that $(1-h^{-1})^2 \leq (1-\lambda)^2 + (\lambda h - h^{-1})^2$.*

Proof. The difference of the two sides is $\lambda^2(1-h)^2 + 2h(\lambda - h^{-1})^2 \geq 0$. □

Proof of Theorem 3.21. As in our proof of Theorem 3.18 with worse constants, we may assume without loss of generality that $\mu, \boldsymbol{\theta} \neq 0$ always holds. We use the notation of Notation 3.4 and Proposition 3.6, letting $q_1, \ldots, q_D$ (respectively, $r_1, \ldots, r_D$) denote the probabilities of $J_{\mathcal{U}}(|\mathbf{1}\rangle)$ (respectively, $J_{\mathcal{U}}(|\widetilde{\mathbf{1}}_y\rangle)$) and $\lambda_j = \sqrt{r_j/q_j}$. Now combining Fact 3.13 with Inequality (3.21) from Proposition 3.6 gives

$$\sum_j q_j(1-\lambda_j)^2 = 2\left(1 - \sum_j q_j\lambda_j\right) \leq 2(1 - |\langle \mathbf{1}|\widetilde{\mathbf{1}}_y\rangle|) = 2(1 - \sqrt{1+\mu^2}/\sqrt{1+s^2}) \leq 2(1 - 1/\sqrt{1+s^2}) \leq s^2. \tag{3.49}$$

At the same time, if we define

$$h_j = \frac{\sqrt{1+s^2}}{|\mu|}\sqrt{\operatorname{hav}\theta_j}, \tag{3.50}$$

we can restate Proposition 3.17 (which has its technical assumption satisfied) and Proposition 3.16 as

$$\sum_j q_j h_j^{-2} = 1, \quad \text{and} \quad \sum_j r_j h_j^2 = 1 \iff \sum_j q_j \lambda_j^2 h_j^2 = 1. \tag{3.51}$$

Let $\boldsymbol{j} \sim J_{\mathcal{U}}(|\mathbf{1}\rangle)$, so $\mathbf{Pr}[\boldsymbol{j} = j] = q_j$. We may express $\boldsymbol{\theta} = \theta_{\boldsymbol{j}}$, and also write $\boldsymbol{h} = h_{\boldsymbol{j}}$ and $\boldsymbol{\lambda} = \lambda_{\boldsymbol{j}}$. Then we may summarize Equation (3.51) and Inequality (3.49) as

$$\mathbf{E}[\boldsymbol{h}^{-2}] = \mathbf{E}[\boldsymbol{\lambda}^2\boldsymbol{h}^2] = 1, \qquad \mathbf{E}[(1-\boldsymbol{\lambda})^2] = 2(1 - \mathbf{E}[\boldsymbol{\lambda}]) \leq s^2 \tag{3.52}$$

(where we used $\mathbf{E}[\boldsymbol{\lambda}^2] = 1$). These imply

$$\mathbf{E}[(\boldsymbol{\lambda}\boldsymbol{h} - \boldsymbol{h}^{-1})^2] = \mathbf{E}[\boldsymbol{\lambda}^2 h^2] + \mathbf{E}[\boldsymbol{h}^{-2}] - 2\,\mathbf{E}[\boldsymbol{\lambda}] = 2(1 - \mathbf{E}[\boldsymbol{\lambda}]) \leq s^2. \tag{3.53}$$

Now applying Lemma 3.22 in expectation and using the above facts, we get

$$\mathbf{E}[(1-\boldsymbol{h}^{-1})^2] \leq \mathbf{E}[(1-\boldsymbol{\lambda})^2] + \mathbf{E}[(\boldsymbol{\lambda}\boldsymbol{h} - \boldsymbol{h}^{-1})^2] \leq s^2 + s^2 = 2s^2. \tag{3.54}$$

Thus by Markov's inequality, except with probability at most $2/C^2$, we have

$$(1-\boldsymbol{h}^{-1})^2 \leq (Cs)^2 \implies |1-\boldsymbol{h}^{-1}| \leq Cs \implies \boldsymbol{h} \in \left[\frac{1}{1+Cs}, \frac{1}{1-Cs}\right], \tag{3.55}$$

(recall $Cs \leq 1$). Putting in the definition of $\boldsymbol{h} = \frac{\sqrt{1+s^2}}{|\mu|}\sqrt{\operatorname{hav}\boldsymbol{\theta}}$ and recalling $\operatorname{hav}\theta = \sin^2(\theta/2)$ completes the proof. □

Note that given the $1 \pm O(s)$ error range of the preceding theorem, the distinctions between $|\sin(\boldsymbol{\theta}/2)|$ and $|\boldsymbol{\theta}|^2$ and between $\mu^2/(1+s^2)$ and μ^2 are more minor. Thus the preceding theorem essentially gives that $|\boldsymbol{\theta}/2| = |\mu| \pm O(|\mu|s)$ with high probability. With some slightly tedious numeric estimates, we can get the following more usable corollary; it immediately implies Theorem 3.18 with its strong constants by taking $t = 1$ (and using $1 - \frac{3}{16} \geq \frac{4}{5}$):

Corollary 3.23. *Fix any $t \geq 1$. Then Theorem 3.18 holds with $s_0 = \frac{1}{16t}$, $c_0 = 1 - 3ts_0$, $C_0 = 1 + 4ts_0$, and $\delta_0 = \frac{2}{9t^2}$.*

Proof. Given t, write $\eta = ts$ and note that $\eta \le \frac{1}{16}$ assuming $s \le s_0$. Now selecting $C = 3t$ in Theorem 3.21, we get that except with probability at most $\frac{2}{9t^2}$:

$$2\cdot|\sin(\boldsymbol{\theta}/2)| \ge 2|\mu|\cdot\frac{1}{\sqrt{1+(\eta/t)^2}}\cdot\frac{1}{1+3\eta}, \qquad 2\cdot|\sin(\boldsymbol{\theta}/2)| \le 2|\mu|\cdot\frac{1}{\sqrt{1+(\eta/t)^2}}\cdot\frac{1}{1-3\eta} \tag{3.56}$$

$$\ge 2|\mu|\cdot\frac{1}{\sqrt{1+\eta^2}}\cdot\frac{1}{1+3\eta} \ge 2|\mu|\cdot(1-3\eta) \qquad \le 2|\mu|\cdot\frac{1}{1-3\eta} \tag{3.57}$$

Since $|\boldsymbol{\theta}| \ge 2\cdot|\sin(\boldsymbol{\theta}/2)|$, we now have the needed lower bound for Inequality (3.34):

$$\mathbf{Pr}[|\boldsymbol{\theta}| < (1-3ts)\cdot 2|\mu|] \le \frac{2}{9t^2}. \tag{3.58}$$

As for the upper bound, let us first weakly observe that $|\mu| \le s \le s_0 \le \frac{1}{16}$, which together with $\eta \le \frac{1}{16}$ means Inequality (3.57) implies $2\cdot|\sin(\boldsymbol{\theta}/2)| \le 2\cdot\frac{1}{16}\cdot\frac{1}{1-3(1/16)} = \frac{2}{13}$. On this range of $|\boldsymbol{\theta}|$, it holds that $|\boldsymbol{\theta}| \le (13\sin^{-1}\frac{1}{13})\cdot 2\cdot|\sin(\boldsymbol{\theta}/2)|$. Combined with Inequality (3.57) and $\eta \le \frac{1}{16}$ this yields

$$|\boldsymbol{\theta}| \le 32\sin^{-1}(\tfrac{1}{13})\cdot|\mu| \implies \tfrac{1}{24}|\boldsymbol{\theta}|^3 \le .63|\mu|^3 \le .63\cdot\frac{1}{16}|\mu|ts \le .04|\mu|\eta, \tag{3.59}$$

where we used $|\mu| \le s \le s_0 \le \frac{1}{16} \le \frac{1}{16}t$. Finally we come to the main use of Inequality (3.57):

$$|\boldsymbol{\theta}| - \tfrac{1}{24}|\boldsymbol{\theta}|^3 \le 2\cdot|\sin(\boldsymbol{\theta}/2)| \le 2|\mu|\cdot\frac{1}{1-3\eta} \le 2|\mu|\cdot(1+\tfrac{48}{13}\eta) \tag{3.60}$$

using $\eta \le \frac{1}{16}$ again. From this, Inequality (3.59), and $\frac{48}{13} + .02 \le 4$, we deduce

$$\mathbf{Pr}[|\boldsymbol{\theta}| > (1+4ts)\cdot 2|\mu|] \le \frac{2}{9t^2}, \tag{3.61}$$

the needed upper bound in Inequality (3.34). □

3.6 A more elementary algorithm One could argue that our algorithm's use of Quantum Phase Estimation is overkill: we have high-probability bounds for the location of $\boldsymbol{\theta}$ (so doing estimations in superposition is not needed), and this location is restricted to two narrow, separated regions: $\boldsymbol{\theta} \approx 2|\mu|$ or $\boldsymbol{\theta} \approx 0$. Thus it is possible to use a more "elementary", Grover-like method to go from Theorem 3.18 to Theorem 1.3, as we now demonstrate.

We first show this by appealing to the very strong constants achieved in Corollary 3.23; we then sketch how even the simply obtained constants from Equation (3.35) suffice.

So suppose first we take $t = 10$ in Corollary 3.23, leading to $\delta_0 = 2/900 \le .003$. We can also take our upper bound s_0 on s as small as we please (see Remark 3.20); let us therefore take it small enough that $4ts_0 \le .01$. We thereby obtain from Corollary 3.23 that

$$\Pr_{\boldsymbol{\theta}\sim\Theta_{\mathcal{U}}(|\mathbf{1}\rangle)}[.99\cdot 2|\mu| \le |\boldsymbol{\theta}| \le 1.01\cdot 2|\mu|] \ge .997. \tag{3.62}$$

Recall we are trying to distinguish the cases (i) $|\mu| \le \epsilon/2$ and (ii) $\epsilon \le |\mu| \le 2\epsilon$. Note also that $|\mu| \le s \le s_0 < .0003$ by assumption, meaning we can assume $\epsilon \le .0003$ without loss of generality. Suppose we now take

$$T = \lfloor \pi/(3\epsilon) \rfloor \tag{3.63}$$

(noting that $\epsilon \le .0003$ means the floor changes T's value by a factor of at most 1.0003). Then Inequality (3.62) implies

$$\Pr_{\boldsymbol{\tau}\sim\Theta_{\mathcal{U}_T}(|\mathbf{1}\rangle)}[.989\cdot\pi/(3\epsilon)\cdot 2|\mu| \le |\boldsymbol{\tau}| \le 1.01\cdot\pi/(3\epsilon)\cdot 2|\mu|] \ge .997. \tag{3.64}$$

(We changed .99 to .989 to account for the floor on T.) So except with probability at most .003 we have the following:

$$\text{(3.65)} \qquad \text{case (i)} \implies |\boldsymbol{\tau}| \le 1.01 \cdot \pi/3 \implies \cos \boldsymbol{\tau} \ge +.49,$$

$$\text{(3.66)} \qquad \text{case (ii)} \implies .989 \cdot 2\pi/3 \le |\boldsymbol{\tau}| \le 1.01 \cdot 4\pi/3 \implies \cos \boldsymbol{\tau} \le -.46.$$

Now suppose we perform the Hadamard Test on $|\mathbf{1}\rangle$ with the unitary $\mathcal{U}^T$, whose application uses the code for $\boldsymbol{y}$ only $O(T) = O(1/\epsilon)$ times. (Recall this means adjoining $|+\rangle$ to $|\mathbf{1}\rangle$, applying controlled-$\mathcal{U}^T$, and then measuring the new qubit in the $|\pm\rangle$ basis.) With probability at least .997, we get back a random variable $\boldsymbol{b} \in \{\pm 1\}$ with expectation $\cos \boldsymbol{\tau}$. Thus in case (i) we get $\boldsymbol{b} = +1$ with probability at least $\frac{1}{2} + \frac{1}{2}(+.49) - .003 \ge 2/3$, and in case (ii) we get $\boldsymbol{b} = +1$ with probability at most $\frac{1}{2} + \frac{1}{2}(-.46) + .003 \le 1/3$. Thus we can distinguish the two cases with confidence 2/3 based on the measurement outcome $\boldsymbol{b}$ of the Hadamard Test. An example depiction of the process underlying this more elementary algorithm is shown in Figures 2 and 3.

We now sketch how one can go from Theorem 3.18 to Theorem 1.3 in a similarly elementary way even with the worse constants from Equation (3.35). For this, we will need to weaken the statement of Theorem 1.3 so that case (i) is "$|\mu| \le c_1\epsilon$" for a very small constant $c_1 > 0$. This is not without loss of generality, as changing the upper bound on s is. Nevertheless, as we will show (see Remark 4.1), this does not affect our ability to deduce Theorem 1.1 from Theorem 1.3.

Recall that with the constants from Equation (3.35) we have

$$\text{(3.67)} \qquad \Pr_{\boldsymbol{\theta} \sim \Theta_{\mathcal{U}}(|\mathbf{1}\rangle)} [.05 \cdot 2|\mu| \le |\boldsymbol{\theta}| \le 50 \cdot 2|\mu|] \ge .995$$

provided $s \le .001$. Now suppose we are in case (ii), $\epsilon \le |\mu| \le 2\epsilon$, so that $.1\epsilon \le |\boldsymbol{\theta}| \le 200\epsilon$ except with probability .005. The trick is to divide this range of multiplicative-width 2000 into, say, $\lceil \log_2 2000 \rceil = 11$ intervals of multiplicative-width at most 2: say, $[.1\epsilon, .2\epsilon], [.2\epsilon, .4\epsilon], \ldots [100\epsilon, 200\epsilon]$. The most frequently encountered such interval for $\boldsymbol{\theta}$ occurs with probability at least $.995/11 \ge .09$. Now suppose we take 11 different values of $T = \Theta(1/\epsilon)$ so that the scaled-by-T intervals approximate $[2\pi/3, 4\pi/3]$. (Note that each T is at most $22/\epsilon$.) Then for at least one such T, the Hadamard Test applied to $|\mathbf{1}\rangle$ and $\mathcal{U}^T$ will output $\boldsymbol{b} = +1$ with probability slightly bounded away from 1; at most $.09 \cdot q + .91$ for $q \approx \frac{1}{2} + \frac{1}{2}\max\{\cos(2\pi/3), \cos(4\pi/3)\} = .25$, in particular, at most .94.

On the other hand, suppose we take case (i) in Theorem 1.3 to be "$|\mu| \le .0001\epsilon$". Then one can infer that except with probability at most .995 over the outcome of $\boldsymbol{\theta} \sim \Theta_{\mathcal{U}}(|\mathbf{1}\rangle)$ we will have $|\boldsymbol{\theta}| \le .01\epsilon$ and hence $|\boldsymbol{\tau}| \le .01T \le .22$ for *each* of the 11 choices of T. Thus $\cos \boldsymbol{\tau} \ge \frac{1}{2} + \frac{1}{2}\cos(.22) \ge .987$ and we conclude the Hadamard Test will report $\boldsymbol{b} = +1$ with probability at least $.987 - .005 \ge .98$.

In summary, we have 11 different "coins" $\boldsymbol{b}$, with the guarantee that in case (i) all come up Heads (+1) with probability at least .98, and in case (ii) at least one comes up Heads with probability at most .94. The two cases may therefore be distinguished with confidence at least 2/3 using a constant number of "coin flips" (each of which uses $O(T) = O(1/\epsilon)$ samples).

Remark 3.24. We observe that the fundamental feature of the constants from Equation (3.35) that make them acceptable is that $\delta_0 \ll 1/\log(C_0/c_0)$. Indeed, our simpler proof of Theorem 3.18 achieves $\delta_0 \le O(c_0 + 1/C_0)$.

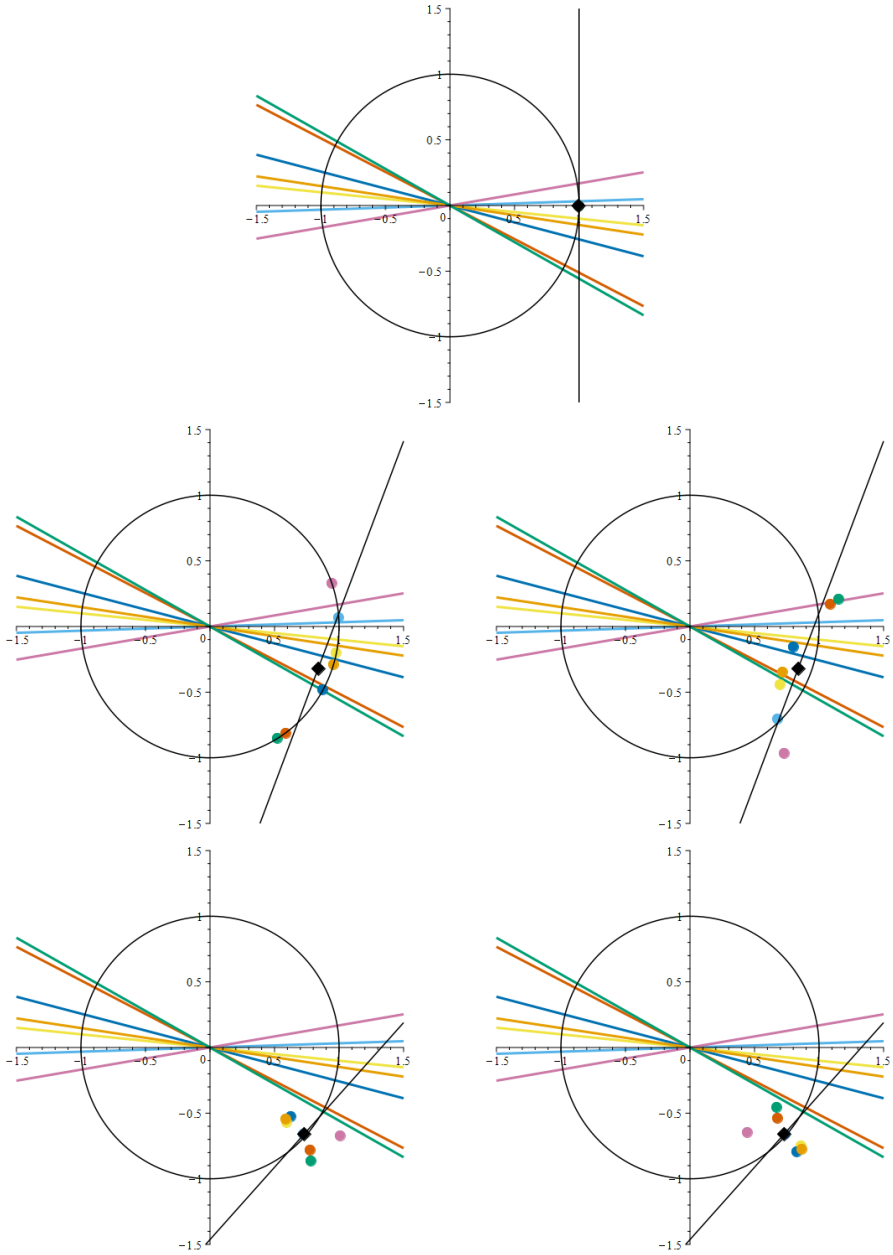

Figure 2: An illustration of $\mathcal{U}$'s action. We have the uniform distribution p over $D = 7$ outcomes, $y_1, \ldots, y_7$, with values approximately $-.169$, $-.032$, $.101$, $.148$, $.258$, $.511$, $.557$ (pink, light blue, light yellow, orange, dark blue, dark orange, green). We have $\mu = \mathbf{E}_p[\boldsymbol{y}] \approx .196 \approx \pi/(2 \cdot 8)$, and $s^2 = \mathbf{E}_p[\boldsymbol{y}^2] = .1$. The colored line corresponding to y_ℓ is at angle α_ℓ; i.e., it passes through $1 - \mathrm{i}y_\ell$ in the complex plane. The five diagrams above show $|\mathbf{1}\rangle$, $\mathrm{ROT}_y |\mathbf{1}\rangle$, $\mathrm{REFL}_p \cdot \mathrm{ROT}_y |\mathbf{1}\rangle = \mathcal{U} |\mathbf{1}\rangle$, $\mathrm{ROT}_y \cdot \mathcal{U} |\mathbf{1}\rangle$, and $\mathrm{REFL}_p \cdot \mathrm{ROT}_y \cdot \mathcal{U} |\mathbf{1}\rangle = \mathcal{U}^2 |\mathbf{1}\rangle$ (from top to bottom, left to right). The black diamond shows the barycenter of the points (through which the reflection occurs), and the black line is orthogonal to it.

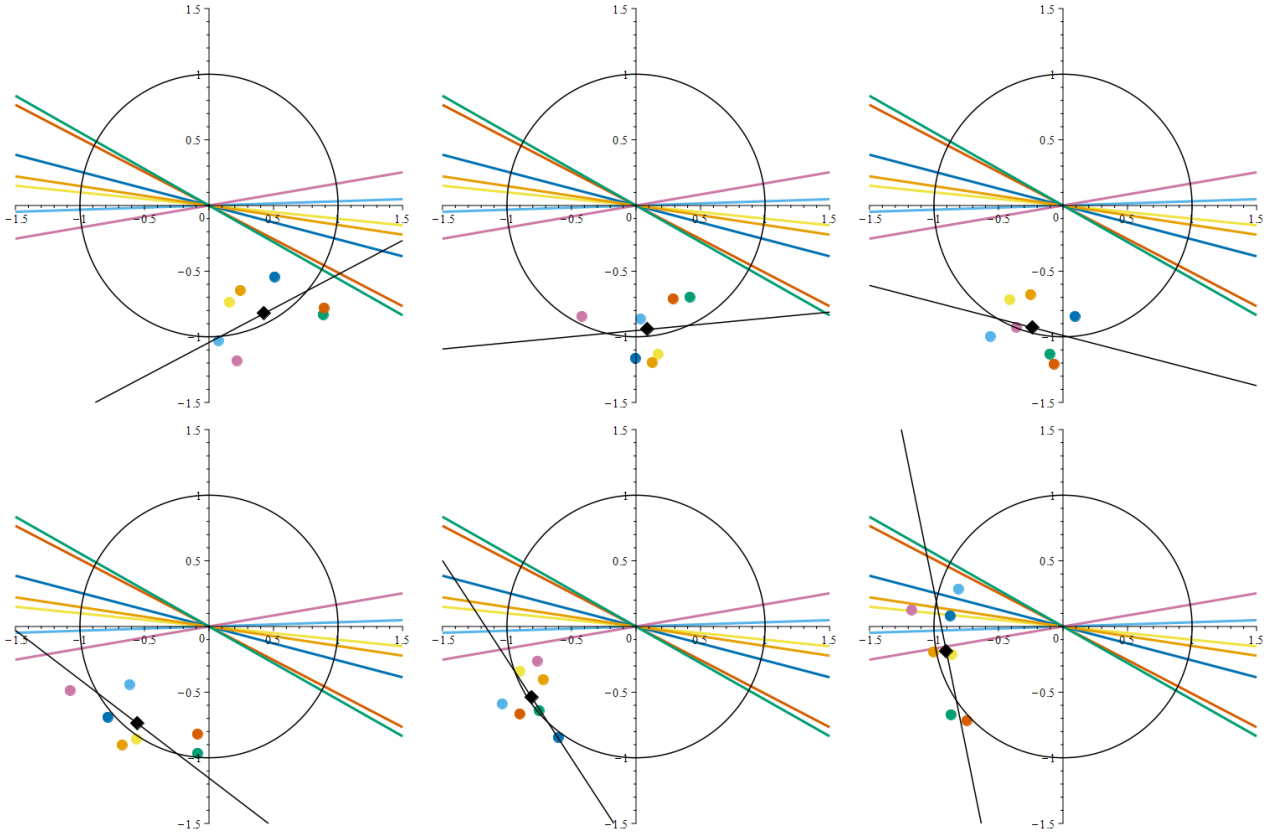

Figure 3: Continuing Figure 2, we illustrate $\mathcal{U}^t|\mathbf{1}\rangle$ for $t = 3 \dots 8$. The result of applying the Hadamard Test at any time is a $\{\pm 1\}$-valued random variable with mean equal to the horizontal displacement of the black diamond. After $8 \approx \pi/(2|\mu|)$ applications of $\mathcal{U}$, this displacement is close to -1.

3.7 Describing all eigenvalues and eigenvectors Here we give a geometric description of all the eigenvalues and eigenvectors of $\mathcal{U}$. Their description is simple enough that one might discover them through intuition. An alternative route (to the eigenvalues, at least) is to observe that

$$\mathcal{U} = \mathrm{REFL}_p \cdot \mathrm{ROT}_y = \mathcal{P}(2|\vec{0}\rangle\langle\vec{0}| - \mathbb{1})\mathcal{P}^\dagger \cdot \mathrm{ROT}_y = 2\mathcal{P}|\vec{0}\rangle\langle\vec{0}| \cdot \mathrm{ROT}_y - \mathrm{ROT}_y \tag{3.68}$$

is a rank-one update of the diagonal matrix ROT_y (up to a minus sign). As such, one can give an explicit rational expression (see e.g. [Ion01]) whose roots are the eigenvalues of $\mathcal{U}$. By working through the details one can arrive at the below geometric description of the eigenvalues.

It will actually be slightly more convenient to consider $\mathcal{U}^\dagger = \mathrm{ROT}_y^\dagger \cdot \mathrm{REFL}_p$, which has the same eigenvectors as $\mathcal{U}$ and the complex-conjugated eigenvalues. To seek the eigenvectors of $\mathcal{U}^\dagger$, consider the random variable $|\mathbf{1} + \mathbf{i}\boldsymbol{y}\rangle$, which we saw is an eigenvector of $\mathcal{U}$ (and $\mathcal{U}^\dagger$) of eigenvalue 1 if $\mu = \mathbf{E}_p[\boldsymbol{y}] = 0$. Plot each value of $|\mathbf{1} + \mathbf{i}\boldsymbol{y}\rangle$ (i.e., $1 + y_\ell$) as a point in the complex plane, along with the line passing through it and the origin, as in the diagram on the left in Figure 4. Now imagine slowly rotating all the lines, always marking the points where they cross the vertical line corresponding to real-part 1. Also, keep track of the mean of these points (that is, the weighted mean under p), which will naturally also be on the same vertical line. Pause rotation whenever this barycenter touches the real axis (i.e., becomes $1 + 0\mathrm{i}$), as in the diagram on the right in Figure 4. Say that after pausing we have rotated by θ and the current points form the random variable $|\mathbf{1} + \mathbf{i}\boldsymbol{y}'\rangle$. Then we claim that $|\mathbf{1} + \mathbf{i}\boldsymbol{y}'\rangle$ is an eigenvector of $\mathcal{U}^\dagger$ with eigenvalue $e^{-\mathrm{i}\cdot 2\theta}$ (hence an eigenvector of $\mathcal{U}$ with eigenvalue $e^{\mathrm{i}\cdot 2\theta}$).

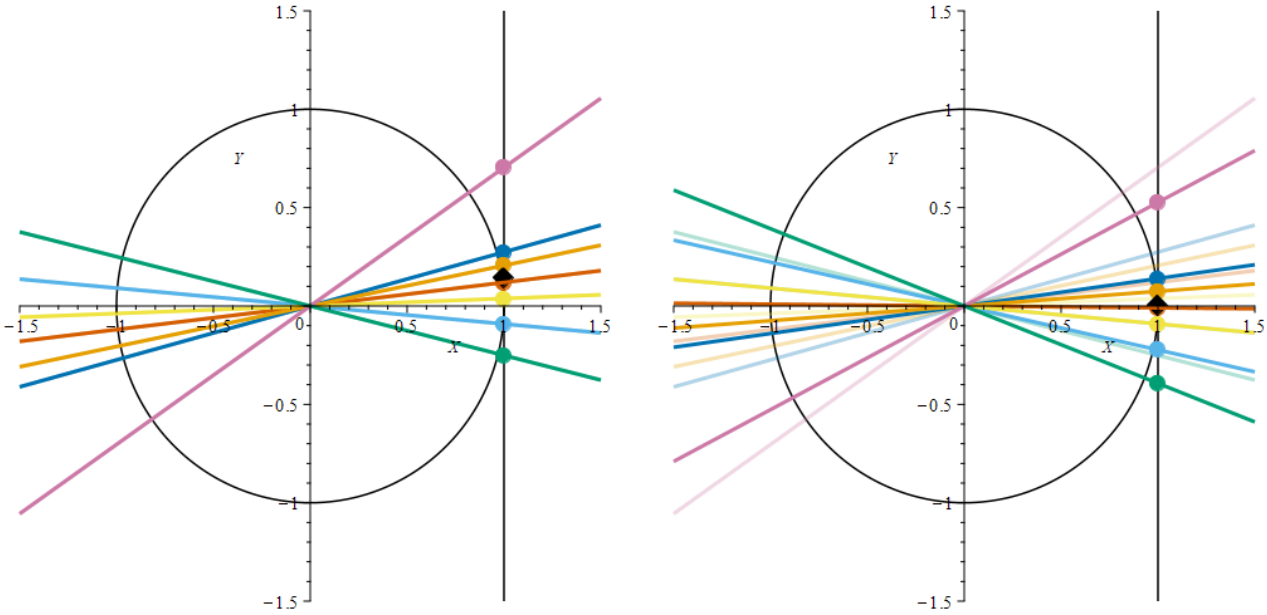

Figure 4: An example with the uniform distribution p on $D = 7$ outcomes again. The y_ℓ values are .560, .258, .057, −.045, −.088, −.250, −.494 (yellow, light blue, dark blue, green, pink, light orange, dark orange). In contrast to Figures 2 and 3, here the associated colored lines pass through $1 + \mathrm{i}y_\ell$ (rather than $1 - \mathrm{i}y_\ell$). As before, the black diamond depicts the barycenter. On the left, we have the initial points $|\mathbf{1} + \mathbf{i}\boldsymbol{y}\rangle$. On the right, after rotating the lines slightly (with the original lines shown with light color), the new points' barycenter is on the real axis. Twice the angle of this rotation is an eigenphase of $\mathcal{U}$, with the point locations on the right forming the associated eigenvector.

To verify this claim, first observe that $\mathrm{REFL}_p|\mathbf{1} + \mathbf{i}\boldsymbol{y}'\rangle = |\mathbf{1} - \mathbf{i}\boldsymbol{y}'\rangle$, since $\mathbf{E}_p[\boldsymbol{y}'] = 0$. Next, recall that $\mathrm{ROT}_y^\dagger$ rotates the ℓth point by $2\arctan y_\ell$, so that $1 - \mathrm{i}y_\ell$ moves to $1 + \mathrm{i}y_\ell$. Since $1 - \mathrm{i}y'_\ell$ is at angle $-\theta$ from $1 - \mathrm{i}y_\ell$, it follows that $\mathrm{ROT}_y^\dagger$ moves the ℓth point so that it is at angle $-\theta$ from $1 + \mathrm{i}y_\ell$, and hence angle -2θ from its starting location of $1 + \mathrm{i}\boldsymbol{y}'$. Thus we see that the composition $\mathcal{U}^\dagger = \mathrm{ROT}_y^\dagger \cdot \mathrm{REFL}_p$ indeed multiplies $|\mathbf{1} + \mathbf{i}\boldsymbol{y}'\rangle$ by $e^{-\mathrm{i}2\theta}$, as claimed.

So far we have explained how to find one eigenvector/value of $\mathcal{U}^\dagger$. To find more, we simply keep rotating the lines, waiting for "black diamond" to cross the real axis; we show the next two such occurrences in Figure 5. Note that as we rotate (clockwise, in the figures), the barycenter moves monotonically downward until such time as one of the colored lines rotates to a vertical position (taking the associated colored point's height to $-\infty$); at this point, the black diamond's height "wraps around" to $+\infty$, and then continues monotonically downward. From this observation, it is easy to see that if $\boldsymbol{y}$'s D values are all distinct, we will get the full complement of D distinct eigenvectors. (Otherwise eigenvalues will occur with multiplicity, but one can reduce to the distinct case by infinitesimal perturbations.)

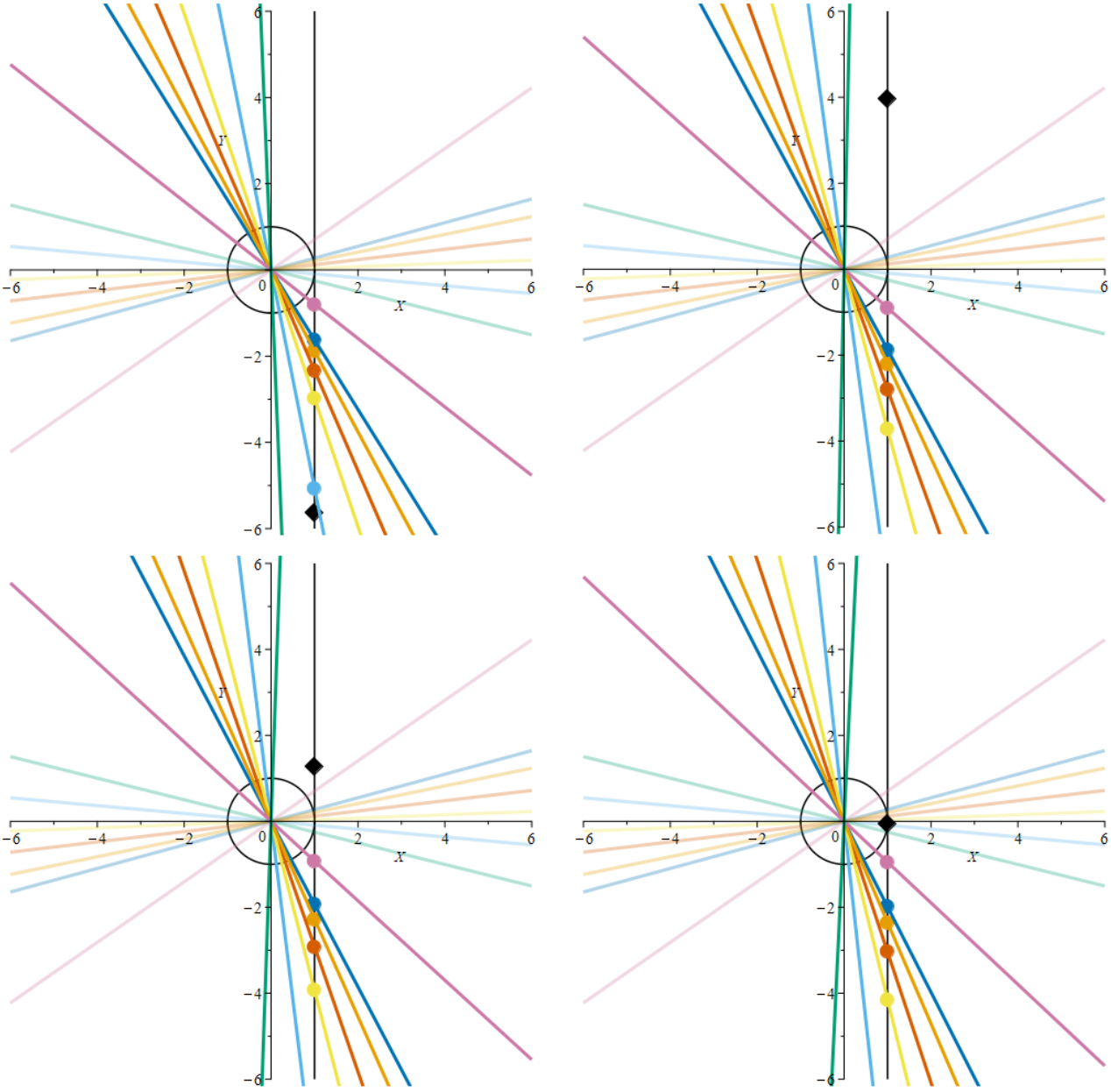

Figure 5: Continuing from Figure 4, we depict further rotations (with the diagram zoomed out by a factor of 4). As the green line approaches vertical, the height of the barycenter (black diamond) approaches $-\infty$. As the green line crosses vertical, the black diamond's height becomes $+\infty$ and begins descending again, until we reach a second eigenvector in the final diagram. The light blue point is outside the picture in the final three diagrams, and the green point is outside the picture in all four diagrams.

We observe that each of the eigenvectors $|\mathbf{1} + \mathbf{i}\boldsymbol{y}'\rangle$ we have described is unnormalized; the unit-length version of it is $|\mathbf{1} + \mathbf{i}\boldsymbol{y}'\rangle / \sqrt{1 + \mathbf{E}_p[(\boldsymbol{y}')^2]}$. Hence the overlap of our algorithm's starting vector, $|\mathbf{1}\rangle$, with the eigenvalue is $1/\sqrt{1 + \mathbf{E}_p[(\boldsymbol{y}')^2]}$. In the figures above, we have depictions of unnormalized eigenvectors as colored points on the $\Re = 1$ line with black diamond on the real axis. Thus the overlap of the *normalized* eigenvector with $|\mathbf{1}\rangle$ is high if and only if the colored points' vertical heights are not "too extreme" on average. In fact, using this viewpoint, it's not too hard to design initial random variables $\boldsymbol{y}$ for which *no single* eigenvector of eigenphase $|\theta| \approx 2\mu$ has large

overlap with $|\mathbf{1}\rangle$; only the whole subspace of them does. This justifies why our analysis in Section 3.4 cannot work (as in Grover's algorithm) by simply identifying one or two eigenvectors of eigenphase around $2|\mu|$ with which $|\mathbf{1}\rangle$ has large overlap.

4 Consequences

Having established Theorem 1.3 which achieves the "Main Task", we will now layer on top a sequence of improvements that will culminate in our solution to the Mean Estimation problem from Theorem 1.1. All but the last of these improvements is simply a classical reduction based on "standard" ideas (binary search, successive halving, etc.). By the end of these classical improvements we will have achieved the results in Table 1 up through [Mon15], but with the improved (optimal) sample complexity of $O(n)$. (We remark that several of the aforementioned "standard" ideas appeared earlier in the works from Table 1.) The final step, which gets us to the optimal Mean Estimation algorithm, requires combining several of the preceding reductions with the quantum Quantile Finding algorithm of Hamoudi [Ham21]. As discussed in Remark 1.2 and Appendix A, all of our algorithms are also gate efficient; however for clarity of exposition, we will focus only on query (sample) complexity in this section.

We describe the improvements below as a sequence of problems to be solved. In all of these problems, the setup remains the same: We have access to "the code" for a random variable $\boldsymbol{y}$, we write

$$\mu = \mathbf{E}_p[\boldsymbol{y}], \qquad \sigma^2 = \mathbf{Var}_p[\boldsymbol{y}], \qquad s^2 = \mathbf{E}_p[\boldsymbol{y}^2], \tag{4.69}$$

and we are trying to solve the given problem with success probability at least $2/3$.

The confidence parameter, and the "log log trick". All of the following problems have the usual feature that the arbitrarily selected confidence parameter of $2/3$ can be boosted to $1-\delta$ at the expense of $O(\log(1/\delta))$ repetitions (followed by taking the majority/median answer). This allows us to chain together constantly many solutions at constant expense; we will omit explicit mention of this standard technique. However, in two cases we will need the following "log log trick" (which has certainly been used before, but doesn't seem to have a standard name).

Assume we plan to solve a sequence of problems with decreasing "accuracy" parameters $1 \geq \eta_1 \geq \eta_2 \geq \cdots \geq \eta_t \geq \eta^*$. Here the values of η_1 and η^* should be fixed in advance, but we do not require that the other η_j's are; it is acceptable if η_{j+1}'s value is chosen only after the solution for accuracy η_j is found. However we *do* always require that $\eta_{j+1} \leq \eta_j/R$ for some fixed constant $R < 1$. It is also assumed that that solving a problem with accuracy η_j and confidence $1-\delta_j$ can be done at a "cost" of $O(1/\eta_j)\cdot\log(1/\delta_j)$.

If we could ignore the issue of confidence, the costs would be upper-bounded by $O(\cdot)$ of a geometric series of ratio $R > 1$, beginning at 1 and ending just past $1/\eta^*$. Hence the total cost would be bounded by the *final* cost of $O(1/\eta^*)$, up to a constant factor depending only on $R-1$. Our goal is to achieve this cost, while properly taking into account the confidence parameter. If, naively, we decided to take $\delta_j = \delta$ for all j, then we would have to take $\delta \leq 1/(3T)$, where $T = O(\log(1/\eta^*))$ is an upper bound on the number of problems solved. This would incur an extra multiplicative cost of $O(\log(1/\delta)) = O(\log T) = O(\log\log(1/\eta^*))$.

To evade this extra "log log" factor, we may solve the jth problem with confidence parameter, say,

$$\delta_j = \exp(-C(\eta_j/\eta^*)^{1/2}), \tag{4.70}$$

where $C = C(R)$ is a certain constant. (Here the exponent $1/2 \in (0,1)$ was chosen arbitrarily.) Note that the algorithm only needs to know η_j, η^* to set this δ_j, not the values of $\eta_{j+1}, \eta_{j+2}, \ldots$. Now on one hand, if the number of stages ends up being t, the union bound implies the total failure probability is at most

$$\sum_{j=1}^{t} \delta_j = \sum_{j=1}^{t} \exp(-C(\eta_j/\eta^*)^{1/2}) \leq \exp(-C) + \exp(-CR^{1/2}) + \exp(-CR^{2/2}) + \exp(-CR^{3/2}) + \cdots \leq 1/3, \tag{4.71}$$

provided $C = C(R)$ is large enough. On the other hand, the total cost is $O(\cdot)$ of

$$\sum_{j=1}^{t} (1/\eta_j)\cdot \tfrac{1}{2}C(\eta_j/\eta_t)^{1/2} = O((1/\eta_t)^{1/2})\cdot\sum_{j=1}^{t}(1/\eta_j)^{1/2} \leq O((1/\eta_t)^{1/2})\cdot O((1/\eta_t)^{1/2}) = O(1/\eta_t), \tag{4.72}$$

where the inequality used that the sum is bounded by a geometric series (of ratio $R^{1/2} > 1$) with final term $1/\eta_t$.

4.1 The classical reductions We begin with the Main Task:

Problem 1 (Main Task)

Input: Parameter $0 < \epsilon \le 1$.
Promise: $s \le 1$; and, either (i) $|\mu| \le c\epsilon$ or else (ii) $\epsilon \le |\mu| \le 2\epsilon$ holds.
Output: Which of (i) or (ii) holds.

Our Theorem 1.3 shows that this problem, with $c = 1/2$, can be solved with $O(1/\epsilon)$ uses of the code for $\boldsymbol{y}$.

Remark 4.1. Here we wrote "c" more generally so we can illustrate that any universal constant $0 < c < 1$ will be acceptable.

Next we show that a solution to the above problem can be used to solve a slightly more general problem where we have to decide if μ is close to some general target $\widehat{\mu}$, not necessarily 0:

Problem 2 (Main Task given target $\widehat{\mu}$)

Input: Parameter $0 < \epsilon \le 1$, and preliminary estimate $\widehat{\mu} \in [-1, 1]$.
Promise: $s \le 1$; and, either (i) $|\widehat{\mu} - \mu| \le c\epsilon$ or else (ii) $\epsilon \le |\widehat{\mu} - \mu| \le 2\epsilon$ holds.
Output: Which of (i) or (ii) holds.

Lemma 4.2. *We can solve Problem 2 with $O(1/\epsilon)$ queries to the code for $\boldsymbol{y}$.*

Proof. To solve Problem 2 for general $\widehat{\mu} \in [-1, 1]$, let $\boldsymbol{y}' = \boldsymbol{y} - \widehat{\mu}$.[13] This has

$$(s')^2 := \mathbf{E}_p[(\boldsymbol{y}')^2] \le 2\mathbf{E}_p[\boldsymbol{y}^2] + 2\widehat{\mu}^2 \le 2 \cdot 1 + 2 \cdot 1^2 = 4, \tag{4.73}$$

where we used $(a-b)^2 \le 2a^2 + 2b^2$. So if we further define $\boldsymbol{y}'' = \boldsymbol{y}'/2$, we get $(s'')^2 := \mathbf{E}_p[(\boldsymbol{y}'')^2] \le 1$. Now it suffices to apply our solution for Problem 1 to $\boldsymbol{y}''$, with $\epsilon/2$ in place of its ϵ. □

The next upgrade is to actually estimate the mean of $\boldsymbol{y}$, using use our solution to the general decision problem above. The idea is to use a form of binary search; this will in addition need the log log trick.

Problem 3 (Mean estimation, promised second moment at most 1)

Input: Parameter $0 < \epsilon \le 1$.
Promise: $s \le 1$.
Output: An estimate $\widehat{\mu}$ such that $|\widehat{\mu} - \mu| \le \epsilon$.

Lemma 4.3. *We can solve Problem 3 with $O(1/\epsilon)$ queries to the code for $\boldsymbol{y}$.*

Proof. Given our algorithm for Problem 2, repeating it $O(\log(1/\delta))$ times and taking the majority answer yields the ability to do the following:

(4.74) For any $\widehat{\mu}$, assuming $|\widehat{\mu} - \mu| \le c\epsilon'$ or $\epsilon' \le |\widehat{\mu} - \mu| \le 2\epsilon'$,
with $O(\log(1/\delta)/\epsilon')$ queries we can distinguish, except with probability at most δ.

Note this is of the form needed for the log log trick, with the "accuracy" parameter being ϵ'. We will be repeatedly using Equation (4.74) in a kind of binary search, with a sequence of ϵ' values starting at 1, decreasing by a factor of $1 - c$ or less at each stage, and terminating with a value at least $\epsilon/2$. Thus the log log trick tells us the final query complexity will be $O(1/\epsilon)$, as desired.

[13] Given the code for $\boldsymbol{y}$, it is easy to produce the new code for $\boldsymbol{y}'$. We defer all similar such observations to Appendix A on gate complexity.

Our binary search will aim to ensure that in its jth stage, μ is guaranteed to be in the interval $I_j = [a_j, b_j]$. We may start with $I_1 = [-1, 1]$; this is guaranteed to contain μ because

$$\mu^2 = \mathbf{E}_p[\boldsymbol{y}]^2 \le \mathbf{E}_p[\boldsymbol{y}^2] = s^2 \le 1, \tag{4.75}$$

the last inequality by the promise of Problem 3.

In the jth stage of the binary search, we employ (4.74), with

$$\epsilon' = \epsilon_j := |I_j|/2 \quad \text{and} \quad \widehat{\mu} = a_j + c\epsilon_j. \tag{4.76}$$

As mentioned, the initial ϵ' value is 1.

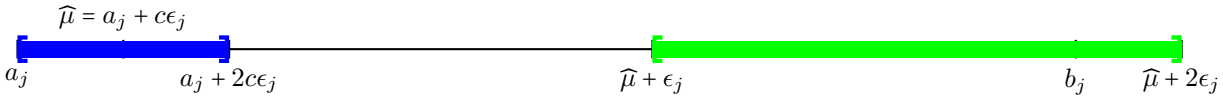

Figure 6: In each step of the binary search procedure, we eliminate either the left interval or the right interval from consideration, which reduces the interval we're left with by a constant factor.

Assuming (by virtue of the log log trick) that all results obtained from (4.74) are correct, let us describe how $I_2, I_3, \ldots$ may be chosen.

- Supposing that in the jth stage we get from (4.74) that $\epsilon_j \le |\widehat{\mu} - \mu| \le 2\epsilon_j$. Then it must be that $|\widehat{\mu} - \mu| \not\le c\epsilon_j$. Thus we may take

$$I_{j+1} = [a_j + 2c\epsilon_j, b_j] = [a_j + c|I_j|, b_j]. \tag{4.77}$$

- Supposing that in the jth stage we get from (4.74) that $|\widehat{\mu} - \mu| \le c\epsilon_j$. Then it must be that $\epsilon_j \le |\widehat{\mu} - \mu| \le 2\epsilon_j$ fails to hold. It can't be that $|\widehat{\mu} - \mu| > 2\epsilon_j = |I_j|$, and hence it must be that $|\widehat{\mu} - \mu| < \epsilon_j$. Thus we may take

$$I_{j+1} = [a_j, a_j + (3/4)|I_j|] \supseteq [a_j, a_j + (c + 1/2)|I_j|] = [a_j, \widehat{\mu} + \epsilon_j]. \tag{4.78}$$

In either case, observe that $|I_{j+1}| \le (1 - c)|I_j|$. That is, the widths of our intervals become smaller by a factor of $1 - c$ or less at each stage, as promised for the log log trick. We may terminate the search once we reach some I_{t+1} with $|I_{t+1}| \le \epsilon$; thus indeed the final use of Equation (4.74) has accuracy parameter at least $\epsilon/2$ so we get total query cost $O(1/\epsilon)$. □

Our next upgrade will be to achieve the result of [BHT98] from Table 1, namely optimal mean estimation for Bernoulli random variables. This uses the standard trick of "successive halving". Besides illustrating that it can be achieved via our methods, we will actually *need* this result as a lemma for our final mean estimation algorithm.

Problem 4 (Bernoulli mean estimation)

Input: Positive integer n.
Promise: $\boldsymbol{y} \in \{0, 1\}$ always.
Output: An estimate $\widehat{\mu}$ such that $|\widehat{\mu} - \mu| \le \sigma/n$.

Lemma 4.4. *We can solve Problem 4 with $O(n)$ queries to the code for $\boldsymbol{y}$.*

Proof. Let $p = \mathbf{Pr}[\boldsymbol{y} = 1]$, so $\mu = p$ and $\sigma = \sqrt{p(1-p)}$. It is convenient to first reduce to the case of small p, say $p \le 3/4$. We can do this using $O(1)$ queries by employing our solution to Problem 3 with $\epsilon = 1/4$. If this yields an estimate $\widehat{\mu} \ge 1/2$, then we can be confident $\mu = p \ge 1/4$. In this case, we simply replace $\boldsymbol{y}$ with $1 - \boldsymbol{y}$ and subtract our final estimate from 1; this leaves σ unchanged and achieves $p \le 3/4$. Note that $\sigma \ge \frac{1}{2}\sqrt{p}$, so it suffices for our algorithm to estimate p to within an additive $\sqrt{p}/(2n)$. Because of this, if ever the algorithm determines that $p \le 1/(4n^2)$, it may acceptably output the estimate $\widehat{\mu} = 0$.

Our algorithm now proceeds in stages, always maintaining an upper bound $\overline{p}$ on p. Initially, $\overline{p} = 3/4$. So long as $\overline{p} > 1/(4n^2)$, the algorithm attempts to lower $\overline{p}$ by a constant factor. It does this by applying the algorithm for Problem 3 on the random variable $\boldsymbol{y}' \coloneqq \boldsymbol{y}/\sqrt{\overline{p}}$, with its error parameter ϵ set to $\frac{1}{4}\sqrt{\overline{p}}$. We will later observe that the log log trick applies, and for now assume all estimates are accurate. So given an estimate $\widehat{\mu}'$ of $\mathbf{E}_p[\boldsymbol{y}']$ that is correct to an additive $\frac{1}{4}\sqrt{\overline{p}}$, multiplying it by $\sqrt{\overline{p}}$ gives an estimate p' of p that is correct to an additive $\frac{1}{4}\overline{p}$.

If $p' \leq \frac{1}{2}\overline{p}$, the algorithm may infer that $p \leq \frac{3}{4}\overline{p}$, and therefore lower $\overline{p}$ by a factor of $\frac{3}{4}$ for the next stage. On the other hand, if $p' \geq \frac{1}{2}\overline{p}$, the algorithm may infer that $p \geq \frac{1}{4}\overline{p}$, and thus $\frac{1}{2}\overline{p}$ is within a factor 2 of p.

The algorithm proceeds in this way until either $\overline{p} \leq 1/(4n^2)$ (at which point it may safely output $\widehat{\mu} = 0$) or else it knows a factor-2 approximation $\widehat{p} \geq 1/(8n^2)$ of p. In the latter case, the algorithm uses the solution to Problem 3 one more time, on the random variable $\boldsymbol{y}/\sqrt{2\widehat{p}}$ (which has second moment at most 1, as needed), with error parameter $\epsilon = \frac{1}{2n}$. This requires $O(n)$ uses of the code for $\boldsymbol{y}$, and — multiplying the estimate by $\sqrt{2\widehat{p}}$ — yields an estimate for p that is within additive error $\sqrt{2\widehat{p}} \cdot \frac{1}{2n} \leq \sqrt{p}/n = \sigma/n$, as desired.

It remains to remark that we can use the log log trick as before to ensure high confidence in all stages succeeding; when our current bound on p is $\overline{p}$, we can define the "accuracy parameter" to be $\epsilon = \frac{1}{4}\sqrt{\overline{p}}$. Then as in the preceding proof, we can achieve this accuracy and δ confidence using $O(1/\epsilon)\log(1/\delta)$ queries. Since $\overline{p}$ decreases by a factor of $3/4$ in each stage, the accuracy parameter decreases by a factor $\sqrt{3/4} < 1$. And since $\overline{p}$ never goes below $1/(4n^2)$, our final accuracy parameter ϵ^* may be set to $1/(8n)$, meaning the total query cost will be $O(n)$, as desired. □

Next we observe that the result of [Ter99] from Table 1 follows almost immediately:

Problem 5 (Mean estimation (suboptimal) for bounded random variables)

Input: Positive integer n.
Promise: $\boldsymbol{y} \in [0,1]$ always.
Output: An estimate $\widehat{\mu}$ such that $|\widehat{\mu} - \mu| \leq \sqrt{\mu}/n$.

Lemma 4.5. *We can solve Problem 5 with $O(n)$ queries to the code for $\boldsymbol{y}$.*

Proof. We reduce from the $\{0,1\}$-valued case essentially as Terhal [Ter99]. Given the code for a random variable $\boldsymbol{y} \in [0,1]$, we can tack on a small amount of additional classical randomness, forming code for a related random variable $\boldsymbol{y}' \in \{0,1\}$ as follows: Draw $\boldsymbol{y}$, and if the outcome is y, let $\boldsymbol{y}'$ be a $\{0,1\}$-valued random variable with expectation y. In this way, $\mathbf{E}_p[\boldsymbol{y}'] = \mu$, and $\mathbf{stddev}_p[\boldsymbol{y}'] \leq \sqrt{\mathbf{E}_p[(\boldsymbol{y}')^2]} = \sqrt{\mu}$. Thus we may apply our solution to Problem 4 to $\boldsymbol{y}'$ to complete the proof. □

Finally we show how to achieve the results due to [Hei02, Mon15] from Table 1 with query complexity $O(n)$: essentially, optimal Mean Estimation in terms of a *known* upper bound on the standard deviation. Aside from a trivial scaling issue, the difference between this and our Problem 3 is that we only wish to assume a bound on σ rather than s. Since $\sigma^2 = s^2 - \mu^2$, we can only have s significantly larger than σ if μ is very large compared to σ. As noted by Montanaro [Mon15], such a situation can easily be fixed by subtracting one "typical" value of $\boldsymbol{y}$ from each subsequent draw.

Problem 6 (Mean estimation in terms of a standard deviation bound)

Input: Positive integer n, and parameter $\sigma_{\text{bound}} \geq 0$.
Promise: $\sigma \leq \sigma_{\text{bound}}$.
Output: An estimate $\widehat{\mu}$ such that $|\widehat{\mu} - \mu| \leq \sigma_{\text{bound}}/n$.

Lemma 4.6. *We can solve Problem 6 with $O(n)$ queries to the code for $\boldsymbol{y}$.*

Proof. The idea essentially appears in [Mon15]. If $\sigma_{\text{bound}} = 0$ then $\boldsymbol{y}$ is constant and one draw suffices to get μ exactly. Otherwise, by scaling we may assume that σ_{bound} is, say, $1/4$; then our target additive error is $1/(4n)$.

The algorithm first uses the code for $\boldsymbol{y}$ to draw a single sample — call the sample $\boldsymbol{m}$, and say its outcome is m. Then the algorithm forms (the code for) a new random variable $\boldsymbol{y}' = \boldsymbol{y} - m$. We have $\mu' \coloneqq \mathbf{E}_p[\boldsymbol{y}'] = \mu - m$, so it suffices to estimate μ' to an additive $1/(4n)$.

By applying Chebyshev's inequality to $\boldsymbol{m}$, we get that that $|m-\mu| \le 2\sigma$ except with probability at most $1/4$. Assuming this happens, we have

$$(4.79)\quad (s')^2 \coloneqq \mathbf{E}_p[(\boldsymbol{y}')^2] = \mathbf{E}_p[(\boldsymbol{y}-\mu+\mu-m)^2] \le 2\mathbf{E}_p[(\boldsymbol{y}-\mu)^2] + 2\mathbf{E}_p[(\mu-m)^2]$$
$$\le 2\sigma^2 + 2(2\sigma)^2 = 10\sigma^2 \le 10\sigma^2_{\text{bound}} = 10/4^2 \le 1.$$

Thus the promise of Problem 3 is satisfied for $\boldsymbol{y}'$, and by taking $\epsilon = 1/(4n)$ (and repeating our algorithm for Problem 3 a few times to get failure probability at most $1/12$), we get the necessary estimate for μ'. □

4.2 The final upgrade: handling an unknown standard deviation With our solution to Problem 6 in hand, the last remaining challenge is to avoid assuming a known upper bound on the standard deviation σ of $\boldsymbol{y}$.

To begin, we return to our solution to Problem 5 concerning $[0,1]$-valued random variables. To make it look more like our final goal, we achieve error $s/n = \sqrt{\mathbf{E}_p[\boldsymbol{y}^2]}/n$ rather than the larger $\sqrt{\mu} = \sqrt{\mathbf{E}_p[\boldsymbol{y}]}/n$. However we will have to assume that $s \ge 1/n$.

Problem 7

Input: Positive integer n.
Promise: $\boldsymbol{y} \in [-1,1]$ and $s \ge 1/n$.
Output: An estimate $\widehat{\mu}$ such that $|\widehat{\mu}-\mu| \le s/n$.

Lemma 4.7. *We can solve Problem 7 with $O(n)$ queries to the code for $\boldsymbol{y}$.*

Proof. The first step is to *estimate* s to within a factor of 2. To do this, we apply our solution to Problem 5 to the random variable $\boldsymbol{z} \coloneqq \boldsymbol{y}^2$. We have $\boldsymbol{z} \in [0,1]$ since $\boldsymbol{y} \in [-1,1]$, so with $O(n)$ queries we can get an additive estimate of $\mathbf{E}_p[\boldsymbol{z}] = \mathbf{E}_p[\boldsymbol{y}^2] = s^2$ that is correct to within an additive $\sqrt{s^2}/(2n) = s/(2n)$. But $s/(2n) \le s^2/2$ since we have the promise $s \ge 1/n$. Thus our estimate of s^2 is within an additive $s^2/2$; i.e., it is a factor-2 multiplicative estimate. So we have a factor-2 (or even $\sqrt{2}$) multiplicative estimate $\hat{s}$ of s.

Given this, we can form the rescaled random variable $\boldsymbol{y}' \coloneqq \boldsymbol{y}/(2\hat{s})$, which has $(s')^2 \coloneqq \mathbf{E}_p[(\boldsymbol{y}')^2] \le 1$. Then applying our solution to Problem 3 with error parameter $\epsilon = 1/(4n)$, we use $O(n)$ queries to get an estimate $\widehat{\mu}'$ of $\mathbf{E}_p[\boldsymbol{y}] = \mu/(2\hat{s})$ that is correct to an additive $1/4n$. Finally, taking $\widehat{\mu} = (2\hat{s}) \cdot \widehat{\mu}'$, we have an estimate of μ that is correct to an additive $(2\hat{s})/(4n) \le s/n$, as desired. □

We now come to the (almost-final) step: using the quantum Quantile Estimation algorithm of Hamoudi [Ham21]. With $O(n)$ queries, this will allow us to find a suitable "cap" value B such that replacing $\boldsymbol{y}$ with its truncation $\boldsymbol{y}'$ to the interval $[-B,B]$ does not substantially change the mean estimation task. As long as we have $B \le n \cdot \sqrt{\mathbf{E}_p[(\boldsymbol{y}')^2]}$, we will be able to employ our solution to Problem 7 (after dividing $\boldsymbol{y}'$ by B).

The correct value to choose for B is, roughly speaking, the "$(1-1/n^2)$"-quantile value for $\boldsymbol{y}$; i.e., the largest B such that $\mathbf{Pr}[\boldsymbol{y} \ge B] \ge 1/n^2$. Hamoudi's algorithm can find this B with $O(n)$ samples from $\boldsymbol{y}$. (Classically, we could find this B by taking $\Theta(n^2)$ draws from $\boldsymbol{y}$ and and outputting the maximum sample seen. The intuition for Hamoudi's algorithm is to take this and apply the square-root quantum speedup afforded by the Minimum Finding algorithm of [DH96].)

On one hand, a Chebyshev-type argument shows that if B is so large that $\mathbf{Pr}[\boldsymbol{y} \ge B] \ll 1/n^2$, then capping $\boldsymbol{y}$ at B does not affect the mean/second-moment enough to make a substantial difference to the mean estimation problem. On the other hand, this value of B will be small enough that the $B \le n \cdot \sqrt{\mathbf{E}_p[(\boldsymbol{y}')^2]}$ required for Problem 7 holds. This is because (roughly speaking) we have $\mathbf{Pr}[\boldsymbol{y} \approx B] \gtrapprox 1/n^2$ (else the quantile value B could be chosen larger), and hence even the capped $\boldsymbol{y}'$ will have $\mathbf{Pr}[\boldsymbol{y}' \approx B] \gtrapprox 1/n^2$, implying $\mathbf{E}_p[(\boldsymbol{y}')^2] \gtrapprox B^2/n^2$.

Problem 8 (Mean estimation in terms of second moment)

Input: Positive integer n.
Output: An estimate $\widehat{\mu}$ such that $|\widehat{\mu}-\mu| \le s/n$.

Lemma 4.8. *We can solve Problem 8 with $O(n)$ queries to the code for $\boldsymbol{y}$.*

Proof. We begin by performing the Quantile Estimation algorithm of Hamoudi [Ham21] on the random variable $|\boldsymbol{y}|$. This uses $O(n)$ queries and (with high probability) determines a number B (a possible outcome for $|\boldsymbol{y}|$) such that:

$$\mathbf{Pr}_p[|\boldsymbol{y}| \geq B] \geq 1/n^2, \qquad \mathbf{Pr}_p[|\boldsymbol{y}| > B] \leq C/n^2. \tag{4.80}$$

(Here C is a large universal constant, and Hamoudi's analysis also requires that n is at least some universal n_0 — but we may freely assume that.)

We now follow Hamoudi's idea and define

$$\boldsymbol{y}' = \begin{cases} \boldsymbol{y} & \text{if } |\boldsymbol{y}| \leq B, \\ +B & \text{if } \boldsymbol{y} > B, \\ -B & \text{if } \boldsymbol{y} < -B. \end{cases} \tag{4.81}$$

Our first goal is to get a good estimate for $\mu' \coloneqq \mathbf{E}_p[\boldsymbol{y}']$. If $B = 0$ then we immediately know $\mu' = 0$, since $\boldsymbol{y}' \equiv 0$. Otherwise, let us consider the random variable $\boldsymbol{y}'' \coloneqq \boldsymbol{y}'/B \in [-1, 1]$. We know

$$(s')^2 \coloneqq \mathbf{E}_p[(\boldsymbol{y}')^2] \geq B^2 \cdot \mathbf{Pr}_p[|\boldsymbol{y}'| \geq B] = B^2 \cdot \mathbf{Pr}_p[|\boldsymbol{y}| \geq B] \geq B^2/n^2, \tag{4.82}$$

where we used the first Inequality (4.80). Thus $s'' = s'/B \geq 1/n$, meaning the promises of Problem 7 are satisfied for $\boldsymbol{y}''$. Thus with $O(n)$ we can obtain an estimate for μ'' to within s''/n, hence an estimate $\widehat{\mu}$ for μ' within s'/n. Since $\boldsymbol{y}'$ is a truncation of $\boldsymbol{y}$, we clearly have $s' \leq s$; thus $|\widehat{\mu} - \mu'| \leq s/n$.

It now suffices to show the claim $|\mu - \mu'| \leq O(s/n)$; this will imply $|\widehat{\mu} - \mu| \leq O(s/n)$, and we can complete the proof of the lemma by adjusting n by a constant factor.

To show this the claim, observe that

$$|\mu - \mu'| = |\mathbf{E}_p[\boldsymbol{y} - \boldsymbol{y}']| \leq \mathbf{E}_p[|\boldsymbol{y} - \boldsymbol{y}'|] \leq \mathbf{E}_p\Big[\Big||\boldsymbol{y}| - B\Big| \cdot 1_{\{|\boldsymbol{y}|>B\}}\Big], \tag{4.83}$$

as $\boldsymbol{y}' = \boldsymbol{y}$ when $|\boldsymbol{y}| \leq B$. Now Cauchy–Schwarz implies the above is at most

$$\sqrt{\mathbf{E}_p[(|\boldsymbol{y}| - B)^2 \cdot 1_{\{|\boldsymbol{y}|>B\}}]}\sqrt{\mathbf{E}_p[1_{\{|\boldsymbol{y}|>B\}}]} \leq \sqrt{\mathbf{E}[|\boldsymbol{y}|^2]} \cdot \sqrt{\mathbf{Pr}_p[|\boldsymbol{y}| > B]} \leq \sqrt{C} \cdot s/n, \tag{4.84}$$

where we used the second Inequality (4.80). Thus we have established the claim $|\mu - \mu'| \leq O(s/n)$, completing the proof. □

Finally, we come to the main Mean Estimation problem; its only difference from Problem 8 is that it has the standard deviation σ in place of s:

Problem 9 (Mean Estimation)

Input: Positive integer n.
Output: An estimate $\widehat{\mu}$ such that $|\widehat{\mu} - \mu| \leq \sigma/n$.

Our main Theorem 1.1 is equivalent to saying that Problem 9 can be done with $O(n)$ queries. But this follows from our solution to Problem 8 via the Montanaro trick, in exactly the same way that Lemma 4.6 follows from Lemma 4.3.

References

[BBHT98] Michel Boyer, Gilles Brassard, Peter Høyer, and Alain Tapp. Tight bounds on quantum searching. *Fortschritte der Physik: Progress of Physics*, 46(4-5):493–505, 1998. doi:10.1002/(sici)1521-3978(199806)46:4/5<493::aid-prop493>3.0.co;2-p.

[Bel19] Aleksandrs Belovs. Quantum algorithms for classical probability distributions. In *Proceedings of the 27th Annual European Symposium on Algorithms (ESA)*, pages 50–59. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2019. doi:10.1007/978-3-030-19955-5_5.

[Ben73] Charles Bennett. Logical reversibility of computation. *IBM Journal of Research and Development*, 17(6):525—-532, 1973. doi:10.1147/rd.176.0525.

[BHMT02] Gilles Brassard, Peter Høyer, Michele Mosca, and Alain Tapp. Quantum amplitude amplification and estimation. In *Quantum Computation and Information*, volume 305 of *Contemporary Mathematics*, pages 53–74. American Mathematical Society, 2002. doi:10.1090/conm/305/05215.

[BHT98] Gilles Brassard, Peter Høyer, and Alain Tapp. Quantum counting. In *Proceedings of the 25th Annual International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 820–831. Springer–Verlag, 1998. doi:10.1007/bfb0055105.

[BS73] Fischer Black and Myron Scholes. The pricing of options and corporate liabilities. *Journal of Political Economy*, 81(3):637–654, 1973. URL: http://www.jstor.org/stable/1831029.

[BvDJ+20] Adam Bouland, Wim van Dam, Hamed Joorati, Iordanis Kerenidis, and Anupam Prakash. Prospects and challenges of quantum finance. Technical Report 2011.06492, arXiv, 2020. doi:10.48550/arXiv.2011.06492.

[BZ11] Richard Brent and Paul Zimmermann. *Modern Computer Arithmetic*. Cambridge University Press, 2011. doi:10.1017/CBO9780511921698.

[CEMM98] Richard Cleve, Artur Ekert, Chiara Macchiavello, and Michele Mosca. Quantum algorithms revisited. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 454(1969):339–354, jan 1998. doi:10.1098/rspa.1998.0164.

[CKM+21] Shouvanik Chakrabarti, Rajiv Krishnakumar, Guglielmo Mazzola, Nikitas Stamatopoulos, Stefan Woerner, and William Zeng. A threshold for quantum advantage in derivative pricing. *Quantum*, 5:463, 2021. doi:10.22331/q-2021-06-01-463.

[DH96] Christoph Dürr and Peter Høyer. A quantum algorithm for finding the minimum. Technical Report quant-ph/9607014, arXiv, 1996. doi:10.48550/arXiv.quant-ph/9607014.

[DKLR00] Paul Dagum, Richard Karp, Michael Luby, and Sheldon Ross. An optimal algorithm for Monte Carlo estimation. *SIAM Journal on computing*, 29(5):1484–1496, 2000. doi:10.1137/s0097539797315306.

[Gro96] Lov Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the 28th Annual ACM Symposium on the Theory of Computing (STOC)*, pages 212–219. ACM, New York, 1996. doi:10.1145/237814.237866.

[Gro98] Lov Grover. A framework for fast quantum mechanical algorithms. In *Proceedings of the 30th Annual ACM Symposium on the Theory of Computing (STOC)*, pages 53–62. ACM, New York, 1998. doi:10.1145/276698.276712.

[Gro05] Lov K. Grover. Fixed-point quantum search. *Phys. Rev. Lett.*, 95:150501, Oct 2005. doi:10.1103/PhysRevLett.95.150501.

[Ham21] Yassine Hamoudi. *Quantum Algorithms for the Monte Carlo Method*. PhD thesis, Université de Paris, 2021.

[Hei02] Stefan Heinrich. Quantum summation with an application to integration. *Journal of Complexity*, 18(1):1–50, 2002. doi:10.1006/jcom.2001.0629.

[Hel76] Carl Helstrom. *Quantum Detection and Estimation Theory*. Academic Press, 1976.

[HGL+22] Dylan Herman, Cody Googin, Xiaoyuan Liu, Alexey Galda, Ilya Safro, Yue Sun, Marco Pistoia, and Yuri Alexeev. A survey of quantum computing for finance. Technical Report 2201.02773, arXiv, 2022. doi:10.48550/arXiv.2201.02773.

[HH64] John Hammersley and David Handscomb. *Monte Carlo methods*. Chapman and Hall Ltd, 1964.

[HvdH21] David Harvey and Joris van der Hoeven. Integer multiplication in time $O(n \log n)$. *Annals of Mathematics. Second Series*, 193(2):563–617, 2021. doi:10.4007/annals.2021.193.2.4.

[Ion01] Eugen Ionascu. Rank-one perturbations of diagonal operators. *Integral Equations and Operator Theory*, 39(4):421–440, 2001. doi:10.1007/BF01203323.

[Kit95] Alexei Kitaev. Quantum measurements and the Abelian stabilizer problem. Technical Report quant-ph/9511026, arXiv, 1995. doi:10.48550/arXiv.quant-ph/9511026.

[Lue14] David Luenberger. *Investment Science*. Oxford University Press, 2014.

[Mer73] Robert Merton. Theory of rational option pricing. *Bell Journal of Economics and Management Science*, 4(1):141–183, 1973. doi:10.2307/3003143.

[Mon15] Ashley Montanaro. Quantum speedup of Monte Carlo methods. *Proceedings of the Royal Society A*, 471(2181):20150301, 20, 2015. doi:10.1098/rspa.2015.0301.

[Nay99] Ashwin Nayak. *Lower Bounds for Quantum Computation and Communication*. PhD thesis, University of California, Berkeley, 1999.

[NC10] Michael Nielsen and Isaac Chuang. Quantum computation and quantum information, 2010.

[OML19] Román Orús, Samuel Mugel, and Enrique Lizaso. Quantum computing for finance: Overview and prospects. *Reviews in Physics*, 4:100028, 2019. doi:10.1016/j.revip.2019.100028.

[RGB18] Patrick Rebentrost, Brajesh Gupt, and Thomas Bromley. Quantum computational finance: Monte Carlo pricing of financial derivatives. *Physical Review A*, 98:022321, 2018. doi:10.1103/PhysRevA.98.022321.

[SES+20] Nikitas Stamatopoulos, Daniel Egger, Yue Sun, Christa Zoufal, Raban Iten, Ning Shen, and Stefan Woerner.

Option pricing using quantum computers. *Quantum*, 4:291, 2020. doi:10.22331/q-2020-07-06-291.

[Ter99] Barbara Terhal. *Quantum algorithms and quantum entanglement*. PhD thesis, University of Amsterdam, 1999.

[WE19] Stefan Woerner and Daniel Egger. Quantum risk analysis. *npj Quantum Information*, 5(1), feb 2019. doi:10.1038/s41534-019-0130-6.

[WSK+21] Daochen Wang, Aarthi Sundaram, Robin Kothari, Ashish Kapoor, and Martin Roetteler. Quantum algorithms for reinforcement learning with a generative model. In *Proceedings of the 38th Annual International Conference on Machine Learning (ICML)*, pages 10916–10926. PMLR, 2021.

A Gate complexity

In this section we sketch how to establish Remark 1.2, that our algorithm's gate complexity is (essentially) minimal given its sample complexity: namely, $O(nS)$, where S is the gate complexity of "the code" for $\boldsymbol{y}$. The only potential excess comes from having to classically compute the arctan function. More precisely, we show the gate complexity is

$$O(nS) + O(n \log n \cdot (\log\log n)^2). \tag{A.1}$$

Remark A.1. The extra additive term $O(n \log n \cdot (\log\log n)^2)$ above can be absorbed into the $O(nS)$ except when $S < o(\log n \cdot (\log\log n)^2)$. On the other hand, if $S < o(\log n)$, with gate complexity $n^{o(1)}$ we can compute $\mathbf{E}_p[\boldsymbol{y}]$ *exactly*[14] by brute-force analysis of all computational paths in the circuit for $\boldsymbol{y}$. Thus only in the unusual case of $\Omega(\log n) \le S < o(\log n \cdot (\log\log n)^2)$ must we report our algorithm's gate complexity as $o(nS \cdot (\log\log n)^2)$, rather than $O(nS)$.

Assumptions. We use the standard model of CNOT gates together with any 1-qubit gate. (From these one can also build Toffoli gates [NC10, Fig. 4.9].) We will assume that the code for $\boldsymbol{y}$ outputs its value in a "floating point" format (of at most S bits). Hence given output values of the code, we can perform the following with gate complexity $O(S)$: subtraction, comparison with 0, rounding to a power of 2, and multiplication/division by a power of 2 (shifting).

A.1 Summary of the steps of the algorithm

Here we summarize the algorithms needed for Mean Estimation with error σ/n.

Solving Problem 3. (I.e., mean estimation for random variables $\boldsymbol{y}$ satisfying $\mathbf{E}_p[\boldsymbol{y}^2] \le 1$.) This algorithm will always be run with precision parameter $\epsilon \ge \Omega(1/n)$:

- Binary search for μ with intervals of of width decreasing geometrically from 2 to $\Omega(\epsilon)$.
- Test each interval centered at $\widehat{\mu}$ by replacing $\boldsymbol{y}$ with $\boldsymbol{y} - \widehat{\mu}$ and performing our solution to the Main Task, namely:
  - Converting the code for $\boldsymbol{y}$ to (controlled versions of) REFL_p and ROT_y.
  - Performing Quantum Phase Estimation on $\mathcal{U} = \mathrm{REFL}_p \cdot \mathrm{ROT}_y$.

The overall Mean Estimation algorithm. This is obtained by reading Section 4 roughly backward:

- Draw one sample m from $\boldsymbol{y}$ and replace $\boldsymbol{y}$ with $\boldsymbol{y} - m$. (This is to go from Problem 9 to Problem 8.)
- Perform Hamoudi's Quantile Estimation on $|\boldsymbol{y}|$ obtaining B; replace $\boldsymbol{y}$ with its truncation to $[-B, B]$ and divide it by B. Call the resulting random variable $\overline{\boldsymbol{y}}$.
- Estimate $\mathbf{E}[\boldsymbol{z}]$ for $\boldsymbol{z} = \overline{\boldsymbol{y}}^2$ to factor 2. This uses the solution to Problem 5, as follows:
  - Replace the $[0,1]$-valued $\boldsymbol{z}$ with a randomized $\{0,1\}$-valued version.
  - Starting with a trivial upper bound $\overline{p}$ for $p = \mathbf{Pr}[\boldsymbol{z} = 1]$, repeatedly use the solution to Problem 3 to estimate $\overline{p}$ to within $\epsilon \approx \frac{1}{4}\sqrt{\overline{p}}$, and lower $\overline{p}$ if necessary (but not below $1/(4n^2)$)
- Having determined $\mathbf{E}_p[\overline{\boldsymbol{y}}^2]$ to factor 2, rescale it so that $\mathbf{E}[\overline{\boldsymbol{y}}^2] \approx 1$; then use the solution to Problem 3 on $\overline{\boldsymbol{y}}$.

[14] Exactly, if the gates used to compute $\boldsymbol{y}$ have amplitudes that are exactly representable. Otherwise, up to $O(\log n)$ bits of precision — which suffices, as we will explain.

A.2 Precision issues Suppose we have done the first two steps of the overall Mean Estimation algorithm, obtaining m and B; these numbers are expressed in floating point with at most S bits. All subsequent stages of the algorithm work with the $[-1,1]$-bounded random variable $\overline{\boldsymbol{y}} = \frac{1}{B} \cdot \mathrm{trunc}_{[-B,B]}(\boldsymbol{y} - m)$. In gate complexity $O(S)$ it is easy to compute $\mathrm{trunc}_{[-B,B]}(\boldsymbol{y} - m)$. Dividing by B is not as easy, but we argue that it is fine if the algorithm simply divides by the next largest power of 2, call it B'. The only properties we needed in from this scaling in Problem 7 were that the resulting random variable has $\overline{\boldsymbol{y}} \in [-1,1]$ and $\sqrt{\mathbf{E}[\overline{\boldsymbol{y}}^2]} \geq 1/n$. If we divide by a slightly larger B', the first property still holds; and while the second is no longer literally true, it is true up to a factor of 4, we we can easily compensate for by adjusting the constant factor on n. Thus once we are dividing by B' — a power of 2 — (and also subsequently multiplying our final estimate by B'), the gate complexity becomes $O(S)$ as this just amounts to bit-shifting.

Thus we have argued so far that we can still get one sample from $\overline{\boldsymbol{y}} \in [-1,1]$ with gate complexity $O(S)$. We now wish to argue that samples and computations done with $\overline{\boldsymbol{y}}$ can be rounded to $O(\log n)$ bits of precision. (Recall also that $S \geq \Omega(\log n)$ without loss of generality.) To see this, it suffices to note that all the remaining steps of the algorithm only care about numbers and interval widths that are at least $\Omega(1/n^2)$; hence round-off to a sufficiently large $O(\log n)$ bits will affect their accuracy/correctness by at most small constant factors. As usual, we can ultimately compensate for these small constant factors by increasing our sample complexity by a constant factor.

Precision conclusion: For the post-Hamoudi part of our algorithm, we can assume the gate complexity of obtaining a sample is still $O(S)$, and then that all subsequent numbers and computations require bit complexity only $O(\log n)$.

A.3 Final gate analysis We now analyze the steps of the algorithm to justify our claim that the overall gate complexity is $O(nS)$ plus $O(n) \cdot \widetilde{O}(\log n)$.

The first step of sampling and subtracting m does not cost more than $O(S)$ gates per sample, so the first serious piece of the algorithm to analyze is Hamoudi's Quantile Estimation algorithm.

Hamoudi's algorithm. Hamoudi does not explicitly analyze the gate complexity of his algorithm; he just shows the sample complexity is $O(n)$. We argue that the gate complexity is $O(nS) + O(n) \cdot \widetilde{O}(\log n)$. To do this, we first sketch his algorithm as applied to a random variable $\boldsymbol{x}$. In short, the algorithm produces a sequence $-\infty = x_0 < x_1 < x_2 < \cdots$ by repeatedly setting y_{t+1} to be a draw from $\boldsymbol{x} \mid (\boldsymbol{x} > x_t)$. Obtaining each draw is done via a "Sequential Amplitude Estimation" quantum algorithm, as in [BBHT98] (similar to our Lemma 4.4). In turn, the pseudocode for this is roughly the following:

1. For $j = 1, 2, 3, \ldots$
2. Let T_j be a random integer in $[(6/5)^{j-1}, (6/5)^j]$.
3. Do Amplitude Amplification [BHMT02] with $O(T_j)$ applications of the "code for $\boldsymbol{x}$". More precisely, the synthesizer for $\boldsymbol{x}$ and its inverse are used $O(T_j)$ times, as is $1 - 2\Pi$, where Π is a projector depending on the code $\mathcal{X}$ for $\boldsymbol{x}$ and comparison with the current x_t. (This comparison takes only $O(S)$ gates.)
4. Measure with $(\Pi, \mathbb{1} - \Pi)$ and quit the "for loop" if Π occurs.
5. Do a measurement to obtain the next x_t.

Moreover, throughout this pseudocode the algorithm maintains a counter of the number of times the code for $\boldsymbol{x}$ has been applied, and it halts (with the final value of x_t as its output) once the budget of $O(n)$ samples has been hit.

As mentioned, we claim that this algorithm can be implemented with a circuit of size $O(nS) + O(n) \cdot \widetilde{O}(\log n)$. Essentially, we can unroll all loops and make one sequence of $O(n)$ applications of (pieces of) the circuit for $\boldsymbol{x}$. Interspersed between of applications of the circuit for $\boldsymbol{x}$ will be conditionals and operations operating on T_j (which is $O(\log n)$ bits), comparisons of $O(S)$-bit quantities, and conditional measurements (including measuring qubits to get classical random bits). These measurements can be moved to the end by the usual Principle of Deferred Measurements.

The last output x_t is the value of "B". As discussed before, with B in hand, we can assume that all future calculations done on draws from $\overline{\boldsymbol{y}}$ are done with $O(\log n)$ bits of precision.

Estimating $\mathbf{E}[\boldsymbol{z}] = \mathbf{E}[\overline{\boldsymbol{y}}^2]$ to factor 2. The next step of the algorithm, estimating the mean of $\overline{\boldsymbol{y}}^2$ to a factor of 2, first involves replacing $\overline{\boldsymbol{y}}$ by $\boldsymbol{z} = \overline{\boldsymbol{y}}^2$. In turn this means we have to multiply an $O(\log n)$-bit sample by itself, creating an overhead of $M(\log n) = O(\log n \log\log n)$ gates per sample [HvdH21].

Next, in Lemma 4.5, we must implement the additional randomness for converting a $[0,1]$-valued random variable to a $\{0,1\}$-valued one. This involves taking a sample outcome z (expressed with $O(\log n)$ bits) and producing $\sqrt{1-z}\,|0\rangle + \sqrt{z}\,|1\rangle$. In turn, this involves $O(\log n)$ controlled rotations, plus the computation of $\arctan\sqrt{z/(1-z)}$ (to $O(\log n)$ bits of precision). The gate complexity of this computation is $O(M(\log n)\cdot \log\log n) = O(\log n \cdot (\log\log n)^2)$ [BZ11, HvdH21]. Thus now one "sample" costs gate complexity $O(S + \log n \cdot (\log\log n)^2)$; this (along with a subsequent arctan computation) is the computational "bottelneck" leading to our final gate complexity of n times the cost of a sample.

To complete the estimate of $\mathbf{E}[\boldsymbol{z}]$, we need to do the perform the "successive halving" routine of Lemma 4.4. This mainly uses our solution to Problem 3, which we analyze below. The only other aspect is the log log trick — for which the appropriate δ_j values can be precalculated — and minor computations on $O(\log n)$ bit numbers (which are within our budget).

It remains to analyze the gate complexity of our solution to Problem 3.

The binary search. In the binary search, it is easy to adjust constants so that the intervals I_j decrease in width by precisely some factor $1 - c'$ at each stage. Thus the whole binary search can be straightforwardly unrolled with the interval widths and the δ_j's precalculated. The only additional work that needs to be done is counting and comparing $O(\log n)$-bit integers (to compute majorities). Again, this portion of the algorithm only incurs an additive overhead of $O(n)\cdot\widetilde{O}(\log n)$.

Finally, we reach:

The Main Task. Finally we come to our algorithm for the Main Task from Section 3. At this point, each of our samples costs $O(S)+\widetilde{O}(\log n)$ gates, we need to do all calculations with precision $O(\log n)$ bits, and the sample complexity is $O(1/\epsilon) = O(n)$. The gate complexity of phase estimation is the cost to prepare the initial state ($O(S)$ for us), plus the sample complexity times the cost of controlled-$\mathcal{U}$, plus the cost of some minor calculations on numbers of $O(\log(1/\epsilon)) = O(n)$ bits. Since our final claimed gate complexity is $O(nS)+O(n\log n\cdot(\log\log n)^2)$, it now remains to argue that the cost to compute controlled-$\mathcal{U}$ is $O(S) + O(M(\log n)\log\log n)$.

The cost to compute controlled-$\mathcal{U}$ is the cost to compute controlled-REFL_p and controlled-ROT_y. The former has gate complexity $O(S)$. As for the latter, we need to take a sample y, compute $\alpha = -2\arctan y$ (to $O(\log n)$ bits of precision), and apply $O(\log n)$ controlled-phase gates to implement the phase $e^{\mathrm{i}\alpha}$. Similarly to before, the main bottleneck is computing the arctan, and the gate complexity is $O(M(\log n)\cdot\log\log n) = \widetilde{O}(\log n)$ [BZ11, HvdH21].