Amplitude Estimation from Quantum Signal Processing

Patrick Rall[1] and Bryce Fuller[2]

[1]IBM Quantum, MIT-IBM Watson AI Lab, Cambridge, Massachusetts 02142, USA
[2]IBM Quantum, Thomas J Watson Research Center, Yorktown Heights, New York 10598, USA

Amplitude estimation algorithms are based on Grover's algorithm: alternating reflections about the input state and the desired outcome. But what if we are given the ability to perform arbitrary rotations, instead of just reflections? In this situation, we find that quantum signal processing lets us estimate the amplitude in a more flexible way. We leverage this technique to give improved and simplified algorithms for many amplitude estimation tasks: we perform non-destructive estimation without any assumptions on the amplitude, develop an algorithm with improved performance in practice, present a new method for unbiased amplitude estimation, and finally give a simpler method for trading quantum circuit depth for more repetitions of short circuits.

Amplitude estimation is a fundamental subroutine in quantum algorithms. It directly yields quantum speedups that improve the accuracy of Monte Carlo estimation [Montanaro15], which has applications in many areas of science and technology. But it also forms a fundamental component of more complicated quantum algorithms like those presented in [HW19, AHNTW20]. A general description of the problem is as follows: given a state $|\psi\rangle$ and a projector Π, estimate $a := |\Pi|\psi\rangle|$. There exist algorithms that can obtain an estimate $\hat{a}$ satisfying $|a - \hat{a}| \leq \varepsilon$ with high probability in $O(\varepsilon^{-1})$ queries to oracles for $|\psi\rangle$ and Π.

The first algorithm for this task was given by [BHMT00], and it is based on applying phase estimation to the Grover operator which consists of the reflections $Z_\psi Z_\Pi$. This method has since become widely used in the literature. But, starting in 2019, there have been several efforts to improve the simplicity, constant-factor performance, and versatility of amplitude estimation.

One line of work studies algorithms that do not require the phase estimation step from the [BHMT00] method. These algorithms simply perform Grover's algorithm: they apply alternating reflections Z_Π and Z_ψ to a copy of $|\psi\rangle$, and then measure the Π observable. Via repeated measurements we can extract information about a. [Suzuki&19] gave an algorithm based on maximum-likelihood estimation that has improved constant-factor performance over [BHMT00] and is non-adaptive, but there is no proof that the algorithm is correct. [AR19] gave an algorithm for relative-error amplitude estimation with a proof of correctness, although the constant-factor performance is poor. [GGZW19] presented an additive-error estimation algorithm which is provably correct, and empirically outperforms [BHMT00] and roughly

Patrick Rall: patrickjrall@ibm.com
Bryce Fuller: Bryce.Fuller@ibm.com

matches the performance of [Suzuki&19]. [VO20] gave a relative-error estimation algorithm that is also non-adaptive. Finally, [GKLPZ20] present a family of estimation algorithms with trade-offs between the circuit depth and the total number of samples. Together, these techniques make amplitude estimation simpler, faster, and more versatile.

A separate line of work studies quantum algorithms for approximating partition functions that leverage amplitude estimation as a subroutine. In an effort to improve an algorithm from [Montanaro15], [HW19] modify the [BHMT00] method in order to be 'non-destructive'. In this situation, $|\psi\rangle$ is much harder to prepare than to reflect about. Thus, we only want to use one copy of $|\psi\rangle$ for the estimation algorithm, and return it undamaged when done. The non-destructive method by [HW19] was later used again in [AHNTW20].

Recent literature has presented many different quantum algorithms, all of which improve over [BHMT00] in some particular way. In this manuscript we build a framework that unifies the underlying quantum subroutine of some of these methods. This allows us to inherit several of these improvements simultaneously, whenever sensible. This 'unification' of amplitude estimation algorithms is achieved through Quantum Signal Processing (QSP).

QSP is a powerful technique for building quantum algorithms, capable of unifying many other existing results [GSLW18, MRTC21]. While existing techniques only consider reflections Z_Π and Z_ψ around Π and $|\psi\rangle$, we consider arbitrary rotations $e^{i\phi Z_\Pi}$ and $e^{i\phi Z_\psi}$. We find that in situations where Grover's algorithm is implemented, it is generally possible to implement these arbitrary rotations at no additional cost. The dynamics of such quantum circuits are well studied [YLC14, LYC16, GSLW18, MRTC21]: we find that careful choice of phases ϕ lets us implement a family of polynomials $P(a)$ in the amplitude. This yields our first result:

Theorem. *(Section 1, Theorem 6)* ***Sampling from polynomials.*** *Say we are given access to one copy of $|\psi\rangle$ as well as the oracles $e^{i\phi Z_\psi}$ and $e^{i\phi Z_\Pi}$. Then it is possible to repeatedly toss coins with $Pr[heads] = |P(a)|^2$ where $P(a)$ can be freely chosen from a certain family of polynomials. The number of oracle queries per coin toss is $O(deg(P))$.*

We find that all the algorithms mentioned above *except* for those presented in [BHMT00, HW19] can be formulated as repeated invocations of special cases of this theorem. This already brings many previous results into a single framework. By considering the full range of polynomials, we find this tool for amplitude estimation can improve or simplify many of the existing algorithms.

In Section 2 we consider the setting where the state $|\psi\rangle$ is expensive to prepare, but the reflection Z_ψ is relatively cheap. This appears in algorithms for estimating observables on ground states such as [LT20], where the manner by which $|\psi\rangle$ is prepared in the first place is by constructing a reflection about the ground space Z_ψ and then employing amplitude amplification. It also appears in the partition function estimation algorithms in [Montanaro15, HW19, AHNTW20], where a sequence of qsamples of thermal states is constructed through amplitude amplification. Here, we can reflect about any of these states using Szegedy walk unitaries, but re-preparing the states corresponding to low temperatures is expensive. To aid in settings like these, [HW19] gave a method for non-destructive amplitude estimation that restores a copy of $|\psi\rangle$ after the algorithm completes. That way, each qsample state can be re-used.

However, we find that the non-destructive estimation algorithm presented in [HW19] makes a subtle implicit assumption: that some κ is known such that $\kappa < a < \sqrt{1-\kappa^2}$. But since the whole point of amplitude estimation is to determine the value of a, such a

bound may not be known beforehand. We resolve this issue by developing a procedure that requires no such bound. Furthermore, our procedure can be combined with any algorithm in the polynomial sampling framework above. However, in the process we incur a polynomial slowdown in the failure probability δ, which only appears polylogarithmically in the overall complexity when a bound κ is known.

Theorem. *(Section 2, Theorem 15)* ***State repair.*** *Say we just sampled from several polynomials, and the sum of the degrees of these polynomials is D. Then we can restore the quantum state to a copy of $|\psi\rangle$ with success probability $\geq 1-\delta$ by sampling from one additional polynomial with degree $O(\delta^{-1/2}D)$.*

We remark that there was also a non-destructive amplitude estimation algorithm presented in [Rall21], but this algorithm estimates the probability a^2, not a, which is weaker[1]. The algorithms in [HW19, AHNTW20] rely on methods for mean estimation from [Montanaro15] which in turn rely on estimates of $\arcsin(a) \sim a$, and it is not clear how to generalize them to rely on a^2 instead.

Parallel work. [CH22] simultaneously presented a different method for achieving non-destructive amplitude estimation algorithm that also improves over that of [HW19].

Next, in Section 3 we ask if the polynomial sampling framework can reduce the constant factors in the number of queries required. Since we want amplitude estimation to be useful in practice, it is essential to study which method can minimize the number of queries, even though the asymptotic performance is already determined to be $\Theta(\varepsilon^{-1})$ [NW98]. The prior state of the art for amplitude estimation with high empirical performance is Iterative Quantum Amplitude Estimation (IQAE), the algorithm presented by [GGZW19], which, according to our experiments, requires about $\approx 9.93/\varepsilon$ queries to the Z_Π oracle to achieve a success probability of $\geq 95\%$.

IQAE relies on the oracles Z_Π and Z_ψ, which can be viewed as special cases of the $e^{i\phi Z_\Pi}$ and $e^{i\phi Z_\psi}$ oracles when $\phi = \pi/2$. In this case, the polynomial $P(a)$ is a Chebyshev polynomial. While in this sense IQAE is restricted to odd Chebyshev polynomials, the polynomial sampling framework admits Chebyshev polynomials of both even and odd degree. But except for this modest increase in flexibility, we find that it is not particularly useful to deviate from $\phi = \pi/2$ in order to improve constant-factor performance. This makes intuitive sense: Chebyshev polynomials exhibit the maximum rate of change among polynomials bounded by $|P(a)| \leq 1$ for $a \in [0,1]$, and we even rely on this fact in Section 2. Thus, as a rule of thumb, deviating from Chebyshev polynomials should waste resources.

However, we still find that it is possible to improve over the performance of IQAE. IQAE features a 'no-overshooting' condition that prevents the algorithm from taking too many samples from the most expensive polynomials near the end of execution. We find that this no-overshooting condition can be tuned in order to significantly improve performance. We combine this optimization with the ability to estimate the amplitude a directly, rather than the Grover angle $\theta := \arcsin(a)$, and the ability to sample from both odd and even Chebyshev polynomials, into a new algorithm we call 'ChebAE'.

[1] When converting an additive-error estimate of a^2 to an additive-error estimate of a, the error is blown up by roughly a factor of $1/a$. This can be expensive when a is small. Converting between a and the Grover angle $\theta := \arcsin(a)$ is not so expensive.

Empirical Claim. *(Section 3, Empirical Claim 18)* ***Chebyshev amplitude estimation 'ChebAE'.*** *There exists an amplitude estimation algorithm that samples from Chebyshev polynomials and returns an estimate of the amplitude with error ε and success probability $\geq 95\%$ in about $\approx 4.66/\varepsilon$ queries to Z_Π.*

Depending on the details of the analysis, ChebAE only requires about 45% to 65% of the queries of IQAE depending on the analysis methodology. To support this claim, we performed a comprehensive numerical study where we benchmark the query complexity of the relevant algorithms in the literature.

In Section 4, we study a problem inspired by recent work [LdW21, vACGN22]: unbiased amplitude amplitude estimation. An amplitude estimation algorithm effectively samples from a random variable $\hat{\mathbf{a}}$ that satisfies $|\hat{\mathbf{a}} - a| \leq \varepsilon$ with high probability. For unbiased estimation, we additionally demand that $\mathbb{E}[\hat{\mathbf{a}}] \approx a$.

This is in contrast to the unbiased *phase* estimation problem considered by [LdW21, vACGN22], which, given an oracle for $e^{i\theta}$, samples from an unbiased estimator $\hat{\boldsymbol{\theta}}$ of θ. Their solution to the problem is based on applying a random shift to the angle θ before estimation, and accounting for it afterward. It is not possible to use [BHMT00] to convert unbiased phase estimation to unbiased amplitude estimation since the relationship $a = \sin\theta$ is not linear. Instead, we invoke Jackson's theorem [Rivlin69] to build a polynomial that guarantees $\mathbb{E}[\hat{\mathbf{a}}] \approx a$ directly.

Theorem. *(Section 4, Theorem 23)* ***Unbiased amplitude estimation.*** *For any $\varepsilon, \eta, \delta$, there exists an amplitude estimation algorithm that samples from a random variable $\hat{\mathbf{a}}$ that satisfies $Pr[|\hat{\mathbf{a}} - a| \geq \varepsilon] \leq \delta$ and $|\mathbb{E}[\hat{\mathbf{a}}] - a| \leq \varepsilon\eta + \delta$. It has a query complexity of $O(\varepsilon^{-1}(\eta^{-1} + \log(\delta^{-1})))$.*

Parallel work. While we were unaware of any applications of this subroutine in other quantum algorithms as we were writing this manuscript, [CH22] were able to improve partition function estimation using an unbiased amplitude estimation algorithm of their own.

Finally, in Section 5 we revisit a problem studied by [GKLPZ20]: trading classical and quantum resources in amplitude estimation. Early quantum computers may be able to perform the oracles $e^{i\phi Z_\Pi}$ and $e^{i\phi Z_\psi}$, but may not be able to perform them $O(\varepsilon^{-1})$ many times within a single circuit as required by most amplitude estimation algorithms. [GKLPZ20] develop two algorithms that, for any $\beta \in [0, 1)$, have query complexity $O(\varepsilon^{-(1+\beta)})$ and query depth $O(\varepsilon^{-(1-\beta)})$. When $\beta = 0$, this corresponds to the traditional scaling of amplitude estimation, but as $\beta \to 1$ we approach the complexity of classical probability estimation.

The algorithms presented in [GKLPZ20] are rather complicated: one algorithm relies on maximum likelihood estimation, which functions very well in practice but rigorously proving its accuracy and convergence rate is challenging. The other algorithm pieces together several estimates of multiples of a via the Chinese remainder theorem. We find that there exists a simpler procedure based on careful construction of polynomials. The polynomials approximate a line with slope k near the current best estimate of a, and have degree $O(k)$. By selecting $k \in O(\varepsilon^{-(1-\beta)})$ and we can achieve depth $\varepsilon^{-(1-\beta)}$ as desired.

The main technical challenge in constructing our polynomial-based method is rigorously proving bounds on the quality of the polynomial approximation. We find that Jackson's theorem is not accurate enough, so we must instead rely on a construction based on the Jacobi-Anger expansion (see [LC17]). However, we find that we can analyze our method

with greater rigor than that of [GKLPZ20], since we do not need to implicitly assume that $a, \beta \in \Theta(1)$ as they do.

Theorem (Section 5, Theorem 31)**.** ***Hybrid quantum-classical amplitude estimation.*** *For any $\varepsilon, \delta > 0$ and $0 \leq \beta < 1$, there exists an amplitude estimation algorithm that estimates a to accuracy ε with probability $\geq 1-\delta$. It makes at most $\tilde{O}((a^{-1}+\varepsilon^{-(1+\beta)})\log(\delta^{-1}))$ queries, and the circuits have depth at most $O(a^{-1/(1-\beta)}+\varepsilon^{-(1-\beta)})$*

Our analysis reveals an additional $O(a^{-1})$ dependence, which we suspect is also present for the algorithms presented in [GKLPZ20]. Of course, if we treat a and β as constants then we recover the same complexity.

1 Sampling from Polynomials

We begin by establishing our general framework for amplitude estimation algorithms. All future sections will be based on the main result of this section, which, informally, is as follows: Say $P(a)$ is a 'Pellian' or a 'semi-Pellian' polynomial in a. Then, there exists a quantum algorithm that samples from a random variable over $\{0,1\}$ with bias $|P(a)|^2$. This algorithm makes $O(\deg(P))$ oracle queries to $|\psi\rangle$ and Π.

The definition of 'Pellian' and 'semi-Pellian', the exact nature of the oracles, as well as the input and output states of this algorithm will be specified formally in this section. In particular, the outline for the rest of the section is as follows. First, we establish some general notation and discuss the implementation of our oracles. Second, we define the restricted family of 'Pellian' polynomials, and give the simpler algorithm for sampling from them. Third, we define the more general family of 'semi-Pellian' polynomials and give a more complicated algorithm for sampling from them. Finally, we put all these results together with some standard notation and show how some existing results fit into this framework.

1.1 Notation and Oracles

We begin by establishing some notation that we borrow from the discussion of Grover's algorithm. Our amplitude estimation framework is very closely related to Grover's algorithm. We are given a state $|\psi\rangle$ and a projector Π and want to estimate $a = |\Pi|\psi\rangle|$. Let $\bar{a} := \sqrt{1-a^2}$, and assume for a moment that $a \notin \{0,1\}$. Then let:

$$|\Pi\rangle := a^{-1} \cdot \Pi|\psi\rangle \tag{1}$$

$$|\Pi^{\perp}\rangle := \bar{a}^{-1} \cdot (I-\Pi)|\psi\rangle \tag{2}$$

Observe that $\bar{a} = |(I-\Pi)|\psi\rangle|$. We see how $|\Pi\rangle$ and $|\Pi^{\perp}\rangle$ satisfy:

$$|\psi\rangle = a|\Pi\rangle + \bar{a}|\Pi^{\perp}\rangle \tag{3}$$

We furthermore define:

$$|\psi^{\perp}\rangle := \bar{a}|\Pi\rangle - a|\Pi^{\perp}\rangle \tag{4}$$

Finally, observe that $\langle\Pi|\Pi^{\perp}\rangle = \langle\psi|\psi^{\perp}\rangle = 0$. We have constructed two orthonormal bases for the two-dimensional Grover subspace: $|\psi\rangle, |\psi^{\perp}\rangle$ and $|\Pi\rangle, |\Pi^{\perp}\rangle$.

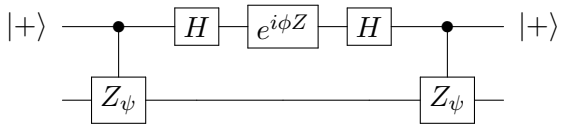
When $a \in \{0, 1\}$ then all the dynamics in the Grover subspace are trivial anyway. Nonetheless we can treat these cases along with $a \notin \{0, 1\}$ in a unified manner. If $a = 0$, let $|\Pi^\perp\rangle := |\psi\rangle$ and select an arbitrary orthogonal state $|\Pi\rangle = |\psi^\perp\rangle$. Similarly, if $a = 1$, let $|\Pi\rangle := |\psi\rangle$ and let $|\Pi^\perp\rangle = |\psi^\perp\rangle$ be an arbitrary orthogonal state.

In the traditional presentation of Grover's algorithm we are given access to reflection oracles $2|\psi\rangle\langle\psi| - I$ and $2\Pi - I$. Projected down into the two-dimensional subspace, we can write these as:

$$Z_\psi := |\psi\rangle\langle\psi| - |\psi^\perp\rangle\langle\psi^\perp| \tag{5}$$

$$Z_\Pi := |\Pi\rangle\langle\Pi| - |\Pi^\perp\rangle\langle\Pi^\perp| \tag{6}$$

In this work we consider what is possible when we are instead given access to a larger family of oracles $e^{i\phi Z_\psi}$ and $e^{i\phi Z_\Pi}$ for arbitrary phases ϕ. These can always be implemented using two queries to controlled-Z_ψ or controlled-Z_Π, by using a Hadamard test. For example, to implement $e^{i\phi Z_\psi}$:



(7)

This lets us implement $e^{i\phi Z_\psi}$ using two queries to controlled-Z_ψ. However, we argue that in essentially all practical situations the oracles $e^{i\phi Z_\psi}, e^{i\phi Z_\Pi}$ do not have twice the complexity of Z_ψ, Z_Π. Here are some examples:

- Say we have a function $f : \{0, 1\}^n \to \{0, 1\}$ and some classical program that computes it. In the usual situation for Grover's search, we apply reflections Z_Π where $\Pi = \sum_x f(x)|x\rangle\langle x|$ by rendering the classical program into a quantum circuit via Toffoli gates. That circuit produces an output bit $|f(x)\rangle$ plus garbage. To build Z_Π, we apply Z to the output bit, and then uncompute. But we could also just apply $e^{i\phi Z}$ to the output bit instead of Z, and obtain $e^{i\phi Z_\Pi}$ with the same number of gates.

- Say we have a preparation unitary U_ψ for an n-qubit state $|\psi\rangle$, satisfying $U|0^n\rangle = |\psi\rangle$. Then, can implement $Z_{0^n} := 2|0^n\rangle\langle 0^n| - I$ via a some Toffoli gates and a Z gate, and Z_ψ via $U_\psi Z_{0^n} U_\psi^\dagger$. But notice that implementing $e^{i\phi Z_{0^n}}$ requires the same number of Toffoli gates as implement Z_{0^n}. Therefore the circuit for $e^{i\phi Z_\psi} = U e^{i\phi Z_{0^n}} U^\dagger$ has the same complexity.

- Say $|\psi\rangle$ is the ground state of a Hamiltonian H, and assume the ground space is non-degenerate. Modern algorithms for estimating ground state energies or other observables on the ground state (such as [LT20]) synthesize approximate reflections via singular value transformation: a polynomial $p(x)$ is approximately 1 in the ground space, and -1 elsewhere, so that $p(H) \approx Z_\psi$. But we can also design polynomials that are $e^{i\phi}$ in the ground space and $e^{-i\phi}$ elsewhere, and then $p(H) \approx e^{i\phi Z_\psi}$. These polynomials have the same degree, so they have the same circuit complexities.

- Quantum algorithms based on Szegedy walk unitaries (such as [HW19, AHNTW20]) prepare a unitary U that has an eigenvalue $|\psi\rangle$ with eigenphase 0, and the other eigenphases are at least Δ. To implement Z_ψ, they apply phase estimation to U to extract

an estimate of the eigenphase with precision Δ – enough to distinguish 0 from values larger than Δ. Then they apply a phase flip depending on if the estimate is 0, and uncompute. Similar to the Toffoli circuit, we can also simply apply $e^{i\phi}$ if the estimate is 0 and $e^{-i\phi}$ otherwise, and we have implemented $e^{i\phi Z_\psi}$ with the same circuit complexity.

It seems that in the black-box setting two queries to Z_Π are necessary to implement $e^{i\phi Z_\Pi}$. But once we 'open the black-box' and actually implement it with a quantum circuit, we find that for every example in the literature that we know of the circuits implementing Z_Π and $e^{i\phi Z_\Pi}$ are essentially the same size. As we shall see, access to these oracles significantly expands our capabilities.

In a fault tolerant setting, we may find that the complexity of executing a circuit may depend less on its overall size and more on the number of non-Clifford gates. In this case, $e^{i\phi Z}$ may be significantly more expensive than Z, since Z is a Clifford gate and $e^{i\phi Z}$ is not. The severity of this effect may vary with the level of precision with which phases ϕ are implemented [CO16]. We leave an investigation of these concerns to future work.

1.2 Sampling from Pellian Polynomials

In this section we present an algorithm based on alternating applications of $e^{i\phi Z_\psi}$ and $e^{i\phi Z_\Pi}$. The behavior of such alternating rotations is extensively studied, initially by [YLC14, LYC16]. We find that this lets us toss a coin that comes up heads with probability $|P(a)|^2$ where P belongs to a certain restricted class of polynomials.

Indeed, [MRTC21] presents a derivation of quantum signal processing in terms of amplitude amplification. Our framework can be seen as a synthesis of this observation with the idea of using quantum signal processing to perform Kitaev's phase estimation algorithm on the Grover operator.

[GSLW18] gives a precise characterization of what these polynomials are that is completely independent of the actual phase rotations ϕ. This is exceedingly convenient for quantum algorithms, since we no longer need to worry about circuit synthesis and can simply construct polynomial approximations to the functions we want to apply. We just need to make sure the polynomial we construct meets some simple constraints. Actually computing the phase rotations that implement the polynomial is studied by several recent works [Haah18, Chao&20, WDL21].

However, one thing that is lacking from the literature is simple nomenclature for the different families of polynomials. [Chao&20] introduces the 'Low-algebra' and 'Haah-algebra' for different families of implementable matrices, and [MRTC21] introduced the idea of 'Quantum Signal Processing (QSP) Conventions' such as $(W_x, S_z, \langle 0| \cdot |0\rangle)$-QSP. These ideas are useful for understanding the inner workings of polynomial synthesis. But if all we want to do is refer to synthesizable polynomials without worrying about how to find the phases, then these terms are still too technical.

Instead, we are motivated by the fact that the Chebyshev polynomials are a family of polynomials that are implementable via quantum signal processing. [Demeyer07] furthermore observes that the Chebyshev polynomials of the first kind T_n and of the second kind U_n are solutions of the 'Pell equation':

$$|T_n(x)|^2 + (1-x^2)|U_{n-1}(x)|^2 = 1. \tag{8}$$

Indeed, the Pell equation is satisfied by exactly the polynomials that are directly implementable by quantum signal processing. We thus propose the following nomenclature:

Definition 1. *A pair of polynomials $P(x), Q(x) \in \mathbb{C}[x]$ **form a Pell pair** if they have fixed and opposite parity*[2] *and, for all $x \in [-1, 1]$, they together satisfy:*

$$|P(x)|^2 + (1 - x^2)|Q(x)|^2 = 1. \tag{9}$$

*A polynomial $P(x) \in \mathbb{C}[x]$ is **Pellian** if there exists a $Q(x) \in \mathbb{C}[x]$ such that $P(x), Q(x)$ form a Pell pair. Similarly, a polynomial $Q(x)$ is **Pell-complementary** if there exists a $P(x)$ such that $P(x), Q(x)$ form a Pell pair.*

The condition is not symmetrical with respect to $P(x)$ and $Q(x)$, and therefore the different classes of polynomials have different names. While the above definition is independent of the existence of the phases, it still does not make it easy to check if a polynomial $P(x)$ is Pellian. This is facilitated by Theorem 3 of [GSLW18], which states that a polynomial $P(x) \in \mathbb{C}[x]$ is Pellian if and only if it has fixed parity, and:

- $\forall x \in [-1, 1]$ we have $|P(x)| \leq 1$,
- $\forall x \in (-\infty, -1] \cup [1, \infty)$ we have $|P(x)| \geq 1$,
- if P is even then $\forall x \in \mathbb{R}$ we have $P(ix)P^*(ix) \geq 1$.

Here, P^* refers to complex conjugation of the coefficients of P only.

We have defined the family of polynomials in terms of simple conditions independent of the implementation. Now we move on to the actual algorithm that samples from these polynomials.

The core idea of quantum signal processing is to apply alternating rotations in different bases. In our case, this will correspond to alternatingly applying $e^{i\phi Z_\psi}$ and $e^{i\phi Z_\Pi}$. Keeping track of the dynamics when both bases are rotating is very complicated, so in our analysis we find it useful to standardize the bases. We let $\{|0\rangle, |0^\perp\rangle\}$ be some other basis for the two-dimensional Grover subspace, and introduce a new unitary V that is used entirely for analysis purposes and never appears in the implementation.

$$V := |\psi\rangle\langle 0| + |\psi^\perp\rangle\langle 0^\perp| \tag{10}$$

$$Z_0 := |0\rangle\langle 0| - |0^\perp\rangle\langle 0^\perp| \tag{11}$$

$$e^{i\phi Z_\psi} = V e^{i\phi Z_0} V^\dagger \tag{12}$$

Now, a pair of rotations in different bases becomes:

$$e^{i\phi_1 Z_\psi} e^{i\phi_2 Z_\Pi} = V e^{i\phi_1 Z_0} V^\dagger e^{i\phi_2 Z_\Pi} \tag{13}$$

[2] A polynomial is even (odd) if it is an even (odd) function. If it is either even or odd then we say it has fixed parity, otherwise it has mixed parity. Two polynomials have opposite parity if one of them is even and the other is odd.

Next, observe what happens when we write out V in the $\{|0\rangle, |0^\perp\rangle\}$ and $\{|\Pi\rangle, |\Pi^\perp\rangle\}$ bases via some abuse of notation:

$$V = a\,|\Pi\rangle\langle 0| + \bar{a}\,|\Pi\rangle\langle 0^\perp| + \bar{a}\,|\Pi^\perp\rangle\langle 0| - a\,|\Pi^\perp\rangle\langle 0^\perp| \tag{14}$$

$$=: \begin{bmatrix} a\,|\Pi\rangle\langle 0| & \bar{a}\,|\Pi\rangle\langle 0^\perp| \\ \bar{a}\,|\Pi^\perp\rangle\langle 0| & -a\,|\Pi^\perp\rangle\langle 0^\perp| \end{bmatrix} \tag{15}$$

Although V is not actually a reflection, when viewed in these particular bases the matrix elements look just like those of a reflection matrix. Our goal is to track the complicated motion through the multiple bases of the Grover subspace via just matrix multiplication over $\mathbb{C}^2$. In that case, the $|\Pi\rangle\langle 0|, |\Pi\rangle\langle 0^\perp|, ...$ labels disappear and we are actually just left with a reflection, call it R. Now we can think of $Ve^{i\phi_1 Z_0}V^\dagger e^{i\phi_2 Z_\Pi}$ just as $Re^{i\phi_1 Z}Re^{i\phi_2 Z}$.

Given access to alternating phase rotations and reflections, we invoke quantum signal processing to implement the desired polynomial. Throughout the paper, products run from left to right: $\prod_{j=1}^{3} c_j = c_1 c_2 c_3$.

Lemma 2. ***Implementing Pellian polynomials via phase rotations and reflections.*** *Say $P, Q \in \mathbb{C}[a]$ form a Pell pair, and say $d := deg(P)$. Then there exist phases $\phi_1, ..., \phi_{d-1}$, and another phase φ such that:*

$$\prod_{j=1}^{d-1}(Re^{i\phi_j Z})R = i^{-d}\begin{bmatrix} e^{i\varphi}P & iQ\bar{a} \\ iQ^*\bar{a} & e^{-i\varphi}P^* \end{bmatrix} \text{ where } R := \begin{bmatrix} a & \bar{a} \\ \bar{a} & -a \end{bmatrix}. \tag{16}$$

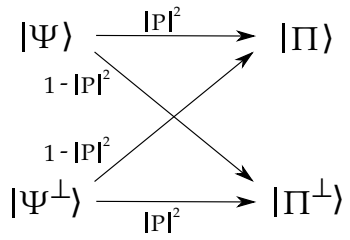
Proof. This follows from Theorem 3 and Corollary 8 of [GSLW18] via the substitution $e^{i\arccos(a)X} = ie^{-i\frac{\pi}{4}Z}Re^{-i\frac{\pi}{4}Z}$. □

Proposition 3. *Say P is a Pellian polynomial. Then, there exist algorithms that:*

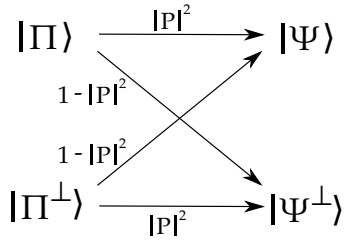
- *take any one of $|\psi\rangle, |\psi^\perp\rangle, |\Pi\rangle, |\Pi^\perp\rangle$ as input,*
- *apply some number of $e^{i\phi Z_\psi}$, $e^{i\phi Z_\Pi}$ gates,*
- *and finally measure in the $|\psi\rangle, |\psi^\perp\rangle$ basis or the $|\Pi\rangle, |\Pi^\perp\rangle$ basis.*

The $|\psi\rangle$ and $|\Pi\rangle$ outcomes occur with probability $|P(a)|^2$, and the $|\psi^\perp\rangle$ and $|\Pi^\perp\rangle$ outcomes occur with probability $1 - |P(a)|^2$. The final measurement basis and the number of queries to $e^{i\phi Z_\psi}$, $e^{i\phi Z_\Pi}$ are determined by the degree of P and the input state. The details are given by the transition diagrams in Figure 1.

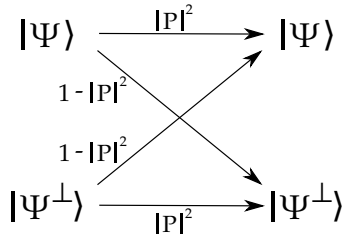
Proof. The transitions in Figure 1 are performed by applying one of the following unitaries, and then measuring in either the $\{|\psi\rangle, |\psi^\perp\rangle\}$ basis or the $\{|\Pi\rangle, |\Pi^\perp\rangle\}$ basis by using a Hadamard test on Z_ψ or Z_Π. Say P, Q form a Pell pair. Then, there exist phases



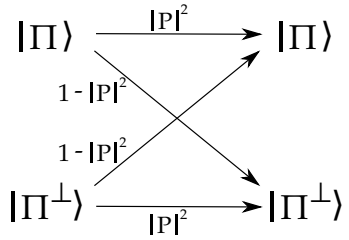
(a) $\deg(P) = 2k+1, \quad Q_\psi = k, \quad Q_\Pi = k+1$



(b) $\deg(P) = 2k+1, \quad Q_\psi = k+1, \quad Q_\Pi = k$



(c) $\deg(P) = 2k, \quad Q_\psi = k, \quad Q_\Pi = k$



(d) $\deg(P) = 2k, \quad Q_\psi = k, \quad Q_\Pi = k$

Figure 1: Transition diagrams implemented by the algorithms in Proposition 3. Given a Pellian polynomial P and one of the states $|\psi\rangle, |\psi^\perp\rangle, |\Pi\rangle, |\Pi^\perp\rangle$, we can implement one of these transitions using some number of queries Q_ψ, Q_Π to $e^{i\phi Z_\psi}, e^{i\phi Z_\Pi}$ respectively.

$\phi_1, ..., \phi_{\deg(P)-1}$ such that:

$$U_{(a)} := \prod_{j=1}^{k} \left(e^{i\phi_{2j-1} Z_\psi} e^{i\phi_{2j} Z_\Pi} \right) = i^{-d} \begin{bmatrix} e^{i\varphi} P |\Pi\rangle \langle\psi| & iQ\bar{a} |\Pi\rangle \langle\psi^\perp| \\ iQ^* \bar{a} |\Pi^\perp\rangle \langle\psi| & e^{-i\varphi} P^* |\Pi^\perp\rangle \langle\psi^\perp| \end{bmatrix} \tag{17}$$

$$U_{(b)} := \prod_{j=1}^{k} \left(e^{i\phi_{2j-1} Z_\Pi} e^{i\phi_{2j} Z_\psi} \right) = i^{-d} \begin{bmatrix} e^{i\varphi} P |\psi\rangle \langle\Pi| & iQ\bar{a} |\psi\rangle \langle\Pi^\perp| \\ iQ^* \bar{a} |\psi^\perp\rangle \langle\Pi| & e^{-i\varphi} P^* |\psi^\perp\rangle \langle\Pi^\perp| \end{bmatrix} \tag{18}$$

$$U_{(c)} := \prod_{j=1}^{k-1} \left(e^{i\phi_{2j-1} Z_\Pi} e^{i\phi_{2j} Z_\psi} \right) e^{i\phi_{2k-1} Z_\Pi} = i^{-d} \begin{bmatrix} e^{i\varphi} P |\psi\rangle \langle\psi| & iQ\bar{a} |\psi\rangle \langle\psi^\perp| \\ iQ^* \bar{a} |\psi^\perp\rangle \langle\psi| & e^{-i\varphi} P^* |\psi^\perp\rangle \langle\psi^\perp| \end{bmatrix} \tag{19}$$

$$U_{(d)} := \prod_{j=1}^{k-1} \left(e^{i\phi_{2j-1} Z_\psi} e^{i\phi_{2j} Z_\Pi} \right) e^{i\phi_{2k-1} Z_\psi} = i^{-d} \begin{bmatrix} e^{i\varphi} P |\Pi\rangle \langle\Pi| & iQ\bar{a} |\Pi\rangle \langle\Pi^\perp| \\ iQ^* \bar{a} |\Pi^\perp\rangle \langle\Pi| & e^{-i\varphi} P^* |\Pi^\perp\rangle \langle\Pi^\perp| \end{bmatrix} \tag{20}$$

We will later show that these equalities hold – suppose for now that they do. Then, the query complexities Q_ψ and Q_Π can just be read off from the products on the left hand sides plus the extra query for measurement, and the transition probabilities can be obtained by taking the magnitude squared of the matrix elements on the right hand sides. Recall that $|\bar{a}Q(a)|^2 = 1 - |P(a)|^2$ since P, Q form a Pell pair. So all that remains to show is that these equalities are indeed true, that $U_{(a)}, U_{(b)}$ correspond to odd P, and that $U_{(c)}, U_{(d)}$ correspond to even P.

The calculation for all of these unitaries are extremely similar, so we will just prove the equality for $U_{(a)}$. Recall the analysis sketch above, where we considered another basis $|0\rangle, |0^\perp\rangle$ for the Grover subspace, and let $V := |\psi\rangle \langle 0| + |\psi^\perp\rangle \langle 0^\perp|$. We then wrote $Z_0 :=$

$|0\rangle\langle 0| - |0^\perp\rangle\langle 0^\perp|$ and $e^{i\phi Z_\psi} = V e^{i\phi Z_0} V^\dagger$. Now we can rewrite the $U_{(a)}$ as:

$$U_{(a)} := \prod_{j=1}^{k} \left(e^{i\phi_{2j-1} Z_\psi} e^{i\phi_{2j} Z_\Pi} \right) = \prod_{j=1}^{k} \left(V e^{i\phi_{2j-1} Z_0} V^\dagger e^{i\phi_{2j} Z_\Pi} \right) V \cdot V^\dagger \tag{21}$$

Now we expand each of the components V, $e^{i\phi Z_0}$ and $e^{i\phi Z_\Pi}$ in the $\{|0\rangle, |0^\perp\rangle\}$ and $\{|\Pi\rangle, |\Pi^\perp\rangle\}$ bases:

$$V = \begin{bmatrix} a|\Pi\rangle\langle 0| & \bar{a}|\Pi\rangle\langle 0^\perp| \\ \bar{a}|\Pi^\perp\rangle\langle 0| & -a|\Pi^\perp\rangle\langle 0^\perp| \end{bmatrix} \tag{22}$$

$$e^{i\phi Z_0} = \begin{bmatrix} e^{i\phi}|0\rangle\langle 0| & \\ & e^{-i\phi}|0^\perp\rangle\langle 0^\perp| \end{bmatrix} \tag{23}$$

$$e^{i\phi Z_\Pi} = \begin{bmatrix} e^{i\phi}|\Pi\rangle\langle \Pi| & \\ & e^{-i\phi}|\Pi^\perp\rangle\langle \Pi^\perp| \end{bmatrix} \tag{24}$$

We observe that the basis expansions of the components line up perfectly when plugged into (21). That means that the unitary implemented by the algorithm can be obtained by matrix multiplication over $\mathbb{C}^2$:

$$\prod_{j=1}^{k} \left(V e^{i\phi_{2j-1} Z_0} V^\dagger e^{i\phi_{2j} Z_\Pi} \right) V = \begin{bmatrix} \alpha|\Pi\rangle\langle 0| & \beta|\Pi\rangle\langle 0^\perp| \\ \gamma|\Pi^\perp\rangle\langle 0| & \delta|\Pi^\perp\rangle\langle 0^\perp| \end{bmatrix} \tag{25}$$

$$\prod_{j=1}^{k} \left(R e^{i\phi_{2j-1} Z} R^\dagger e^{i\phi_{2j} Z} \right) R = \begin{bmatrix} \alpha & \beta \\ \gamma & \delta \end{bmatrix} \tag{26}$$

Since $R = R^\dagger$ this multiplication over $\mathbb{C}^2$ matches the expression in Lemma 2. Recall $d := \deg(P)$.

$$\prod_{j=1}^{k} \left(R e^{i\phi_{2j-1} Z} R^\dagger e^{i\phi_{2j} Z} \right) R = \prod_{j=1}^{d} \left(R e^{i\phi_j Z} \right) R = i^{-d} \begin{bmatrix} e^{i\varphi} P & iQ\bar{a} \\ iQ^*\bar{a} & e^{-i\varphi} P^* \end{bmatrix} \tag{27}$$

$$\prod_{j=1}^{k} \left(V e^{i\phi_{2j-1} Z_0} V^\dagger e^{i\phi_{2j} Z_\Pi} \right) V = i^{-d} \begin{bmatrix} e^{i\varphi} P|\Pi\rangle\langle 0| & iQ\bar{a}|\Pi\rangle\langle 0^\perp| \\ iQ^*\bar{a}|\Pi^\perp\rangle\langle 0| & e^{-i\varphi} P^*|\Pi^\perp\rangle\langle 0^\perp| \end{bmatrix} \tag{28}$$

So we have established:

$$U_{(a)} := \prod_{j=1}^{k} \left(e^{i\phi_{2j-1} Z_\psi} e^{i\phi_{2j} Z_\Pi} \right) = i^{-d} \begin{bmatrix} e^{i\varphi} P|\Pi\rangle\langle \psi| & iQ\bar{a}|\Pi\rangle\langle \psi^\perp| \\ iQ^*\bar{a}|\Pi^\perp\rangle\langle \psi| & e^{-i\varphi} P^*|\Pi^\perp\rangle\langle \psi^\perp| \end{bmatrix} \tag{29}$$

The calculations for $U_{(b)}, U_{(c)}$, and $U_{(d)}$ proceed similarly: plug in $e^{i\phi Z_\psi} = V e^{i\phi Z_0} V^\dagger$, and write out $V, e^{i\phi Z_0}$, and $e^{i\phi Z_\Pi}$ explicitly in the $\{|0\rangle, |0^\dagger\rangle\}$ and $\{|\Pi\rangle, |\Pi^\perp\rangle\}$ bases, while inserting $VV^\dagger$ as needed. All the bases will line up, so we can treat this as just regular multiplication of 2x2 matrices. Then invoke Lemma 2.

Observe also that the number of phases that appear in Lemma 2 is $\deg(P) - 1$. Since each phase corresponds to one call to either $e^{i\phi Z_\psi}$ or $e^{i\phi Z_\Pi}$, we see that $U_{(a)}, U_{(b)}$ correspond to odd polynomials and $U_{(c)}, U_{(d)}$ to even polynomials.

□

1.3 Sampling from Semi-Pellian Polynomials

Some of the later sections in this paper require sampling access to a very broad family of polynomials. The polynomials are obtained from techniques that approximate arbitrary functions, and are then post-processed through shifting and scaling. To this end, Pellian polynomials are still rather restricted. The condition that $P(ix)P^*(ix) > 1$ for even P is very difficult to ensure, essentially forcing us to use odd P only. Furthermore, Pellian polynomials must also satisfy $|P(\pm 1)| = 1$, which is also not always desirable. Thus, we would like access to a more flexible family of polynomials.

Fortunately, there exists an implementable family of real polynomials that is not so restricted.

Definition 4. *A polynomial $P \in \mathbb{R}[x]$ is **semi-Pellian** if it is the real part of a Pellian polynomial.*

Indeed, [GSLW18] not only characterize this family, but also show how to obtain the corresponding Pellian polynomials: Corollary 10 of [GSLW18] states that a polynomial $P(x) \in \mathbb{R}[x]$ is semi-Pellian if and only if it has fixed parity, and $\forall x \in [-1, 1]$ we have $|P(x)| \leq 1$.

To sample from semi-Pellian polynomials, we follow a similar technique to Corollary 18 of [GSLW18]. Say P is semi-Pellian, and say $\tilde{P}$ is a Pellian polynomial with $\mathrm{Re}(\tilde{P}) = P$. We conditionally apply phase rotations $e^{i\phi Z}$ or $e^{-i\phi Z}$, which implements either $\tilde{P}$ or $\tilde{P}^*$. Then we use a linear combination of unitaries to average these together to obtain P.

Linear combinations of unitaries require an additional flag qubit, and depending on the value of the flag we either obtain $\mathrm{Re}(\tilde{P})$ or $i\mathrm{Im}(\tilde{P})$. Thus, the bit $\mathbf{b}$ that we sample from depends on the joint outcome of the flag qubit and the output register. Independently of the flag qubit, the state found in the output register depends only on $|\tilde{P}|$ just like Proposition 3. The proof of Proposition 5 shows this calculation in detail.

To efficiently implement phase rotations with a conditional sign flip, we require oracles of the form:

$$e^{i\phi Z_\psi} \oplus e^{-i\phi Z_\psi} = |0\rangle\langle 0| \otimes e^{i\phi Z_\psi} + |1\rangle\langle 1| \otimes e^{-i\phi Z_\psi} \tag{30}$$

and similarly for Π. Such an oracle can be implemented with one controlled and one regular query to $e^{i\phi Z_\psi}$. But, again, we claim that in most practical situations $e^{i\phi Z_\psi} \oplus e^{-i\phi Z_\psi}$ can be implemented with about the same query complexity as Z_ψ. As articulated above, most quantum algorithms take the following form: some computation flips a qubit depending on if a phase should be applied, applies Z to that qubit, and then performs an uncomputation. The cost of the circuit is dominated by the computation and uncomputation. So, if the Z is replaced with $e^{i\phi Z}$ or even $e^{i\phi Z} \oplus e^{-i\phi Z}$, the additional cost is negligible compared to the rest of the circuit.

Proposition 5. *Say P is a semi-Pellian polynomial, and say $\tilde{P}$ is a Pellian polynomial such that $P = Re(\tilde{P})$. Then, there exist algorithms similar to the ones in Proposition 3 that output a state $|out\rangle \in \{|\psi\rangle, |\psi^\perp\rangle\}$ or $|out\rangle \in \{|\Pi\rangle, |\Pi^\perp\rangle\}$ as well as an additional state $|flag\rangle \in \{|0\rangle, |1\rangle\}$.*

Each of these algorithms have a target state $|target\rangle \in \{|\psi\rangle, |\psi^\perp\rangle, |\Pi\rangle, |\Pi^\perp\rangle\}$ so that:

$$Pr[|flag\rangle = |0\rangle \text{ and } |out\rangle = |target\rangle] = |P(a)|^2 \tag{31}$$

$$Pr[|out\rangle = |target\rangle] = \left|\tilde{P}(a)\right|^2 \tag{32}$$

The output basis, $|target\rangle$, as well as the query complexities to $e^{i\phi Z_\psi} \oplus e^{-i\phi Z_\psi}$ and $e^{i\phi Z_\Pi} \oplus e^{-i\phi Z_\Pi}$ are shown in the following table:

$deg(P)$	*Input*	*Output Basis*	$\|target\rangle$	$e^{i\phi Z_\psi} \oplus e^{-i\phi Z_\psi}$	$e^{i\phi Z_\Pi} \oplus e^{-i\phi Z_\Pi}$
2k+1	$\|\psi\rangle$	$\|\Pi\rangle, \|\Pi^\perp\rangle$	$\|\Pi\rangle$	*k+1*	*k+2*
2k+1	$\|\psi^\perp\rangle$	$\|\Pi\rangle, \|\Pi^\perp\rangle$	$\|\Pi^\perp\rangle$	*k+1*	*k+2*
2k+1	$\|\Pi\rangle$	$\|\psi\rangle, \|\psi^\perp\rangle$	$\|\psi\rangle$	*k+2*	*k+1*
2k+1	$\|\Pi^\perp\rangle$	$\|\psi\rangle, \|\psi^\perp\rangle$	$\|\psi^\perp\rangle$	*k+2*	*k+1*
2k	$\|\psi\rangle$	$\|\psi\rangle, \|\psi^\perp\rangle$	$\|\psi\rangle$	*k+3*	*k*
2k	$\|\psi^\perp\rangle$	$\|\psi\rangle, \|\psi^\perp\rangle$	$\|\psi^\perp\rangle$	*k+3*	*k*
2k	$\|\Pi\rangle$	$\|\Pi\rangle, \|\Pi^\perp\rangle$	$\|\Pi\rangle$	*k*	*k+3*
2k	$\|\Pi^\perp\rangle$	$\|\Pi\rangle, \|\Pi^\perp\rangle$	$\|\Pi^\perp\rangle$	*k*	*k+3*

Proof. Just as in Proposition 3 there are actually only four algorithms since the algorithm only depends on the input basis and the parity of the degree. We only work though the case when $\deg(P) = 2k+1$ and the input is $|\psi\rangle$ or $|\psi^\perp\rangle$, since the other cases follow the same pattern. Let $\phi_1, ..., \phi_{2k}$ and φ be the phases from Lemma 2 corresponding to $\tilde{P}$ and its Pell-complementary polynomial $\tilde{Q}$. Select $\phi_0 = \phi_{2k+1} = -\varphi/2$ to achieve:

$$e^{i\phi_0 Z} \prod_{j=1}^{2k+1} (Re^{i\phi_j Z}) = (-i)^{2k+1} \begin{bmatrix} \tilde{P} & i\tilde{Q}\bar{a} \\ i\tilde{Q}\bar{a} & \tilde{P} \end{bmatrix} \tag{33}$$

Using the same techniques from Proposition 3 we show:

$$i^{2k+1} e^{i\phi_0 Z_\Pi} \prod_{j=1}^{k} (e^{i\phi_{2j-1} Z_\psi} e^{i\phi_{2j} Z_\Pi}) e^{i\phi_{2k+1} Z_\psi} = \begin{bmatrix} \tilde{P} |\Pi\rangle \langle\psi| & i\tilde{Q}\bar{a} |\Pi\rangle \langle\psi^\perp| \\ i\tilde{Q}^*\bar{a} |\Pi^\perp\rangle \langle\psi| & \tilde{P}^* |\Pi^\perp\rangle \langle\psi^\perp| \end{bmatrix} =: A \tag{34}$$

$$(-i)^{2k+1} e^{-i\phi_0 Z_\Pi} \prod_{j=1}^{k} (e^{-i\phi_{2j-1} Z_\psi} e^{-i\phi_{2j} Z_\Pi}) e^{-i\phi_{2k+1} Z_\psi} = \begin{bmatrix} \tilde{P}^* |\Pi\rangle \langle\psi| & -i\tilde{Q}^*\bar{a} |\Pi\rangle \langle\psi^\perp| \\ -i\tilde{Q}\bar{a} |\Pi^\perp\rangle \langle\psi| & \tilde{P} |\Pi^\perp\rangle \langle\psi^\perp| \end{bmatrix} =: B \tag{35}$$

Now we apply the trick from Corollary 18 of [GSLW18]: we use a linear combinations of unitaries circuit to perform the average of two circuits A, B, so the top left matrix element becomes P. The circuit involves preparing an ancilla qubit in the $|+\rangle$ state, applying either A or B depending on the ancilla, and then applying a Hadamard gate to the ancilla:

$$(H \otimes I)(A \oplus B)(|+\rangle \otimes I) = |0\rangle \otimes \frac{A+B}{2} + |1\rangle \otimes \frac{A-B}{2} \tag{36}$$

This extra register on the left is $|\text{flag}\rangle$. We find:

$$\frac{A+B}{2} = \begin{bmatrix} \text{Re}(\tilde{P}) |\Pi\rangle \langle\psi| & -\text{Im}(\tilde{Q})\bar{a} |\Pi\rangle \langle\psi^\perp| \\ \text{Im}(\tilde{Q})\bar{a} |\Pi^\perp\rangle \langle\psi| & \text{Re}(\tilde{P}) |\Pi^\perp\rangle \langle\psi^\perp| \end{bmatrix} \tag{37}$$

$$\frac{A-B}{2} = \begin{bmatrix} i\text{Im}(\tilde{P}) |\Pi\rangle \langle\psi| & i\text{Re}(\tilde{Q})\bar{a} |\Pi\rangle \langle\psi^\perp| \\ i\text{Re}(\tilde{Q})\bar{a} |\Pi^\perp\rangle \langle\psi| & -i\text{Im}(\tilde{P}) |\Pi^\perp\rangle \langle\psi^\perp| \end{bmatrix} \tag{38}$$

If we begin in the state $|\psi\rangle$, then the $|0\rangle|\Pi\rangle$ component of the output state has amplitude $\mathrm{Re}(\tilde{P}) = P$, so we obtain this case with probability $|P|^2$. The $|1\rangle|\Pi\rangle$ component has amplitude $i\mathrm{Im}(\tilde{P})$, so the total probability of seeing $|\mathrm{out}\rangle = |\Pi\rangle$ regardless of $|\mathrm{flag}\rangle$ is $|\mathrm{Re}(\tilde{P})|^2 + |i\mathrm{Im}(\tilde{P})|^2 = |\tilde{P}|^2$. A similar pattern holds for the input state $|\psi^\perp\rangle$.

All that remains to show is how to actually implement the unitary $A \oplus B$ and to count the total query complexity. Using the identity $CD \oplus EF = (C \oplus E)(D \oplus F)$ we see from (34, 35) that $A \oplus B$ factors into a product of a $(i^{2k+1} \oplus (-i)^{2k+1})$ gate, and $k+1$ many $(e^{i\phi Z_\psi} \oplus e^{-i\phi Z_\psi})$ and $(e^{i\phi Z_\Pi} \oplus e^{-i\phi Z_\Pi})$ gates each. Finally, we require one more query to $(e^{i\phi Z_\Pi} \oplus e^{-i\phi Z_\Pi})$ to measure in the $\{|\Pi\rangle, |\Pi\rangle^\perp\}$ basis. □

1.4 Previous Results in this Framework

Having established Propositions 3 and 5, we can state the full version of the theorem.

Theorem 6. ***Sampling from a polynomial.*** *Say $P(a)$ is a Pellian or a semi-Pellian polynomial. Then, there exists a quantum algorithm that samples from a random variable $\mathbf{b}_P \in \{0, 1\}$ such that:*

$$Pr[\mathbf{b}_P = 1] = |P(a)|^2. \tag{39}$$

This algorithm takes a copy of $|\psi\rangle, |\psi^\perp\rangle, |\Pi\rangle$ or $|\Pi^\perp\rangle$ as input, and also outputs one of these four states at random. This algorithm makes $O(deg(P))$ oracle queries to $e^{i\phi Z_\psi}, e^{i\phi Z_\Pi}$ if P is Pellian, and $(e^{i\phi Z_\psi} \oplus e^{-i\phi Z_\psi})$, $(e^{i\phi Z_\Pi} \oplus e^{-i\phi Z_\Pi})$ if P is semi-Pellian.

We call invoking this theorem on a particular polynomial P to **sample from the polynomial** P. The algorithm is specifically designed to be used repeatedly. It accepts any of the four states $|\psi\rangle, |\psi^\perp\rangle, |\Pi\rangle, |\Pi^\perp\rangle$ as input, and also only ever outputs one of these. So, regardless of the actual measurement outcomes and regardless of the sequence of polynomials sampled from, we can always chain invocations of Theorem 6 into a very long circuit.

This is only useful in situations where an input state $|\psi\rangle$ is expensive to prepare. In that case we might want to reuse that state as much as possible, and indeed Theorem 6 can in principle be applied arbitrarily many times given only one copy of $|\psi\rangle$. But we are by no means forced to do this: at any point, we can discard the output of the algorithm and re-prepare $|\psi\rangle$. Furthermore, the schedule of polynomials is completely independent of this choice.

Recall that Chebyshev polynomials of the first kind $T_n(a) = \cos(n \arccos(a))$ are Pellian, and can be sampled from by setting all the ϕ_j to $\pi/2$, in which case $e^{i\phi_j Z_\psi} = Z_\psi$ and $e^{i\phi_j Z_\Pi} = Z_\Pi$. In this case, $U_{(a)}$ from Proposition 3 simply becomes Grover's algorithm: alternating applications of Z_Π and Z_ψ, followed by a $\{|\Pi\rangle, |\Pi^\perp\rangle\}$-basis measurement. So we see that several previous results in amplitude estimation can be viewed as invoking a special case of Theorem 6. We will list these momentarily.

To aid in comparing to previous results and also describe our new results in later sections, we set up some standard nomenclature about amplitude estimation algorithms. In particular, we define the random variables $\hat{\mathbf{a}}, \mathbf{Q}_\Pi, \mathbf{Q}_\psi, \mathbf{d}$ and $\mathbf{D}$. The query complexities $\mathbf{Q}_\Pi, \mathbf{Q}_\psi$, and degrees $\mathbf{d}, \mathbf{D}$ may be random since the algorithms are adaptive sometimes and can vary their runtime based on the results of measurement outcomes.

Definition 7. *An **amplitude estimation algorithm** is an algorithm that, starting with $|\psi\rangle$, repeatedly samples from polynomials until it finally outputs an estimate of a as well as one of $|\psi\rangle, |\psi^\perp\rangle, |\Pi\rangle, |\Pi^\perp\rangle$. The estimate is denoted by $\hat{\mathbf{a}}$ and is a random variable over $\mathbb{R}$. The total number of queries to $e^{i\phi Z_\Pi}$, $e^{i\phi Z_\psi}$ or $(e^{i\phi Z_\psi} \oplus e^{-i\phi Z_\psi})$, $(e^{i\phi Z_\Pi} \oplus e^{-i\phi Z_\Pi})$ are denoted by the random variables $\mathbf{Q}_\Pi, \mathbf{Q}_\psi \in \mathbb{Z}^+$ respectively. Finally, the highest degree of a polynomial ever sampled from is $\mathbf{d} \in \mathbb{Z}^+$ and the sum of the degrees of all the sampled polynomials is $\mathbf{D} \in \mathbb{Z}^+$.*

$\mathbf{D}$ can be regarded as the overall query complexity of the algorithm. If only one state $|\psi\rangle$ is used, then $\mathbf{D}$ is also the circuit depth. But if the state $|\psi\rangle$ is reset every time, then maximum circuit depth is only $\mathbf{d}$, even though the total query complexity is still $\mathbf{D}$.

Proposition 8. *Say an amplitude estimation algorithm only ever samples from Pellian polynomials. Then $\mathbf{Q}_\Pi + \mathbf{Q}_\psi = \mathbf{D}$.*

Now we list some previous results on amplitude estimation using the notation above.

Relative-error estimation. ([AR19], Theorem 3) For any $\varepsilon, \delta \in [0, 1/2]$ there exists an amplitude estimation algorithm such that:

$$\Pr[a(1-\varepsilon) \leq \hat{\mathbf{a}} \leq a(1+\varepsilon)] \geq 1-\delta \tag{40}$$

$$\exists f(a,\varepsilon,\delta) \in O\left(a^{-1}\varepsilon^{-1}\log(\delta^{-1})\right) \text{ such that } \Pr[\mathbf{D} \leq f(a,\varepsilon,\delta)] \geq 1-\delta. \tag{41}$$

Fast additive-error estimation. ([GGZW19], Theorem 1) For any $\varepsilon, \delta \in [0, 1/2]$ there exists an amplitude estimation algorithm such that:

$$\Pr[|\hat{\mathbf{a}} - a| \leq \varepsilon] \geq 1-\delta \tag{42}$$

$$\Pr\left[\mathbf{Q_\Pi} \leq \frac{14}{\varepsilon}\log\left(\frac{2}{\delta}\log_2\left(\frac{\pi}{4\varepsilon}\right)\right)\right] \geq 1-\delta \tag{43}$$

Non-adaptive relative-error approximate counting. ([VO20], Algorithm 1) Suppose $a \geq 1/N$ for some known $N \in \mathbb{Z}^+$. For any $\varepsilon, \delta \in [0, 1/2]$ there exists an amplitude estimation algorithm such that all polynomials depend exclusively on ε, δ, N and are thus all known beforehand. It satisfies:

$$\Pr[a(1-\varepsilon) \leq \hat{\mathbf{a}} \leq a(1+\varepsilon)] \geq 1-\delta \tag{44}$$

$$\mathbf{D} \text{ is not random and } \mathbf{D} \in O\left(\sqrt{N/\varepsilon}\,\log(\delta^{-1})\right) \tag{45}$$

Hybrid quantum-classical estimation. ([GKLPZ20], QoPrime Algorithm.) For any $\varepsilon, \delta \in [0, 1/2]$, and for any $0 < \beta \leq 1$ there exists an amplitude estimation algorithm such that:

$$\Pr[|\hat{\mathbf{a}} - a| \geq \varepsilon] \leq \delta \tag{46}$$

$$\mathbf{D} \text{ is not random and } \mathbf{D} \in O\left(\varepsilon^{-(1+\beta)}\,\log(\delta^{-1})\right) \tag{47}$$

$$\mathbf{d} \text{ is not random and } \mathbf{d} \in O\left(\varepsilon^{-(1-\beta)}\right) \tag{48}$$

We focused here on results that proved the correctness of their algorithms, which leaves out [Suzuki&19]. All of the above algorithms, [Suzuki&19] included, repeatedly invoke the special case of Theorem 6 where $P(a) = T_n(a)$. The research papers presenting these algorithms describe discarding and re-preparing the input state $|\psi\rangle$ every single time. But since Theorem 6 does not require this, these algorithms could all also work with just a single copy of $|\psi\rangle$.

This means our framework has already extended the capabilities of previous works. Although, for the result of [GKLPZ20] in particular, the whole point of separating $\mathbf{d}$ of $\mathbf{D}$ was to reduce the maximum circuit depth. If a single copy of $|\psi\rangle$ were re-used, the maximum depth would be $\mathbf{D}$, not $\mathbf{d}$. So this extension is not always sensible.

We also did not list [BHMT00] and [HW19] because these do not fit into the framework of Theorem 6. Both of these algorithms involve phase estimation of the Grover operator, rather than simply repeated application. However, this complicated and expensive technique has largely been superseded by the faster, simpler, and more versatile algorithms listed above. There is only one capability that has not yet been 'modernized': the non-destructive estimation algorithm from [HW19]. This task is the subject of the next section.

2 State Repair

In the traditional setting for Grover's algorithm where we are searching for a marked item, the state $|\psi\rangle$ is just a uniform superposition over all items. In this case, $|\psi\rangle$ is trivial to prepare and to reflect about, at least compared to querying which items are marked.

However, there also exist settings where $|\psi\rangle$ is extremely expensive to prepare despite being relatively easy to reflect about. In this situation, one might hope for an amplitude estimation algorithm that could make do with a single copy of $|\psi\rangle$, and ideally also give that copy of $|\psi\rangle$ back once an estimate of $|\Pi|\psi\rangle|$ has been extracted. $|\psi\rangle$ could then be used for another part of the algorithm. We call this task **non-destructive amplitude estimation**.

We know of two examples of such situations in the literature. One of them stems from the study of physical systems: say we have a Hamiltonian H with a ground state $|\psi_0\rangle$. Our goal is to estimate the expectations of several observables $\langle\psi_0|O_1|\psi_0\rangle, \langle\psi_0|O_2|\psi_0\rangle$, etc. Suppose for the moment that the ground space is non-degenerate. In that case, following [LT20], we can design a polynomial $p(x)$ that is ≈ 1 in the ground space and ≈ -1 elsewhere, so that $p(H) \approx Z_{\psi_0}$. Assuming the spectral gap of H is not too small, we can now efficiently reflect about $|\psi_0\rangle$. In order to prepare $|\psi_0\rangle$ however, we have to guess some trial state $|\phi\rangle$ that hopefully has large overlap $|\langle\phi|\psi_0\rangle|$ with $|\psi_0\rangle$, and use amplitude amplification to transform $|\phi\rangle \to |\psi_0\rangle$. Since ground state finding is QMA-complete, this step may take exponential time in general. So if it is really worth investing the time to prepare $|\psi_0\rangle$, we better only need to do so once. The previous section already describes an algorithm that requires only one copy of $|\psi_0\rangle$ to estimate a quantity, but a non-destructive algorithm could be used to extract several expectations from a single state[3].

Another example is the kind of quantum algorithm studied by [HW19] and [AHNTW20]. These works consider a classical Hamiltonian $H(x)$, and are interested in preparing a qsample

[3]To be more explicit here, say we can prepare $|\psi_0\rangle$ using P reflections, and we can repair $|\psi_0\rangle$ using R reflections. If the trial state $|\phi\rangle$ has small overlap with $|\psi_0\rangle$ then $P \gg R$. A naive approach for estimating M many observables would then cost $O(MP)$, while an approach based on state repair only costs $O(P + MR)$.

of its thermal distribution $|\psi_\beta\rangle$, where $\langle x|\psi_\beta\rangle \propto e^{-\beta H(x)/2}$. We are given reflection operators $2|\psi_\beta\rangle\langle\psi_\beta| - I$ not just for the target temperature β^* but also all intermediate temperatures. The idea is to find a sequence of temperatures $\beta_0, \beta_1, ...\beta^*$ called a 'Chebyshev cooling schedule' that guarantees $\langle\psi_{\beta_i}|\psi_{\beta_{i+1}}\rangle$ is never too small. With $\beta_0 = 0$, $|\psi_{\beta_0}\rangle$ is just the uniform superposition which is easy to prepare. Then we can use amplitude amplification to transform $|\psi_{\beta_0}\rangle \to |\psi_{\beta_1}\rangle \to ... \to |\psi_{\beta^*}\rangle$. The remaining challenge is to identify the temperatures $\beta_1, \beta_2, ...$ such that $\langle\psi_{\beta_i}|\psi_{\beta_{i+1}}\rangle$ is never too small. If we have a copy of $|\psi_{\beta_i}\rangle$, we can identify β_{i+1} via binary search, invoking amplitude estimation at each candidate temperature to estimate the overlap $\langle\psi_{\beta_i}|\psi_{\beta_{i+1}}\rangle$. This crucially requires amplitude estimation to be non-destructive, since the state $|\psi_{\beta_i}\rangle$ was expensive to prepare and needs to be re-used for several estimations and finally transformed into $|\psi_{\beta_{i+1}}\rangle$.

To this end, [HW19] developed a non-destructive amplitude estimation algorithm based on the methods of [BHMT00]. After running [BHMT00] to obtain an estimate, we obtain an eigenstate of the Grover operator:

$$|\psi_\pm\rangle := \frac{1}{\sqrt{2}}\left(|\psi\rangle \pm i\,|\psi^\perp\rangle\right) \tag{49}$$

Measuring this state in the $\{|\psi\rangle, |\psi^\perp\rangle\}$ basis, we have 'repaired' a copy $|\psi\rangle$ with probability 1/2. If we obtain $|\psi^\perp\rangle$, we can just repeat the procedure until we succeed.

Proposition 9. ***State repair given bounds on*** a**.** *([HW19], Theorem 18.) Say we are given any quantum state in the Grover subspace span$(|\psi\rangle, \Pi|\psi\rangle)$, and suppose $a := |\Pi|\psi\rangle|$ satisfies $\kappa < a < \sqrt{1-\kappa^2}$ for some known $\kappa > 0$. Then, for any $\delta > 0$, there exists a quantum algorithm that, with success probability $\geq 1-\delta$, prepares either $|\psi\rangle$ or $|\psi^\perp\rangle$, each with probability 1/2. It makes $O(\kappa^{-1}\log(\delta^{-1}))$ queries to Z_ψ and Z_Π.*

However, [HW19], featured a hidden assumption. We made this assumption explicit in the above: that some bounds $\kappa < a < \sqrt{1-\kappa^2}$ are known. κ informs the precision of phase estimation on the Grover operator: if precision is not high enough, then we fail to distinguish the eigenvectors $|\psi_\pm\rangle$, so measuring the output fails to collapse the superposition over them. That means we do not have a copy of $|\psi_\pm\rangle$ after running [BHMT00], so the probability of obtaining $|\psi\rangle$ after measuring might be very small[4]. In their analysis, [HW19] implicitly focus on the case where $\varepsilon < a < 1-\varepsilon$, in which case the procedure always works. But if no bounds on a are known, then it is not clear how to use their result to achieve non-destructive estimation.

In this section we present an amplitude estimation algorithm that overcomes this limitation. In Theorem 15 we show that any amplitude estimation algorithm with total degree $\mathbf{D}$ can be modified to also output a copy of the input state $|\psi\rangle$ at the end with probability $\geq 1-\delta$ for any $\delta > 0$. This new algorithm has total degree $O(\delta^{-1/2}\,\mathbf{D})$.

Just like [HW19], the main idea is to add a 'repair step' to the end of another amplitude estimation algorithm that transforms whatever state we have back into $|\psi\rangle$. However, the only piece of information that the repair step needs is the *length* of the estimation algorithm that was just executed. We emphasize the flexibility of this result: our repair procedure can be appended to any amplitude estimation algorithm in the framework of Theorem 6, which

[4] In particular, looking at [HW19]'s informal discussion on page 17, the states $F_M^{-1}|S_M(\theta/\pi)\rangle$ and $F_M^{-1}|S_M(1-\theta/\pi)\rangle$ are only orthogonal when θ is sufficiently far from 0 or π.

includes several algorithms from previous works as well as the other new algorithms we present in later sections of this work.

We now outline the proof. Since a is unknown, the main idea is to split the analysis into two cases. Let κ be some threshold depending on δ and $\mathbf{D}$, and let $\bar{\kappa} := \sqrt{1-\kappa^2}$ as usual.

- Say $a \leq \kappa$ or $\bar{\kappa} \leq a$. Then the estimation algorithm is unlikely to have damaged $|\psi\rangle$ at all.
- Say $\kappa < a < \bar{\kappa}$. Then $|\psi\rangle$ can be repaired using $O(\kappa^{-1})$ rounds of amplitude amplification.

In other words, we have used the runtime $\mathbf{D}$ of the previous procedure to essentially obtain bounds on a. This is possible because we need $\mathbf{D}$ to be $\sim a^{-1}$ in order to perturb $|\psi\rangle$. We establish this fact first, and then we discuss amplitude amplification.

Amplitude estimation algorithms in the framework of Theorem 6 may appear difficult to analyze in such generality because they could sample from so many polynomials P. But we can simplify things via two observations. First, consider sampling from a semi-Pellian polynomial P that is the real part of a Pellian polynomial $\tilde{P}$. Looking at Proposition 5, the actual transition probabilities between states are determined by $\tilde{P}$, not P. So we can assume without loss of generality that we only ever sample from Pellian polynomials. Second, it turns out that, among Pellian polynomials, Chebyshev polynomials essentially maximize the probability of damaging $|\psi\rangle$ near when $a \leq \kappa$ or $\bar{\kappa} \leq a$. That means that we can assume without loss of generality that the previous algorithm only ever sampled from Chebyshev polynomials.

Our first goal is to use properties of Chebyshev polynomials to determine good choices for bounds on a, that is, to determine κ. To do so, we employ a geometrical argument involving small rotations on the Bloch sphere. Chebyshev polynomials are commonly interpreted in terms of rotation: For $|a| \leq 1$, it is a standard fact that $T_d(a) = \cos(d \arccos(a))$. However, for small a, $\arccos(a)$ is a large angle, not a small angle. We would prefer to consider rotations about $\theta := \arcsin(a)$, which is small whenever a is small. Fortunately we also have the following:

Fact 10. *Say $|x| \leq 1$. If d is odd then $|T_d(x)|^2 = |\sin(d \arcsin(x))|^2$. If d is even, then $|T_d(x)|^2 = 1 - |\sin(d \arcsin(x))|^2$*

In the previous section, we expressed Pellian polynomials in terms of alternating reflections and rotations. However, it is also possible to express them in terms of alternating rotations alone. Rephrasing part of Theorem 3 of [GSLW18], we find that if $P, Q \in \mathbb{C}[a]$ is a Pell pair, then there exist phases $\phi_1, ..., \phi_d$ such that:

$$\langle 0 | e^{i\phi_0 Z} \prod_{j=1}^{d} (W e^{i\phi_j Z}) | 0 \rangle = P \text{ where } W := \begin{bmatrix} a & i\bar{a} \\ i\bar{a} & a \end{bmatrix}. \tag{50}$$

Notice that $W = e^{iX \arccos(a)}$, but again we would like to write things in terms of $\theta := \arcsin(a)$. This is achieved by adding an extra factor of iX:

$$W = \begin{bmatrix} \sin(\theta) & i\cos(\theta) \\ i\cos(\theta) & \sin(\theta) \end{bmatrix} = \begin{bmatrix} \cos\left(\frac{\pi}{2} - \theta\right) & i\cos\left(\frac{\pi}{2} - \theta\right) \\ i\cos\left(\frac{\pi}{2} - \theta\right) & \sin\left(\frac{\pi}{2} - \theta\right) \end{bmatrix} = e^{i\left(\frac{\pi}{2} - \theta\right)X} = iXe^{i\theta X} \tag{51}$$

Now we can express both general Pellian polynomials and Chebyshev polynomials in terms of small rotations θ. This allows us to show that Chebyshev polynomials always bound Pellian polynomials near $a \approx 0$ or $a \approx 1$.

Lemma 11. ***Chebyshev polynomials extremize Pellian polynomials.*** *Say P is a Pellian polynomial of degree d, and say $a \leq \sin\left(\frac{\pi}{2d}\right)$. If d is odd, then $|P(a)| \leq |T_d(a)|$. If d is even, then $|P(a)| \geq |T_d(a)|$.*

Similarly, say $\cos\left(\frac{\pi}{2d}\right) \leq a$. Then $|P(a)| \geq |T_d(a)|$, regardless of if d is even or odd.

Proof. We begin by plugging $W = iXe^{i\theta X}$ into (50). Observe that $e^{i\phi Z}X = Xe^{-i\phi Z}$, so we can find new ϕ'_j such that:

$$P = \langle 0| e^{i\phi_0 Z} \prod_{j=1}^{d} (iXe^{i\theta X} e^{i\phi_j Z}) |0\rangle = \langle 0| (iX)^d e^{i\phi'_0 Z} \prod_{j=1}^{d} (e^{i\theta X} e^{i\phi'_j Z}) |0\rangle \tag{52}$$

Next, we repeatedly insert $e^{i\phi Z}e^{-i\phi Z}$ for various values of ϕ, and absorb these rotations into the ϕ'_j to make ϕ''_j such that:

$$P = \langle 0| (iX)^d e^{i\phi''_0 Z} \prod_{j=1}^{d} (e^{i\phi''_j Z} e^{i\theta X} e^{-i\phi''_j Z}) |0\rangle \tag{53}$$

Now we make a geometric argument involving the Bloch sphere to bound $|P|^2$. We begin with the $a \leq \sin\left(\frac{\pi}{2d}\right)$ case. Our goal is to show that if d is odd, then $|P(a)| \leq |T_d(a)|$. If d is even, then $|P(a)| \geq |T_d(a)|$.

Consider the dynamics of (53) on the Bloch sphere, as depicted in Figure 2. We begin in the $|0\rangle$ state, and repeatedly make rotations $e^{i\theta X_\phi}$ around axes $X_\phi := e^{i\phi Z}Xe^{-i\phi Z}$ that are confined to the XY-plane. Finally, $|P|^2$ is the probability of measuring $|0\rangle$ in the even case, and $|1\rangle$ in the odd case.

Say we are in the odd case, where we want to give an upper bound on $|P|^2$. This is the probability of measuring $|1\rangle$, which corresponds to how close to the south pole of the Bloch sphere we are. Each rotation $e^{i\theta X_\phi}$ moves us an arclength of θ, satisfying $\theta \leq \frac{\pi}{2d}$. While these could cancel each other out depending on the ϕ''_j, they achieve the greatest distance when they all rotate in the same direction, in which case they achieve a total arclength of $d\theta$. Since $d\theta \leq \frac{\pi}{2}$, we can never overshoot the south pole $|1\rangle$ if we do this. So the final state's amplitude on $|1\rangle$ is at most $\sin(d\theta)$, so $|P|^2 \leq \sin^2(d\theta) = |T_d(a)|^2$.

In the even case we want to give a lower bound on $|P|^2$. This is the probability of measuring $|0\rangle$, which corresponds to how close to the north pole we are. But a lower bound on $|P|^2$ can be obtained from an upper bound on $1 - |P|^2$, which measures how close to the south pole we are. But this is exactly the quantity we bounded in the previous paragraph: we have $1 - |P|^2 \leq \sin^2(d\theta)$. With some trigonometric identities this yields $|P|^2 \geq |T_d(a)|^2$.

The $\cos\left(\frac{\pi}{2d}\right) \leq a$ case is very similar. We first notice that $\bar{a} \leq \sin\left(\frac{\pi}{2d}\right)$, so we define $\bar{\theta} := \arcsin(\bar{a}) = \arccos(a)$, and observe that $W = e^{i\bar{\theta}X}$. We repeat the calculation without needing to propagate forward an iX, and arrive at the existence of some $\bar{\phi}''_j$ such that:

$$P = \langle 0| e^{i\bar{\phi}''_0 Z} \prod_{j=1}^{d} (e^{i\bar{\phi}''_j Z} e^{i\bar{\theta}X} e^{-i\bar{\phi}''_j Z}) |0\rangle \tag{54}$$

So, regardless of the parity of d, we arrive in a similar situation: we start at the north pole of the Bloch sphere with $|0\rangle$, apply d small rotations of angle $\bar{\theta} \leq \frac{\pi}{2d}$ around various axes $X_{\bar{\phi}''_k}$. Afterwards, $|P|^2$ is the probability that we measure $|0\rangle$, or equivalently $1-|P|^2$ is the probability that we measure $|1\rangle$. As we argued before we have $1-|P|^2 \leq \sin^2(d\bar{\theta})$, so we have $|P|^2 \geq |T_n(a)|$.

□

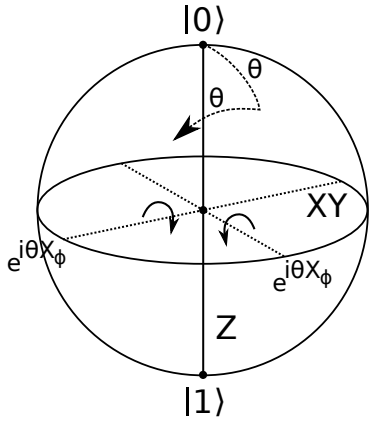


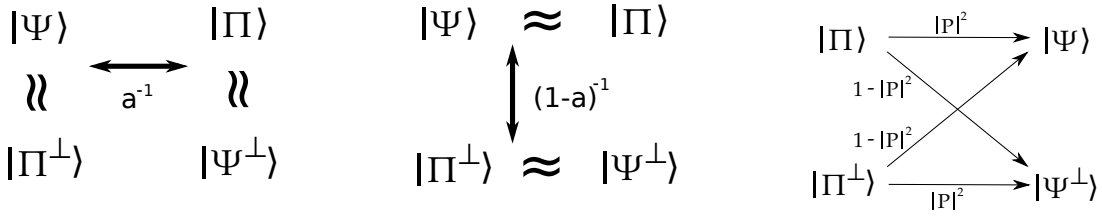
Figure 2: Visualization of the proof of Lemma 11. In particular, consider (53). The quantum state starts at $|0\rangle$, and is then acted on by several rotations $e^{i\theta X_\phi}$ around axes $X_\phi := e^{i\phi Z} X e^{-i\phi Z}$. The arclength is always θ, and the axes are confined to the XY-plane. We see that the furthest angle we could achieve after d many such rotations is $d\theta$.

We have demonstrated that for sufficiently extreme a (that is, $a \leq \kappa$ or $\bar{\kappa} \leq a$), we can consider Chebyshev polynomials without loss of generality. We want to argue that for extreme a, the probability that we damage $|\psi\rangle$ is very small. However, recalling Figure 1, sampling from an odd polynomial essentially forces us to obtain a copy of $|\Pi\rangle$ or $|\Pi^\perp\rangle$, therefore unavoidably damaging $|\psi\rangle$. We argue that this is fine, since for extreme a, one of these is always very close to $|\psi\rangle$ and the other is very far from $|\psi\rangle$. In particular, if $a \leq \kappa$, then as we sample from odd polynomials we will just bounce back and forth between $|\psi\rangle$ and $|\Pi^\perp\rangle$ with high probability. Figure 3 summarizes the two cases.

Proposition 12. ***For extreme*** a***, we are unlikely to damage*** $|\psi\rangle$*. Say we just ran an amplitude estimation algorithm such that the sum of the degrees of all the sampled polynomials is* $\mathbf{D}$*. Then, the following holds for any* $\delta > 0$*:*

- *If* $a \leq \sin(\sqrt{\delta}/\mathbf{D})$*, then the algorithm returns a copy of* $|\psi\rangle$ *or* $|\Pi^\perp\rangle$ *with probability* $\geq 1-\delta$*.*
- *If* $a \geq \cos(\sqrt{\delta}/\mathbf{D})$*, then the algorithm returns a copy of* $|\psi\rangle$ *or* $|\Pi\rangle$ *with probability* $\geq 1-\delta$*.*

Proof. For the purposes of transitions between states $\{|\psi\rangle, |\psi^\perp\rangle, |\Pi\rangle, |\Pi^\perp\rangle\}$ we can without loss of generality assume that the algorithm sampled from Pellian polynomials only. This is because sampling from a semi-Pellian polynomial P entails finding a Pellian polynomial $\tilde{P}$



(a) a is small: $a \le \kappa$ (b) a is large: $\bar{\kappa} \le a$ (c) Reprint of Figure 1b

Figure 3: For extreme a, the states $|\psi\rangle, |\psi^\perp\rangle, |\Pi\rangle, |\Pi^\perp\rangle$ group into two pairs. The states in each pair are very close together, and the two pairs are very far apart. That means that if we sample from an odd polynomial, then unless that polynomial has very high degree we are going to just bounce back and forth between $\{|\psi\rangle, |\Pi^\perp\rangle\}$ for $a \le \kappa$, or between $\{|\psi\rangle, |\Pi\rangle\}$ when $a \ge \bar{\kappa}$.

such that $\mathrm{Re}(\tilde{P}) = P$, and the transitions between states are determined by $\tilde{P}$ in exactly the same way (see Proposition 5).

While in general $\mathbf{D}$ is a random variable, the randomness does not really matter for this proof so we drop the boldface and just write D. We begin with the $a \le \sin(\sqrt{\delta}/D)$ case. Looking at Figure 1, we will show that with probability $\ge 1-\delta$, we only see the transitions $|\psi\rangle \to |\Pi^\perp\rangle$, $|\Pi^\perp\rangle \to |\psi\rangle$, $|\psi\rangle \to |\psi\rangle$ or $|\Pi^\perp\rangle \to |\Pi^\perp\rangle$. That way, we can only finish with $|\psi\rangle$ or $|\Pi^\perp\rangle$.

Say we sampled from m polynomials total, and the j'th polynomial had degree d_j. That way $\sum_{j=1}^m d_j = D$. If the j'th polynomial P_j is odd, we must show that the $|\psi\rangle \to |\Pi\rangle$ and $|\Pi^\perp\rangle \to |\psi^\perp\rangle$ transitions are unlikely. These each occur with probability $|P_j|^2$. Since $\arcsin(a) \le \sqrt{\delta}/D$, we can invoke Lemma 11 to obtain $|P_j(a)| \le |T_{d_j}(a)|$:

$$(\text{odd } d_j) \;\to\; |P_j(a)|^2 \le |T_{d_j}(a)|^2 \le |\sin(d_j \arcsin(a))|^2 \le |\sin(\sqrt{\delta}\cdot d_j/D)|^2 \le \delta \cdot d_j^2/D^2 \tag{55}$$

If the j'th polynomial P_j has even degree d_j, then we must show that the $|\psi\rangle \to |\psi^\perp\rangle$ and $|\Pi^\perp\rangle \to |\Pi\rangle$ transitions are unlikely. These occur with probability $1-|P_j|^2$. Lemma 11 gives us $|P_j(a)| \ge |T_{d_j}(a)|$.

$$(\text{even } d_j) \;\to\; 1-|P_j(a)|^2 \le 1-|T_{d_j}(a)|^2 \le |\sin(d_j \arcsin(a))|^2 \le \delta \cdot d_j^2/D^2 \tag{56}$$

Finally, we observe that $\sum_j d_j^2 \le \sum_{j,k} d_j d_k = D^2$ to complete the proof with a union bound:

$$\Pr[\text{ever create } |\psi^\perp\rangle, |\Pi^\perp\rangle] \le \sum_{j=1}^m \delta \cdot d_j^2/D^2 \le \delta. \tag{57}$$

Now we consider $a \ge \cos(\sqrt{\delta}/D)$, which implies $\arccos(a) \le \sqrt{\delta}/D$. In this case we want to demonstrate that we only ever see the transitions $|\psi\rangle \to |\Pi\rangle$, $|\Pi\rangle \to |\psi\rangle$, $|\psi\rangle \to |\psi\rangle$ or $|\Pi\rangle \to |\Pi\rangle$, so we finish only with $|\psi\rangle$ or with $|\Pi\rangle$. Since $a \ge \cos(\sqrt{\delta}/D)$, Lemma 11 gives us $|P_j(a)| \ge |T_{d_j}(a)|$ regardless of d_j.

If d_j is odd, then we want to show that the $|\psi\rangle \to |\Pi^\perp\rangle$ and $|\Pi\rangle \to |\psi^\perp\rangle$ transitions are unlikely. These each occur with probability $1 - |P_j|^2$, so:

$$(\text{odd } d_j) \to \ 1 - |P_j(a)|^2 \leq 1 - |T_{d_j}(a)|^2 = 1 - |\cos(d_j \arccos(a))|^2 \tag{58}$$

$$= |\sin(d_j \arccos(a))|^2 \leq |\sin(\sqrt{\delta} \cdot d_j/D)|^2 \leq \delta \cdot d_j^2/D \tag{59}$$

Similarly, if d_j is even then we want the $|\psi\rangle \to |\psi^\perp\rangle$ and $|\Pi\rangle \to |\Pi^\perp\rangle$ transitions to be unlikely. These occur with probability $1 - |P_j|^2$, so:

$$(\text{even } d_j) \ \to \ 1 - |P_j(a)|^2 \leq 1 - |T_{d_j}(a)|^2 \leq |\sin(d_j \arccos(a))|^2 \leq \delta \cdot d_j^2/D^2. \tag{60}$$

The same union bound argument shows that we never create $|\psi^\perp\rangle, |\Pi\rangle$ with probability $\geq 1 - \delta$. □

This completes the first part of the argument: if a is sufficiently close to 0 or 1 then it is probably not necessary to repair the state at all, since we either already have $|\psi\rangle$ or something very close to it. Next, we simply turn this argument on its head: say we do not have $|\psi\rangle$ or something close to it. Then a is probably far from 0 or 1, meaning it is easy to repair using amplitude amplification.

We just ran an amplitude estimation algorithm, so we probably have a decent estimate of a. This means we could just use that estimate to inform a very standard amplitude estimation protocol based on Grover rotations only. However, part of the strength of this section's state repair algorithm is that it can be appended to *any* amplitude estimation algorithm, including those that give pretty weak guarantees on the quality of the final estimation. So, we will stick to our polynomial sampling framework and construct an algorithm for fixed-point amplitude estimation within it. We can construct Pellian polynomials that, when sampled from, will essentially force the desired transitions in Figure 1b, reprinted for convenience in Figure 3c. It turns out that this method is extremely similar to [YLC14].

To construct the desired polynomials, we require a characterization of Pell-complementary polynomials from Theorem 3 of [GSLW18] we know that a polynomial $Q \in \mathbb{C}[x]$ is Pell-complementary if and only if:

- $\forall x \in [-1, 1]$, we have $\bar{x}|Q(x)| \leq 1$
- If Q is even, then $\forall x \in \mathbb{R}$ we have $(1 + x^2)Q(ix)Q^*(ix) \geq 1$.

Next, we will leverage this fact to construct the two polynomials we need. We will need two: we sample from $J(a)$ when $a \leq \bar{\kappa}$, and we will sample from $K(a)$ when $\kappa \leq a$. The polynomials are plotted in Figure 4. The construction of $J(a)$ already appears in the literature due to its applications of fixed-point Grover search, but the construction of $K(a)$ is novel as far as we know.

Lemma 13. ***Polynomials for fixed-point amplitude amplification.*** *Take any $\kappa, \eta \in (0, 1)$, and let $\bar{\kappa} := \sqrt{1 - \kappa^2}$. There exists an odd Pellian polynomial $J(x)$ satisfying $|J(x)|^2 < \eta$ for $|x| < \bar{\kappa}$. There also exists an odd Pellian polynomial $K(x)$ satisfying $|K(x)|^2 > 1 - \eta$ for $|x| > \kappa$. The degree of these polynomials is the smallest odd number $\geq \kappa^{-1} \ln\left(2/\sqrt{\eta}\right)$.*

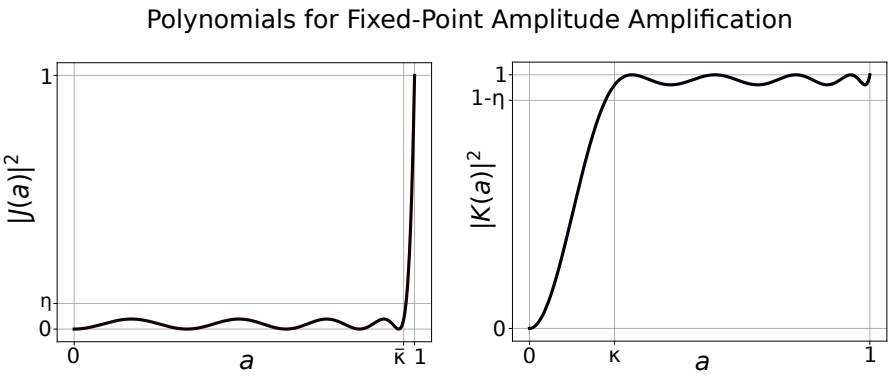


Figure 4: Polynomials constructed in Lemma 13 and used in Proposition 14. Here we selected $\eta = 0.1$ and $\kappa = 0.25$, which makes these polynomials have degree $l = 9$.

Proof. For the construction of $J(x)$ we follow Lemma 4.1 in [AKLV13], but note that [YLC14] do something similar. For some odd $l \in \mathbb{Z}^+$ and some $\gamma \in [0, 1]$, let:

$$J(x) := \frac{T_l(x/\gamma)}{T_l(1/\gamma)} \tag{61}$$

First, we show that for any such l, γ, there is a polynomial Q such that J, Q form a Pell pair because J satisfies the first set of constraints of Theorem 3 of [GSLW18]. Chebyshev polynomials $T_l(x)$ are already Pellian. Since $1/\gamma \geq 1$, we have $1/|T_l(1/\gamma)| \leq 1$. Then, for $x/\gamma \leq 1$ we have $|T_l(x/\gamma)| \leq 1$, so $|J(x)| \leq 1$. As for $1 \leq x/\gamma$, we utilize the fact that $|T_l(x/\gamma)|$ is increasing. That means for $x \leq 1$ we have $|T_l(x/\gamma)| \leq |T_l(1/\gamma)|$, so we also have $|J(x)| \leq 1$. But for $x \geq 1$ we have $|T_l(x/\gamma)| \geq |T_l(1/\gamma)|$, so $|J(x)| \geq 1$.

Second, we show how to actually select l, γ such that $|J(x)|^2 \leq \eta$ for $|x| \leq \bar{\kappa}$. Certainly for $|x| \leq \gamma$ we have $|T_l(x/\gamma)| \leq 1$, so therefore $|J(x)| \leq 1/|T_l(1/\gamma)|$. We select $\gamma := \bar{\kappa}$ and pick the minimum odd l such that $1/|T_l(\gamma^{-1})| \leq \sqrt{\eta}$. We have $T_l(\gamma^{-1}) = \cosh(l\text{arccosh}(\gamma^{-1})) \geq \frac{1}{2}\exp(l\text{arccosh}(\gamma^{-1}))$. Also, we have:

$$\text{arccosh}(\gamma^{-1}) \geq 2\tanh(\text{arccosh}(\gamma^{-1})/2) \geq 2\sqrt{\frac{\gamma^{-1}-1}{\gamma^{-1}+1}} \tag{62}$$

We desire $1/|T_l(\gamma^{-1})| \leq \sqrt{\eta}$, or rather $|T_l(\gamma^{-1})| \geq 1/\sqrt{\eta}$, which is ensured when:

$$\frac{1}{2}\exp\left(2l\sqrt{\frac{\gamma^{-1}-1}{\gamma^{-1}+1}}\right) \geq \frac{1}{\sqrt{\eta}} \tag{63}$$

With $\gamma := \bar{\kappa}$, we satisfy the above if l satisfies:

$$l \geq \frac{1}{2}\ln\left(\frac{2}{\sqrt{\eta}}\right)\sqrt{\frac{\bar{\kappa}^{-1}+1}{\bar{\kappa}^{-1}-1}} = \frac{1}{2}\ln\left(\frac{2}{\sqrt{\eta}}\right)\sqrt{\frac{(1+\bar{\kappa})^2}{1-\bar{\kappa}^2}} \tag{64}$$

which in turn is implied by $l \geq \frac{1}{\bar{\kappa}}\ln\left(\frac{2}{\sqrt{\eta}}\right)$, seeing as $\bar{\kappa} < 1$.

We move on to the construction of the Pellian polynomial $K(x)$ satisfying $|K(x)|^2 \geq 1-\eta$ for $|x| \geq \kappa$, using $J(x)$ as a starting point. Indeed, the desired properties follow from:

$$|K(x)|^2 = 1 - |J(\bar{x})|^2 \tag{65}$$

We obtain K from the Pell-complementarity characterization of Theorem 3 from [GSLW18]: we show that there is an even polynomial $E(x)$ such that $J(\bar{x}) = \bar{x}E(x)$, and that $E(x)$ satisfies the conditions in the lemma.

$J(\bar{x})$ is an odd polynomial in $\bar{x}$, so $J(\bar{x})/\bar{x}$ is an even polynomial in $\bar{x}$. Even polynomials in $\bar{x}$ are really polynomials in $\bar{x}^2 = 1 - x^2$, so $E(x) := J(\bar{x})/\bar{x}$ is an even polynomial in x as well. By construction $J(\bar{x}) = \bar{x}E(x)$.

The first property $\bar{x}|E(x)| \leq 1$ for $|x| \leq 1$ is guaranteed by the fact that $|J(\bar{x})| \leq 1$ for $|x| \leq 1$. Since $E(x)$ is even, we also need to show the second property that $(1+x^2)E(ix)E^*(ix) \geq 1$ for all real x. Since the coefficients of $J(x)$ are real, the coefficients of $E(x)$ are also real, so $E^*(ix) = E(ix)$. So it is sufficient to show $(1+x^2)E(ix)^2 \geq 1$.

$$(1+x^2)E(ix)^2 = J(\sqrt{1-(ix)^2})^2 \frac{1+x^2}{1-(ix)^2} = J(\sqrt{1+x^2}) \geq 1 \tag{66}$$

Since $\sqrt{1+x^2} \geq 1$ for all real x, and $J(x) \geq 1$ for all positive x, we have shown the second condition for E. Thus, by Theorem 3 of [GSLW18] E is Pell-complementary, so there exists an odd polynomial $K(x)$ such that K and E form a Pell pair. We have $|K(x)|^2 = 1-\bar{x}^2|E(x)|^2 = 1 - |J(\bar{x})|^2$, so $|K(x)|^2 \geq 1-\eta$ for $|x| \geq \kappa$ as desired. □

These polynomials immediately yield an algorithm for amplitude amplification. At this point we finally identify $\kappa := (4/5)\cdot\sqrt{\delta}/D$ in order to ensure $\kappa \leq \sin(\sqrt{\delta}/D)$ and $\cos(\sqrt{\delta}/D) \leq \bar{\kappa}$.

Proposition 14. ***Obtaining*** $|\psi\rangle$ ***from*** $|\Pi\rangle$ ***or*** $|\Pi^\perp\rangle$**.** *Suppose $a < \cos(\sqrt{\delta}/D)$, and we have a copy of $|\Pi^\perp\rangle$. Then, for any $\eta > 0$ we can prepare a copy of $|\psi\rangle$ with success probability $\geq 1-\eta$ by sampling from a polynomial of degree $\lceil (5/4)(D/\sqrt{\delta})\ln(2/\sqrt{\eta})\rceil$.*

Similarly, suppose $\sin(\sqrt{\delta}/D) < a$, and say we have a copy of $|\Pi\rangle$. Then, we can prepare $|\psi\rangle$ with success probability $\geq 1-\eta$ by sampling from another polynomial with the same degree.

Proof. Figure 3c shows the transition probabilities when sampling from an odd polynomial starting with $|\Pi\rangle$ or $|\Pi^\perp\rangle$.

Given $|\Pi^\perp\rangle$ and $a < \cos(\sqrt{\delta}/D)$, our goal is to implement the $|\Pi^\perp\rangle \to |\psi\rangle$ transition with probability $\geq 1-\eta$. If we implement the transition with an odd Pellian polynomial J, then this transition happens with probability $1 - |J(a)|^2$. So our goal is to find a $J(a)$ satisfying $|J(a)|^2 \leq \eta$ for $a < \cos(\sqrt{\delta}/D)$. The polynomial $J(x)$ from Lemma 13 exactly has this property when $\cos(\sqrt{\delta}/D) \leq \bar{\kappa}$. We pick $\kappa := (4/5)\cdot\sqrt{\delta}/D$. Since $0.8x \leq \sin(x)$ for $x \in [0,1]$, we have $\kappa \leq \sin(\sqrt{\delta}/D)$ which implies the desired bound.

If we are given $|\Pi\rangle$ and $a \geq \sin(\sqrt{\delta}/D)$, then can implement the transition $|\Pi\rangle \to \psi$ by sampling from an odd Pellian polynomial with K with success probability $|K(a)|^2$. The polynomial $K(x)$ from Lemma 13 satisfies $|K(a)|^2 \geq 1-\eta$ for the same choice of κ. □

Finally, we combine Propositions 12 and 14 to prove the main result of this section. The main challenge of this algorithm is that we do not actually know the value of a, and moreover

never obtain conclusive evidence that $\sin(\sqrt{\delta}/\mathbf{D}) < a$ or $a < \cos(\sqrt{\delta}/\mathbf{D})$. All we have is the output state of the previous amplitude estimation algorithm and its total degree $\mathbf{D}$. Given this information we essentially *guess* that one of the bounds on a hold.

Theorem 15. ***Non-destructive amplitude estimation.*** *Take any amplitude estimation algorithm that produces an estimate* $\hat{\mathbf{a}}$ *and has total degree* $\mathbf{D}$*. For any* $\mu > 0$ *there exists another amplitude estimation algorithm that produces the same* $\hat{\mathbf{a}}$*, outputs a copy of* $|\psi\rangle$ *at then end with probability* $\geq 1 - \delta$*, and has total degree at most* $\mathbf{D} + \lceil (5/4)(\mathbf{D}/\sqrt{\delta}) \ln(2/\sqrt{\eta}) \rceil$ *where* $\delta = (4/5) \cdot \mu$ *and* $\eta = (1/5) \cdot \mu$*.*

Proof. The new algorithm is just the old algorithm plus an extra repair step. We first run the old algorithm, which gives our final estimate $\hat{\mathbf{a}}$ in some total degree $\mathbf{D}$, as well as one of $|\psi\rangle, |\psi^{\perp}\rangle, |\Pi\rangle, |\Pi^{\perp}\rangle$. Then:

- If we have $|\psi\rangle$, then there is nothing to do, and we successfully return $|\psi\rangle$.
- If we have $|\psi^{\perp}\rangle$, then we measure in the $|\Pi\rangle, |\Pi^{\perp}\rangle$ basis (i.e., sample from $P(a) = a$) and proceed according to the next two cases.
- If we have $|\Pi^{\perp}\rangle$, then we guess that $a \leq \sin(\sqrt{\delta}/\mathbf{D})$ and sample from the polynomial $J(x)$ from Proposition 14. If we obtain $|\psi\rangle$, then we have succeeded and we are done. If we obtain $|\psi^{\perp}\rangle$, then we have failed and we give up.
- Finally, if we have $|\Pi\rangle$, then we guess that $a \geq \cos(\sqrt{\delta}/\mathbf{D})$ and sample from the polynomial $K(x)$ from Proposition 14. Similarly, we return $|\psi\rangle$ if we obtain it, and otherwise we give up.

We want to show that the probability that we fail and give up is at most μ. We split the analysis into three cases depending on a, although we do not need know know which case we are in in order to run the procedure above.

Say $a \leq \sin(\sqrt{\delta}/\mathbf{D})$. Then, by Proposition 12, we have a copy of $|\psi\rangle$ or $|\Pi^{\perp}\rangle$ with probability at least $1 - \delta$. If we have $|\psi\rangle$ we are done, and if we have $|\Pi^{\perp}\rangle$ then we sample from $J(a)$ which returns a copy of $|\psi\rangle$ with probability $\geq 1 - \eta$. So overall we fail with probability at most $\delta + \eta = \mu$.

Say $a \geq \cos(\sqrt{\delta}/\mathbf{D})$. Then, we have either $|\psi\rangle$ or $|\Pi\rangle$ with probability $\geq 1 - \delta$. If we have $|\Pi\rangle$ then we sample from $K(a)$ and obtain $|\psi\rangle$ with probability $\geq 1 - \eta$. So again we fail with probability at most $\delta + \eta$.

Finally, say $\sin(\sqrt{\delta}/\mathbf{D}) \leq a \leq \cos(\sqrt{\delta}/\mathbf{D})$. Then we could get any of $|\psi\rangle, |\psi^{\perp}\rangle, |\Pi\rangle, |\Pi^{\perp}\rangle$. As usual if we have $|\psi\rangle$ we are done, and if we have $|\psi^{\perp}\rangle$ then we turn it into $|\Pi\rangle$ or $|\Pi^{\perp}\rangle$. the condition on a satisfies *both* of the required assumptions in Proposition 14, so the $|\Pi\rangle \to |\psi\rangle$ and $|\Pi^{\perp}\rangle \to |\psi\rangle$ both succeed with probability $\geq 1 - \eta$. So we fail with probability at most $\eta < \mu$.

We numerically find the choice $\delta = (4/5) \cdot \mu$ and $\eta = (1/5) \cdot \mu$ is reasonably close to minimizing $(1/\sqrt{\delta}) \ln(2/\sqrt{\eta})$. □

In practice, it may be advantageous to actually use the information on a obtained from the previous amplitude estimation algorithm in order to reduce the overall complexity. In theory, however, we have demonstrated that this is not really necessary and that state repair is always possible regardless of the actual value of a.

We conclude the section with some additional remarks about this result.

Remark 16. ***Always returning*** $|\psi\rangle$***, even if it takes forever.*** *The modified algorithm from Theorem 15 is a Monte Carlo algorithm: it only adds at most a fixed number of additional queries, but does not always succeed at preparing* $|\psi\rangle$*. But it can easily be transformed into a Las Vegas algorithm: First, observe that when* $a = 0$ *or* $a = 1$*, then sampling from a polynomial can never produce anything other than* $|\psi\rangle$*. Otherwise, if* $0 < a < 1$ *then we consider the following protocol. If we have* $|\psi^{\perp}\rangle$ *then measure in the* $|\Pi\rangle, |\Pi^{\perp}\rangle$ *basis. If we have* $|\Pi\rangle$ *or* $|\Pi^{\perp}\rangle$*, measure in the* $|\psi\rangle, |\psi^{\perp}\rangle$ *basis. Since all measurement outcomes have nonzero probability, this protocol must reach* $|\psi\rangle$ *eventually. Now we have Las Vegas algorithm that will eventually produce* $|\psi\rangle$ *with certainty, but can only inherit the performance guarantees of the original algorithm with probability* $\geq 1 - \mu$*.*

Remark 17. ***State repair via sampling given bounds on*** a***.*** *Say we are in the same situation as [HW19], where we are given a bound* $\kappa < a < \bar{\kappa}$*. Then we can also achieve state repair with only* $O(\kappa^{-1} \log(\delta^{-1}))$ *queries by sampling from another polynomial. To do so, we follow Theorem 15, but instead of using the bounds* $\sin(\sqrt{\delta}/D) < a < \cos(\sqrt{\delta}/D)$ *to determine* κ*, we just use the* κ *we were given.*

3 Improved Performance

The asymptotic performance of amplitude estimation has been known for some time. However, for those interested in forecasting the resource requirements of quantum algorithms built on amplitude estimation, constant-factor improvements in query complexity could make a large difference in the time-scale on which certain quantum applications may be practically realizable. To this end, many competing implementations of amplitude estimation have emerged in the last few years with differing empirical query complexity.

The first recent attempt at improving the constant-factor performance of amplitude estimation was [Suzuki&19], which describes an algorithm called Maximum Likelihood Amplitude Estimation (MLAE). There is sound empirical evidence that this algorithm significantly improves over the performance of [BHMT00] and numerics indicate that it achieves $O(1/\varepsilon)$ scaling. However, there is no rigorous proof that this algorithm always delivers an accurate answer and the desired performance.

Later, [GGZW19] gave an algorithm called Iterative Quantum Amplitude Estimation (IQAE) that seems to have the best of both worlds. Its empirical performance also outperforms [BHMT00] and seems comparable to that of MLAE, but it also has a rigorous proof of correctness. The numerics supporting the speedup have been independently reproduced [YLRJ20]. The proofs in [GGZW19] even offer bounds on the constant-factor performance, but these bounds are not strong enough to rigorously prove outperforming [BHMT00]. As the desired precision grows, the likelihood-landscape of which MLAE needs to find the maximum becomes more and more complicated, requiring more and more classical resources to analyze. IQAE also avoids this problem with a simpler classical component to the algorithm.

Are there any amplitude estimation algorithms that improve over the constant-factor performance of IQAE while simultaneously featuring a rigorous proof? [Nakaji20] claims to present a modified version of IQAE with better query complexity, but the experiments in the manuscript do not support this claim[5]. Further, [Zhao&22] gave an algorithm where the

[5]Compare Figure 3 in [Nakaji20] to Figure 3 in [GGZW19]. Recent work [Zhao&22] also makes this obser-

classical processing is significantly faster then IQAE, but the query complexity is about the same.

In this section we present a modified version of IQAE called 'ChebAE' that we empirically demonstrate to require only about 50% to 60% of the queries of IQAE. Since it is a fairly simple modification, it inherits the proof of correctness presented in [GGZW19]. ChebAE was discovered in an attempt to improve the performance of IQAE using the polynomial sampling framework in Theorem 6. Recall that IQAE is a special case of such algorithms where only odd-degree Chebyshev polynomials are considered. Does access to a larger family of polynomials allow us to improve the query complexity in terms of constant factors? We were not able to devise an algorithm that answers this question in the affirmative because it seems that the Chebyshev polynomials deliver the best performance. However, in our efforts to perform hyperparameter optimization, we discovered a faster algorithm that is based on Chebyshev polynomials only. The algorithm also has the property that it requires a smaller query complexity when $a \approx 1$.

We now give a brief description of IQAE in terms of the polynomial sampling framework in order to point out what optimization was made. IQAE is inspired by [AR19] in that it gradually improves a confidence interval $[a_{\min}, a_{\max}]$ that contains a with high probability. The method for improving this confidence interval is sketched in Figure 5. First, we find the largest odd integer d such that $|T_d(a)|^2$ is invertible on $[a_{\min}, a_{\max}]$. Then, we sample from T_d in order to obtain a confidence interval $[p_{\min}, p_{\max}]$ containing $|T_d(a)|^2$ with high probability. For this purpose we can rely on the Clopper-Pearson method [CP34] which is based on tight bounds on the binomial distribution. Then, since $|T_d(a)|^2$ is invertible, we can compute a new interval $[a^*_{\min}, a^*_{\max}]$ where (if $|T_d(a)|^2$ is increasing on the interval):

$$|T_d(a^*_{\min})|^2 = p_{\min} \text{ and } |T_d(a^*_{\max})|^2 = p_{\max} \tag{67}$$

If $|T_d(a)|^2$ is decreasing on the interval, then we just swap $p_{\min}$ and $p_{\max}$ in the equations above. We repeat this process until the confidence interval is as small as desired, or until a better Chebyshev polynomial can be found.

In order to guarantee a bound δ on the failure probability, we anticipate the total number of confidence intervals $[p_{\min}, p_{\max}]$ required and divide the failure probability evenly among them. If we demand that the degree d increases by at least a factor of r at every iteration, then the confidence interval must also shrink by a factor of r. Thus, at most $\log_r(1/\varepsilon)$ iterations are required. With this method, the final asymptotic complexity is $O(\varepsilon^{-1}\log(\delta^{-1}\log(\varepsilon^{-1})))$. [AR19] gave a method to remove this extra $\log(\log(\varepsilon^{-1}))$ factor, but this method blows up the constant factor on the query complexity.

How many samples from T_d should we take at every step? We need to take enough samples in order to shrink the confidence interval by a factor of r. But since the shape of $|T_d(a)|^2$ on $[a_{\min}, a_{\max}]$ is non-linear and the width of the Clopper-Pearson confidence interval depends on the value of $|T_d(a)|^2$ itself, this exact number is hard to anticipate. The best method would be to take samples from T_d one at a time until the confidence interval is small enough so that a larger d can be found. But [GGZW19] found that this is prohibitively slow in terms of classical performance. Observing that the query complexity is dominated by the final iterations of the algorithm, they introduced a parameter $N_{\text{shots}} = 100$ - the number of samples for the 'early' iterations. Once we get to the 'later' iterations we take fewer samples at a time.

vation.

This is where our optimization comes in: how do we decide what iterations are 'early' and what iterations are 'late'? IQAE leverages a heuristic based on the maximum error of any Clopper-Pearson confidence interval as well as the assumption that $|T_d(a)|^2$ is linear on the interval. We found that the cutoff from this heuristic is too late, causing IQAE to take more samples than needed. Instead, we introduce a hyperparameter ν that determines when we switch from sampling N_{shots} to 1. When we set $\nu = 8$, then ChebAE needs only about 50% to 60% of the queries of IQAE.

Having established context, we briefly digress to give intuition as to why Chebyshev polynomials appear to deliver better performance over the broader class of Pellian polynomials. Optimizing Pellian polynomials introduces several computational overheads relative to using only Chebyshev polynomials. Classical evaluation of a Pellian polynomial takes $\mathcal{O}(d)$ time where Chebyshev polynomials can be evaluated in constant time. This can have a large impact on the efficiency of choosing a new polynomial once the confidence interval has been updated. Additionally, checking whether a Chebyshev polynomial is invertible can be done exactly, but for Pellian polynomials one must check for extrema within the target interval via binary search. Failure to detect an extrema within the target interval results in an increased failure probability. Finally, the optimization of QSP angles is a highly non-convex problem which must be tuned carefully. Given that nearly optimal Chebyshev polynomials can be found inexpensively, the potential benefit of using Pellian polynomials is overshadowed by the associated overhead and reduced algorithmic stability.

ChebAE does add a minor additional flexibility over IQAE: rather than considering only odd-degree Chebyshev polynomials, ChebAE considers Chebyshev polynomials of any degree. But we do not find that this improves performance much. Nonetheless, we find that phrasing the algorithm in terms of Chebyshev polynomials significantly declutters the algorithm compared to IQAE. We also find that our implementation of ChebAE is faster to study classically than the implementation of IQAE that was kindly provided by the authors of [GGZW19]. Our numerical experiments of IQAE were based on the same implementation as the one used for [GGZW19]. Although the modification of IQAE underlying ChebAE is rather slight, we believe that the ChebAE's simpler presentation as well as significant performance improvement merit a full presentation of the algorithm here.

Empirical Claim 18. *There is an amplitude estimation algorithm 'ChebAE' that for any $\varepsilon, \delta > 0$ samples from a random variable $\hat{\mathbf{a}}$ satisfying $Pr[|\hat{\mathbf{a}} - a| \geq \varepsilon] \leq \delta$. For $a = 0.5$, $\delta = 0.05$, and $\varepsilon \in [10^{-3}, 10^{-6}]$, the observed average query complexity $\langle Q_\Pi \rangle$ satisfies[6]:*

$$\langle Q_\Pi \rangle \approx_{3.15\%} \frac{1.71}{\varepsilon} \ln\left(2.08 \ln\left(\frac{1}{\varepsilon}\right)\right) \tag{68}$$

Proof. The ChebAE algorithm relies on a subroutine FIND_NEXT_CHEB which we present after the description of the main algorithm. It also has three hyperparameters r, N_{shots}, ν: r is the factor by which we grow the degree of the Chebyshev polynomial at each iteration, N_{shots} is the number of samples from T_d we take at each iteration in the 'early' phase of the algorithm, and ν determines when we switch to the 'late' phase of the algorithm. We find the best values of these parameters are:

$$r = 2, \qquad N_{\text{shots}} = 100, \qquad \nu = 8 \tag{69}$$

[6]When we write $A \approx_\eta B$, we mean $|1 - A/B| \leq \eta$. So for $\eta = 4\%$, the smallest values of $\langle Q_\Pi \rangle$ are about $0.96\times$ the prediction, and the largest are about $1.04\times$ the prediction.

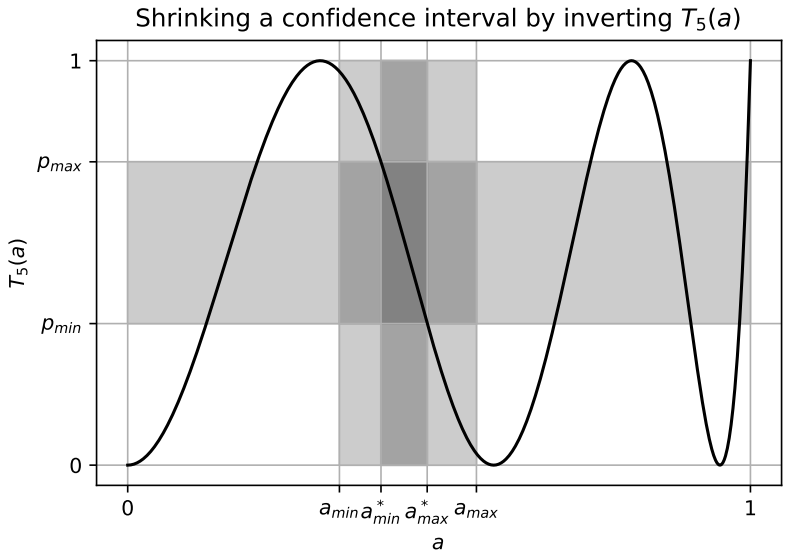


Figure 5: Say we had established that $a \in [a_{\min}, a_{\max}] = [0.34, 0.56]$. On this interval, $|T_5(a)|^2$ is invertible. By sampling from the polynomial we obtain a confidence interval $|T_5(a)|^2 \in [p_{\min}, p_{\max}] = [0.35, 0.75]$. Since $T_5(a)$ is invertible on $[a_{\min}, a_{\max}]$, we can invert $[p_{\min}, p_{\max}]$ to obtain an improved confidence interval $a \in [a^*_{\min}, a^*_{\max}] = [0.405, 0.481]$. Both IQAE and ChebAE repeat this step for various $T_d(a)$ until the desired accuracy is achieved.

The ChebAE algorithm is as follows.

1. Let $T := \lceil \log_r((2\varepsilon)^{-1}) \rceil$ be a bound on the number of confidence intervals needed.
2. Let $\varepsilon^p_{\max}$ be the largest possible error on the estimate of the bias of a coin guaranteed by the Clopper Pearson method with N_{shots} flips and confidence $1 - \delta/T$.
3. Initialize the confidence interval $[a_{\min}, a_{\max}] \leftarrow [0, 1]$ and the coin toss tally $n_{\text{heads}}, n_{\text{flips}} \leftarrow 0, 0$. Initialize the degree $d \leftarrow 1$.
4. While $a_{\max} - a_{\min} \geq 2\varepsilon$ do:
   (a) Call FIND_NEXT_CHEB$(a_{\min}, a_{\max})$ to try to find a new degree d_{new}. If $d_{\text{new}} \geq rd$, reset the tally $n_{\text{heads}}, n_{\text{flips}} \leftarrow 0, 0$ and set $d \leftarrow d_{\text{new}}$.
   (b) If the following condition holds then we are 'late', otherwise we are 'early'.
   $$\varepsilon^p_{\max} \cdot \frac{a_{\max} - a_{\min}}{|T_d(a_{\max}) - T_d(a_{\min})|} \leq \varepsilon\nu \tag{70}$$
   (c) If we are 'early', sample N_{shots} many times from T_d. If we are late, sample from T_d once. Increment the tally $n_{\text{heads}}, n_{\text{flips}}$ accordingly.
   (d) Compute a δ/T confidence interval $[p_{\min}, p_{\max}]$ on $|T_d(a)|^2$ using the Clopper-Pearson method [CP34].
   (e) Invert $[p_{\min}, p_{\max}]$ to $[a^*_{\min}, a^*_{\max}]$ using the method in Figure 5.

(f) Update the interval: $[a_{\min}, a_{\max}] \leftarrow [a^*_{\min}, a^*_{\max}] \cap [a_{\min}, a_{\max}]$.

5. Let $\hat{a}$ be the midpoint of $[a_{\min}, a_{\max}]$.

The purpose of the FIND_NEXT_CHEB$(a_{\min}, a_{\max})$ subroutine is to find the highest d such that $|T_d(a)|^2$ is invertible on the interval $[a_{\min}, a_{\max}]$. The method is based on the identity $T_d(a) = \cos(d \arccos(a))$.

1. Let $[\theta_{\min}, \theta_{\max}] := [\arccos(a_{\max}), \arccos(a_{\min})]$
2. Initialize $d \leftarrow \lfloor \frac{\pi}{2}(\theta_{\max} - \theta_{\min})^{-1} \rfloor$.
3. Decrement d until $\cos^2(d\theta)$ has no extrema for $\theta \in [\theta_{\min}, \theta_{\max}]$. This is equivalent to $\left\lfloor \frac{2}{\pi} d\theta_{\min} \right\rfloor = \left\lfloor \frac{2}{\pi} d\theta_{\max} \right\rfloor$.
4. Return d.

The proof of correctness is identical to that of IQAE from [GGZW19]: since we grow the degree by a factor of r with every confidence interval, we will need at most T many confidence intervals. Since each interval contains $T_d(a)$ with probability $\geq 1 - \delta/T$, the union bound implies that the algorithm fails with probability at most δ overall. By the same analysis, the asymptotic query complexity is $O(\varepsilon^{-1} \log(\delta^{-1} \log(\varepsilon^{-1})))$.

The data supporting the empirical claim is presented in Figure 6. We took 1000 many runs of the above procedure for each of 9 logarithmically spaced ε in the interval $\varepsilon \in [10^{-2}, 10^{-6}]$ with $\delta = 0.05$ and $a = 0.5$. We found that for each ε the fraction of runs where $|a - \hat{a}| > \varepsilon$ was indeed always $< \delta$.

We took the mean of the sampled query complexities Q_Π for each value of ε, call it $\langle Q_\Pi \rangle$, and fitted the function $f_{A,B}(\varepsilon) := A\varepsilon^{-1} \ln(B \ln(\varepsilon^{-1}))$. Using brute force, we found the parameters $(A, B) = (1.71, 2.08)$ ensured that the model $f_{A,B}(\varepsilon)$ was always within a 3.15% relative error of $\langle Q_\Pi \rangle$. The same parameters ensure that $f_{A,B}(\varepsilon)$ are within 71.77% of all values of Q_Π. We also fitted a simpler model $f_C(\varepsilon) := C/\varepsilon$ to $\langle Q_\Pi \rangle$, and found that the best parameter choice $C = 4.66$ approximates both $\langle Q_\Pi \rangle$ to within 19.65% and Q_Π to within 74.28%.

The software that performed this analysis is available at:

https://github.com/qiskit-community/ChebAE/blob/main/chebae.ipynb

□

A key question in this data analysis is if we should consider in our fits the extra $\log(\log(\varepsilon^{-1}))$ factor, which is present in the asymptotic performance. This is complicated by the fact that $\mathbf{Q}_\Pi$ is a random variable, and visibly exhibits significant variation about the mean. The $f_{A,B}$ fit function captures the empirical mean $\langle Q_\Pi \rangle$ of our data rather accurately, suggesting that the $\log(\log(\varepsilon^{-1}))$ does affect the empirical query complexity. But the most extreme values of Q_Π are only within $\sim 70\%$ of the $f_{A,B}$ model. However, the simpler f_C model does not capture Q_Π or $\langle Q_\Pi \rangle$ any better. This suggests that the large error of $\sim 70\%$ does not stem from poor fit quality but from actual variation within the random variable $\mathbf{Q}_\Pi$.

Having presented the algorithm, we seek to demonstrate three things. First, we want to argue a significant improvement in query complexity relative to prior art. Second, we want to

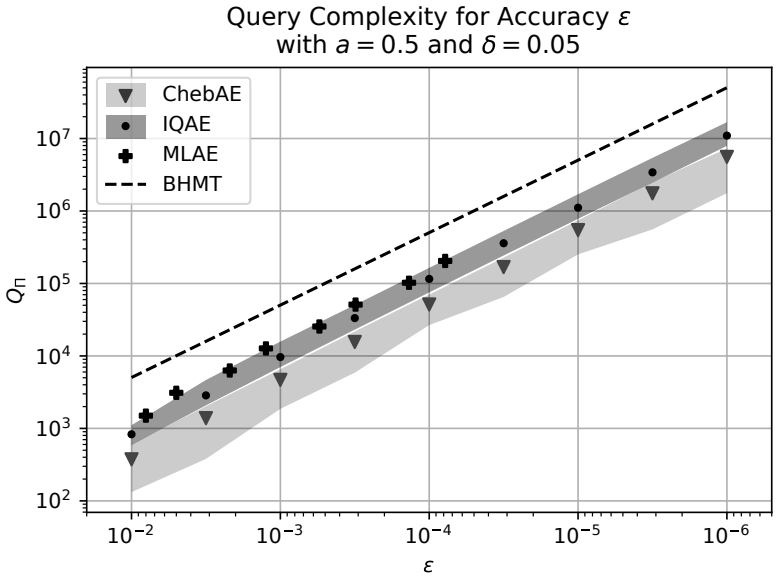


Figure 6: Comparison of amplitude estimation algorithms with a fixed underlying amplitude and failure probability, and varying target accuracy. Each point is the average of 1000 samples, and the shaded regions around IQAE and ChebAE indicate the minimum and maximum query complexities observed. For MLAE, each point is the average of 100 runs.

demonstrate that it was indeed the 'late' vs 'early' heuristic that resulted in the improvement over IQAE. Third, we show that ChebAE's requires fewer queries when $a \approx 1$.

To compare to the prior art, we characterize the constant-factor query complexity performance of some of the previous algorithms. We consider [BHMT00] for reference, and also look at MLAE and IQAE since these two seem to have similar performance according to prior literature [GGZW19, YLRJ20]. Fortunately, the amplitude estimation method from [BHMT00] has a closed-form expression for its query complexity.

Proposition 19. *[BHMT00] For any $\varepsilon, \delta > 0$, there is an amplitude estimation algorithm (not in the form of Definition 7) that outputs an estimate $\hat{\mathbf{a}}$ satisfying $Pr[|\hat{\mathbf{a}} - a| \leq \varepsilon] > 1 - \delta$ with deterministic query complexity Q_Π as follows:*

$$Q_\Pi = \left\lceil \frac{\pi}{\arcsin \varepsilon} \right\rceil \cdot \left\lceil \frac{1}{2} \left(\frac{8}{\pi^2} - \frac{1}{2} \right)^{-2} \ln \frac{1}{\delta} \right\rceil \tag{71}$$

At $\delta = 0.05$ this is about $Q_\Pi \approx_{1\%} 50/\varepsilon$ for $\varepsilon < 10^{-2}$.

For MLAE and IQAE, we perform similar numerical experiments as in Empirical Claim 18.

Empirical Claim 20. *For $a = 0.5$, $\delta = 0.05$ and $\varepsilon \in [10^{-3}, 10^{-6}]$, the IQAE algorithm from [GGZW19] satisfies:*

$$\langle Q_\Pi \rangle \approx_{7.27\%} \frac{2.62}{\varepsilon} \ln \left(6.61 \ln \left(\frac{1}{\varepsilon} \right) \right) \tag{72}$$

For the same ε, δ, the MLAE algorithm from [Suzuki&19] satisfies:

$$\langle Q_\Pi \rangle, Q_\Pi \approx_{13.91\%} \frac{3.48}{\varepsilon} \ln \left(9.89 \ln \left(\frac{1}{\varepsilon} \right) \right) \tag{73}$$

Proof. For IQAE we took 1000 runs with $N_{\text{shots}} = 100$, $r = 2$ for the same 9 logarithmically spaced ε in the interval $\varepsilon \in [10^{-3}, 10^{-6}]$ with $\delta = 0.05$ and $a = 0.5$. The $f_{A,B}$ model with $(A, B) = (2.62, 6.61)$ captured the mean with $\langle Q_\Pi \rangle$ to within 7.27%, and captured all values with Q_Π to within 57.62%. The $f_C(\varepsilon)$ model fitted to $\langle Q_\Pi \rangle$ with $C = 9.93$ achieved $\langle Q_\Pi \rangle \approx_{16.41\%} f_C$ and $Q_\Pi \approx_{72.31\%} f_C$.

MLAE does not take ε as an input parameter. Instead, following [GGZW19], we take 100 samples from T_{2k+1} with $k = 0, 2^0, 2^1, ..., 2^k$ and compute a confidence interval based on likelihood ratio method. The reported ε is the half-width of this interval. In some sense, this gives MLAE an unfair advantage: IQAE and ChebAE also produce a confidence interval, and that confidence interval's half-width may be smaller than the requested ε. IQAE and ChebAE do not report that smaller potentially smaller half-width, while MLAE does.

Either way, we perform 100 runs with $k = \{1, \ldots, 11\}$ with $\delta = 0.05$ and $a = 0.5$. Interestingly, we observed that Q_Π hardly deviated from its mean $\langle Q_\Pi \rangle$. The $f_{A,B}(\varepsilon)$ model with $(A, B) = (3.48, 9.89)$ achieved $\langle Q_\Pi \rangle, Q_\Pi \approx_{13.91\%} f_{A,B}$. The $f_C(\varepsilon)$ model with $C = 9.99$ achieved $\langle Q_\Pi \rangle, Q_\Pi \approx_{70.49\%} f_C$.

The results of these experiments are also shown in Figure 6, and the fraction of incorrect estimates was $< \delta$ for all ε (IQAE) and for all k (MLAE). □

Again we are in a situation where the performance can be measured differently depending on what model is chosen. Just like ChebAE, for IQAE the $f_{A,B}$ model captures $\langle Q_\Pi \rangle$ more accurately than f_C. This again suggests that the $\log(\log(\varepsilon^{-1}))$ dependence is empirically visible. The accuracy of the $f_{A,B}$ and f_C models for MLAE are about the same.

Now we justify the claim that ChebAE only needs 45% to 65% of the queries of IQAE. Any such claim will be severely reductive: both of the query complexities are random, depend on the amplitude a being estimated, and the fit methodology also affects the claim. Certainly from Figure 6 we can see that ChebAE exhibits a clear speedup over IQAE, with the worst ChebAE Q_Π being about the same as the best IQAE Q_Π. The speedup is visible despite the enormous variation of Q_Π around $\langle Q_\Pi \rangle$. To quantify the reduction in query complexiy we will focus only on the average $\langle Q_\Pi \rangle$ and rely on the fits we performed. Both ChebAE and IQAE are better explained by the $f_{A,B}$ model. The fact that we need two parameters to properly fit ChebAE and IQAE makes a comparison more complicated, but ChebAE improves over IQAE in both parameters A, B. We can thus be generous to IQAE and ignore the B parameter, and say the fraction is about $1.71/2.62 \approx 65\%$. If we use the f_C model then the fraction is $4.66/9.93 \approx 45\%$. By this metric, IQAE only needs about $9.93/50 \approx 20\%$ of the queries of [BHMT00].

It remains to justify that it was the altered 'late' vs 'early' heuristic that was the primary source of the speedup. ChebAE is extremely similar to IQAE, and the differences can be summarized with three alterations: First, rather than obtain an estimate of the Grover angle $\theta := \arcsin(a)$, ChebAE estimates a directly. Second, ChebAE considers both odd and even Chebyshev polynomials whereas IQAE only considers odd polynomials. Third, the 'late' vs 'early' heuristic in step 4(b) in Empirical Claim 18 has a different form and also has the ν optimization parameter.

We compare these changes individually in Figure 7. Here we observe that (for $a = 0.5$) modifying IQAE to estimate a instead of θ significantly worsens the query complexity on average, and increases the amount of random variation. We also observe that when ChebAE is restricted to consider only odd polynomials just like IQAE does, then the performance

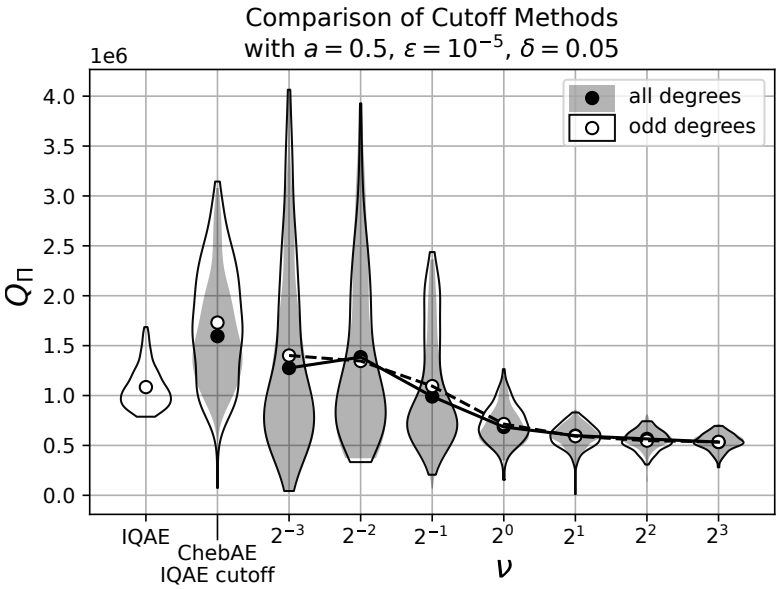


Figure 7: Comparison of different variants of the ChebAE algorithm to the IQAE algorithm. All violins are based on 200 samples and have $a = 0.5, \varepsilon = 10^{-5}$, and $\delta = 0.05$, as well as $r = 2$ and $N_{\text{shots}} = 100$. We see how considering Chebyshev polynomials with even degrees as well as odd degrees does not affect the performance much. The primary deciding factor of performance is the cutoff method - step 4(b) in Empirical Claim 18. The leftmost violin shows [GGZW19]'s implementation of IQAE for reference. The violin marked 'Chebae IQAE cutoff' is obtained by taking ChebAE and replacing step 4(b) with the cutoff criterion taken from [GGZW19]'s IQAE implementation (see also lines 12-15 in Algorithm 1 of [GGZW19]). The violins on the right leave step 4(b) as in Empirical Claim 18, but vary ν.

hardly changes. When switching to the new heuristic in step 4(b), the performance is slightly improved even when ν is small. Then, the query complexity decreases with increasing ν until even the worst ChebAE runs are faster than the best IQAE runs. So clearly it was the altered heuristic and the tuning of ν that caused the improvement.

The behavior of ν can be explained as follows: as ν increases the heuristic switches from 'early' to 'late' for earlier iterations, thus decreasing the chances of taking more samples from the expensive Chebyshev polynomials near the end. The improvement with ν exhibits diminishing returns, and worsens the classical performance of algorithm analysis. We find little benefit in increasing ν beyond $\nu = 8$, and the classical analysis time is still faster than that of IQAE.

Finally, we remark on the performance as a function of a. In Proposition 19 we claimed that the accuracy [BHMT00] is independent of a. However, Theorem 12 in [BHMT00] actually has an extra dependence on a. Both IQAE and [BHMT00] estimate the Grover angle $\theta := \arcsin(a)$, and then convert the estimate of θ to an estimate of a. When $a \approx 1$, a less accurate estimate of θ suffices for the same accuracy on a, if only some bounds on a were known beforehand. We consider a setting where nothing is known about a before running the algorithm. Furthermore, we request that the algorithms return an estimate of a certain accuracy ε, and then do not reward the algorithms for achieving a higher accuracy than what was requested. In this more realistic setting for estimation, IQAE and [BHMT00] must assume the worst-case value of a to determine their query complexity and are then penalized

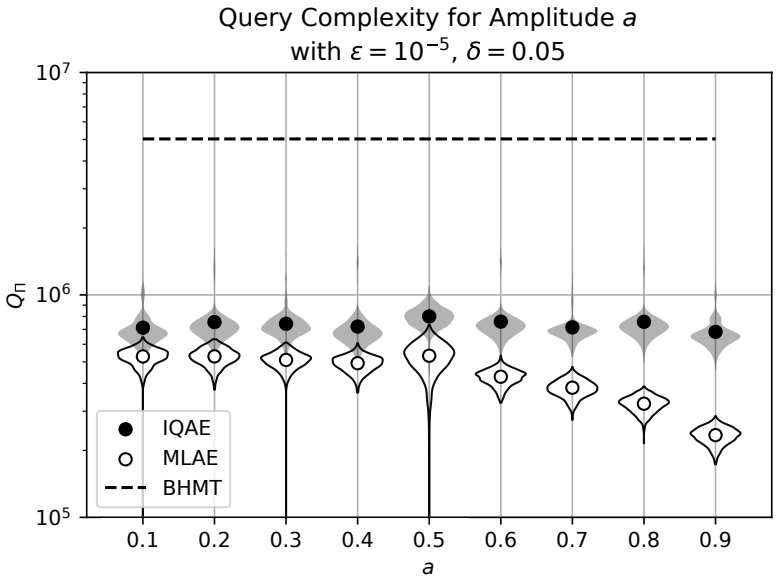


Figure 8: Comparison of amplitude estimation algorithms at a fixed accuracy and success probability while varying underlying amplitude a being estimated. Each violin has 1000 samples, and the dots indicate the means.

for overshooting their accuracy. Consequently, their query complexity as a function of a is constant.

The experiments presented in Figure 8 shows that the query complexity of ChebAE delivers smaller query complexities when a is closer to 1. This stems from the fact that ChebAE does not indirectly estimate a via the Grover angle θ, and instead maintains a confidence interval on a directly. By estimating a directly, ChebAE can take advantage of the fact that Chebyshev polynomials have larger slope when a is closer to 1, and thereby exhibit improved performance for these amplitudes even if nothing is known about a beforehand. The algorithm terminates as soon as the estimate of a is good enough, so it does not overshoot and obtain a higher estimate than necessary. Note that for the numerical experiments supporting our claim that ChebAE outperforms IQAE more generally, we deliberately chose $a = 0.5$ because that way we compare IQAE to the slowest possible version of ChebAE.

4 Unbiased Estimation

In this section we present a simple new algorithm for amplitude estimation based on sampling from semi-Pellian polynomials. This algorithm has a new capability: it samples from a random variable $\hat{\mathbf{a}}$ satisfying $\mathbb{E}[\hat{\mathbf{a}}] \approx a$. We call this task **unbiased amplitude estimation.**

This is the first algorithm achieving unbiased amplitude estimation in the literature to our knowledge. On the other hand, unbiased *phase* estimation has appeared in [LdW21, vACGN22, CH22]. The main idea is that if U has eigenphase $e^{i\phi}$, we can prepare $Ue^{i\boldsymbol{\theta}}$ with eigenphase $e^{i(\phi+\boldsymbol{\theta})}$ for a random shift $\boldsymbol{\theta}$. After applying phase estimation to $Ue^{i\boldsymbol{\theta}}$ and subtracting off the random shift yields an unbiased estimator for ϕ.

However, one cannot use [BHMT00] to directly translate the technique of [LdW21] to amplitude estimation. [BHMT00] directly estimates $\theta := \arcsin(a)$ via phase estimation. Say

we had an unbiased estimator $\hat{\boldsymbol{\theta}}$ of θ. Then, $\sin(\hat{\boldsymbol{\theta}})$ is not an unbiased estimator of a. A different approach is needed.

Instead, we rely on the following strategy. Say we have established a confidence interval $[a_{\min}, a_{\max}]$ within which a is contained with high probability. Suppose furthermore that $a_{\max} - a_{\min} < \varepsilon$, so any value in the interval is within ε of a. We construct a polynomial $P(a)$ such that:

$$|P(a)|^2 \approx \frac{a - a_{\min}}{a_{\max} - a_{\min}} \tag{74}$$

Observe that $|P(a)|^2$ approximates a line with $|P(a_{\min})|^2 \approx 0$ and $|P(a_{\max})|^2 \approx 1$. So, we let our final output be:

$$\hat{\mathbf{a}} = a_{\max} \text{ if } \mathbf{b}_P = 1, \text{ else } a_{\min} \tag{75}$$

recalling that $\Pr[\mathbf{b} = 1] = |P(a)|^2$ from Theorem 6. Then $\mathbb{E}[\hat{\mathbf{a}}] = |P(a)|^2 a_{\max} + (1 - |P(a)|^2) a_{\min} \approx a$, so the estimator is approximately unbiased. But since we also have $\hat{\mathbf{a}} \in [a_{\min}, a_{\max}]$, we also have a separate guarantee that $\hat{\mathbf{a}}$ is within ε of a.

Sampling from the polynomial $P(a)$ immediately solves the problem once we have obtained the confidence interval $[a_{\min}, a_{\max}]$. But it turns out that the same polynomial can also be used to refine such a confidence interval, that is, shrinking it by a factor of 0.9 while keeping a in the interval with high probability. This immediately yields a full algorithm for amplitude estimation that is rather similar to [AR19, GGZW19], but dramatically simpler than either of them.

Thus, the main technical work of this section is to find a polynomial $P(a)$ with the desired properties. To this end, we use a standard result form the theory of function approximation using polynomials.

Theorem 21. ***Jackson's Theorem.*** *[Rivlin69] For any continuous function $g(x)$ on the interval $[-1, 1]$ there exists a polynomial $P(x)$ of degree at most d so that for all $x \in [-1, 1]$:*

$$|J(x) - g(x)| \leq 6\omega_g(1/d), \tag{76}$$

where $\omega_g(\delta)$ is the modulus of continuity of $g(x)$ given by:

$$\omega_g := sup\{|g(x) - g(y)| \text{ for } x, y \in [-1, 1] \text{ with } |x - y| \leq \delta\}. \tag{77}$$

Essentially, we can approximate any continuous function $g(x)$ with a polynomial to error ε. The degree will be on the order of the maximum slope of g times ε^{-1}. One key complication remains: we want $|P(a)|^2$ to approximate a line, not $P(a)$. This means we need to account for the Born rule in our choice of $g(x)$, which makes the error analysis a bit more involved.

Lemma 22. *For $0 < a_{min} < a_{max} < 1$, let $\Delta := a_{max} - a_{min}$. Then for any $\eta > 0$, there exists an even semi-Pellian polynomial $P_\eta(x)$ such that:*

$$\left| |P_\eta(x)|^2 - \frac{x - a_{min}}{\Delta} \right| \leq \eta \tag{78}$$

for all $x \in [a_{min}, a_{max}]$. Furthermore $deg(P_\eta) \in O(\eta^{-1}\Delta^{-1})$.

Proof. Let c be some constant that we will pick later in the proof, and let $g(x)$ be defined on the interval $[-1,1]$ as

$$g(x) = \begin{cases} 0 & |x| < a_{\min} \\ \sqrt{c \cdot \frac{|x|-a_{\min}}{\Delta}} & |x| \in [a_{\min}, a_{\max}] \\ \sqrt{c} & |x| > a_{\max} \end{cases} \tag{79}$$

Observe that $g(x)$ is even, continuous, and that $\omega_g(1/d) \leq \sqrt{\frac{c}{\Delta d}}$. Following Theorem 21, let $J(x)$ be a degree d polynomial satisfying $|J(x) - g(x)| \leq 6\omega_g(1/d)$. We will also pick d later.

We would like to ensure that our final polynomial $P_\eta(x)$ is even, even though $J(x)$ might not be. Actually, under the hood, Jackson's theorem relies on a Chebyshev expansion of $g(x)$. Since $g(x)$ is even, all the odd components of the expansion will vanish, so $J(x)$ will be even anyway. But we do not need to rely on this fact to proceed: we simply let $P_\eta(x)$ be the even part of $J(x)$. Then:

$$P_\eta(x) := \frac{1}{2}(J(x) + J(-x)) \tag{80}$$

$$|P_\eta(x) - g(x)| \leq \frac{1}{2}(|J(x) - g(x)| + |J(-x) - g(-x)|) \tag{81}$$

$$\leq 6\omega_g(1/d) \tag{82}$$

$P_\eta(a)$ is even, so it has fixed parity. If we can also show that $|P_\eta(x)| \leq 1$ then by Corollary 10 of [GSLW18] we have demonstrated that $P_\eta(x)$ is semi-Pellian.

$$\left||P_\eta(x)|^2 - g(x)^2\right| \leq (6\omega_g(1/d))^2 \leq \frac{36c}{\Delta d} \tag{83}$$

$$|P_\eta(x)|^2 \leq \max_x g(x)^2 + \left||P_\eta(x)|^2 - g(x)^2\right| \tag{84}$$

$$\leq c + \frac{36c}{\Delta d} \tag{85}$$

Thus, if we select $c := \left(1 + \frac{36}{\Delta d}\right)^{-1}$, then we have $|P_\eta(x)|^2 \leq 1$, so $|P_\eta(x)| \leq 1$, so P_η is semi-Pellian.

It remains to show that $|P_\eta(x)|^2$ approximates $\frac{x-a_{\min}}{\Delta}$ on the interval $[a_{\min}, a_{\max}]$, and to select d. Let $\bar{c} := 1 - c$ and apply the triangle inequality:

$$\left||P_\eta(x)|^2 - \frac{x-a_{\min}}{\Delta}\right| \leq \left||P_\eta(x)|^2 - c\frac{x-a_{\min}}{\Delta}\right| + \left|c\frac{x-a_{\min}}{\Delta} - \frac{x-a_{\min}}{\Delta}\right| \tag{86}$$

$$\leq \frac{c}{\Delta d} + (1-c) \tag{87}$$

$$\leq \frac{1-\bar{c}}{\Delta d} + \bar{c} \tag{88}$$

$$\leq \frac{1 - \bar{c} + \bar{c}\Delta d}{\Delta d} \tag{89}$$

$$\leq \frac{1}{\Delta d} - \bar{c}\frac{1+\Delta d}{\Delta d} \leq \frac{1}{\Delta d} \tag{90}$$

So, we achieve $\frac{1}{\Delta d} \leq \eta$ when $d := \lceil \eta^{-1}\Delta^{-1} \rceil$. This completes the proof. □

Now that we have a construction for $P_\eta(a)$, we can present the algorithm for unbiased estimation. The main idea is to take an interval $[a_{\min}, a_{\max}]$ starting at $[0, 1]$, and to refine it until $\Delta := a_{\min} - a_{\max}$ satisfies $\Delta < \varepsilon$. Then we apply the trick we described at the beginning of the section to sample a $\hat{\mathbf{a}}$ satisfying both $\mathbb{E}[\hat{\mathbf{a}}] \approx a$ and $\Pr[|\hat{\mathbf{a}} - a| \geq \varepsilon] \leq \delta$.

Before we step into the main proof, we take note of three things:

- Since we always shrink Δ by the same factor, it is possible to anticipate how many iterations we need: $\sim \log(\varepsilon^{-1})$ many. If we grow our failure probability at each iteration δ_t via a geometric series, we can avoid an extra $\log(\varepsilon^{-1})$ factor in the query complexity. This analysis is taken directly from [AR19].

- For shrinking the interval, we actually can get away with a constant accuracy $\eta = 0.1$ in our polynomial approximation. It is only the final step where we sample $\hat{\mathbf{a}}$ that requires high accuracy.

- Since any random variable bounded in the interval $[a_{\min}, a_{\max}]$ must also have its expectation in that interval, we automatically get $|\mathbb{E}[\hat{\mathbf{a}}] - a| \leq \varepsilon$. So the main capability of the algorithm is to make expected value even closer to a. This additional accuracy η enters the runtime as η^{-1}, stemming from the accuracy of Jackson's theorem. Is there a polynomial approximation achieves polylogarithmic scaling in η^{-1}?

Theorem 23. ***Unbiased amplitude estimation.*** *For every $\varepsilon, \delta, \eta > 0$ there exists an amplitude estimation algorithm satisfying:*

$$Pr[|\hat{\mathbf{a}} - a| \geq \varepsilon] \leq \delta \tag{91}$$

$$\mathbf{D} \text{ is not random and } \mathbf{D} \in O(\varepsilon^{-1}(\log(\delta^{-1}) + \eta^{-1})) \tag{92}$$

$$|\mathbb{E}[\hat{\mathbf{a}}] - a| \leq \varepsilon\eta + \delta \tag{93}$$

Proof. The algorithm proceeds as follows:

1. Initialize $a_{\min}^{(0)} \leftarrow 0$ and $a_{\max}^{(0)} \leftarrow 1$. Let $\Delta_t := a_{\max}^{(t)} - a_{\min}^{(t)}$ throughout.

2. Let $T := \lceil \log_{0.9}(\varepsilon) \rceil$. For $t = 0, 1, 2, ..., T-1$, do:

   (a) Use Lemma 22 to construct a polynomial $P_{0.1}^{(t)}(a)$ whose square is 0.1-close to $(a - a_{\min}^{(t)})/\Delta_t$ on $[a_{\min}^{(t)}, a_{\max}^{(t)}]$.

   (b) Let $\delta_t := (\varepsilon\delta/10) \cdot 0.9^{-t}$, sample from $\mathbf{b}_{P_{0.1}^{(t)}}$ a total of $m_t := 6\ln(\delta_t^{-1})$ times, and let $\mathbf{S}^{(t)}$ be the total.

   (c) If $\mathbf{S}^{(t)} > m_t/2$, then set:

$$[a_{\min}^{(t+1)},\ a_{\max}^{(t+1)}] \leftarrow [a_{\min}^{(t)} + 0.1\Delta_t,\ a_{\max}^{(t)}] \tag{94}$$

   Otherwise, set:

$$[a_{\min}^{(t+1)},\ a_{\max}^{(t+1)}] \leftarrow [a_{\min}^{(t)},\ a_{\max}^{(t)} - 0.1\Delta_t] \tag{95}$$

3. Use Lemma 22 to construct a polynomial $P_\eta^{(T)}(a)$ whose square is η-close to $(a - a_{\min}^{(T)})/\Delta_T$ on $[a_{\min}^{(T)}, a_{\max}^{(T)}]$.

4. Sample from $\mathbf{b}_{P_\eta^{(T)}}$ once. If $\mathbf{b}_{P_\eta^{(T)}} = 1$, then return $\hat{\mathbf{a}} \leftarrow a_{\max}^{(T)}$. Otherwise return $\hat{\mathbf{a}} \leftarrow a_{\min}^{(T)}$.

At the beginning we have $\Delta_0 = 1$, and each of the T iterations of step 2 transforms $\Delta_{t+1} = 0.9\Delta_t$. So, at the end we have $\Delta_T = 0.9^T \leq \varepsilon$.

We first establish that $\Pr[|\hat{\mathbf{a}} - a| \geq \varepsilon] \leq \delta$. This is guaranteed if $a \in [a_{\min}^{(T)}, a_{\max}^{(T)}]$ after step 2 finishes, since afterwards both $|a_{\max}^{(T)} - a| \leq \Delta_T \leq \varepsilon$ and $|a_{\min}^{(T)} - a| \leq \Delta_T \leq \varepsilon$. So it suffices to show that the probability we ever have $a \notin [a_{\min}^{(T)}, a_{\max}^{(T)}]$ is at most δ.

Say we are at step t of step 2, and we have $a \in [a_{\min}^{(t)}, a_{\max}^{(t)}]$. Then, there are two ways we could update the interval to make $a \notin [a_{\min}^{(t+1)}, a_{\max}^{(t+1)}]$: either $a \leq a_{\max}^{(t)} - 0.9\Delta_t$ and $\mathbf{S}^{(t)} > m_t/2$, or $a \geq a_{\min}^{(t)} + 0.9\Delta_t$ and $\mathbf{S}^{(t)} \leq m_t/2$. We show that both of these cases occur with probability at most δ_t.

Say $a \leq a_{\max}^{(t)} - 0.9\Delta_t = a_{\min}^{(t)} + 0.1\Delta_t$. We have:

$$|P_{0.1}^{(t)}(a)|^2 \leq \frac{a - a_{\min}^{(t)}}{\Delta_t} + 0.1 \tag{96}$$

$$\leq 0.1 + 0.1 = 0.2. \tag{97}$$

So $\mathbb{E}[\mathbf{b}_{P_{0.1}^{(t)}}] \leq 0.2$, implying $\mathbb{E}[\mathbf{S}^{(t)}] \leq 0.2m_t$. Using the Chernoff-Hoeffding theorem:

$$\Pr[\mathbf{S}^{(t)} > m_t/2] \leq \Pr[\mathbf{S}^{(t)} \geq \mathbb{E}[\mathbf{S}^{(t)}] + 0.3m_t] \tag{98}$$

$$\leq e^{-2(0.3)^2 m_t} \leq \delta_t. \tag{99}$$

Similarly say $a \geq a_{\min}^{(t)} + 0.9\Delta_t$. Then $\mathbb{E}[\mathbf{b}_{P_{0.1}^{(t)}}] \geq 0.8$, so $\Pr[\mathbf{S}^{(t)} \leq m_t/2] = \Pr[\mathbf{S}^{(t)} \leq \mathbb{E}[\mathbf{S}^{(t)}] - 0.3m_t] \leq \delta_t$ by the same argument.

To finish the argument that $\Pr[|\hat{\mathbf{a}} - a| \geq \varepsilon] \leq \delta$, we follow the analysis of [AR19]. Let $b := 1/0.9 \approx 1.11$ and observe that $T = \lceil \log_b(\varepsilon^{-1}) \rceil$. We bound the probability that any of the steps fail with a union bound:

$$\sum_{t=0}^{T-1} \delta_t = \frac{\varepsilon\delta}{10} \sum_{t=0}^{T-1} b^t \leq \frac{\varepsilon\delta}{10} \frac{b^T - 1}{b - 1} \leq \frac{\delta}{10} \frac{1}{b-1} \leq \delta. \tag{100}$$

Now we show that $\mathbf{D}$ only takes a single value with probability 1. We observe that in Lemma 22, while $P_\eta^{(t)}(a)$ itself depends on the individual values of $a_{\min}^{(t)}, a_{\max}^{(t)}$, the degree only depends on their difference Δ_t. While the values $a_{\min}^{(t)}, a_{\max}^{(t)}$ may vary randomly in the algorithm, Δ_t is always exactly 0.9^t. Since the total number of polynomials we sample from is also independent of $a_{\min}^{(t)}, a_{\max}^{(t)}$, to total degree of all polynomials $\mathbf{D}$ is deterministic. This makes asymptotic claims on $\mathbf{D}$ well defined.

At the t'th iteration of step 2, we sample from a polynomial of degree $\in O(\Delta_t^{-1} 0.1^{-1}) = O(0.9^{-t}) = O(b^t)$ a total of m_t times. Then, in step 4, we sample from a polynomial of degree

$\in O(\Delta_t^{-1}\eta^{-1}) \subseteq O(\varepsilon^{-1}\eta^{-1})$ once. So the total degree is:

$$\mathbf{D} \in O\left(\sum_{t=0}^{T-1} \frac{m_t}{0.9^t} + \frac{1}{\eta\varepsilon}\right) \leq O\left(\sum_{t=0}^{T-1} \frac{1}{0.9^t} \ln\left(\frac{1}{\delta_t}\right) + \frac{1}{\eta\varepsilon}\right) \tag{101}$$

$$\leq O\left(\sum_{t=0}^{T-1} \frac{1}{0.9^t} \ln\left(\frac{0.9^t}{\varepsilon\delta}\right) + \frac{1}{\eta\varepsilon}\right) \tag{102}$$

$$\leq O\left(\frac{1}{\varepsilon}\ln\left(\frac{1}{\delta}\right) + \frac{1}{\eta\varepsilon}\right) + O\left(\sum_{t=0}^{T-1} b^t \left[\ln\left(\frac{1}{\varepsilon}\right) - t\ln(b)\right]\right) \tag{103}$$

The first term is the desired bound, so all that remains to show is that the second term is dominated by the first term. Indeed, a somewhat cumbersome calculation in equations (38-44) of [AR19] demonstrates that the second term is $\leq O(\varepsilon^{-1})$ (albeit with $T+1$ instead of T, which is a stronger result).

Finally, we prove the fact that makes this construction unique: that $|\mathbb{E}[\hat{\mathbf{a}}] - a| \leq \varepsilon\eta + \delta$. To do so, we interpret the dynamics of the algorithm as a binary decision tree: at each node we can either proceed to the left child where we increase $a_{\min}$ or the right child where we decrease $a_{\max}$. At the t'th iteration of step 2 we are at the t'th layer of the tree, and we can think of step 4 as the T'th layer containing all the leaves. See Figure 9.

We can label each node in the t'th layer of the tree with a bit string $\vec{x} \in \{0,1\}^t$, where the bits encode the choices made to get to that node. We say a node $\vec{x}$ is 'good' if the corresponding interval $[a_{\min}^{(t)}, a_{\max}^{(t)}]$ contains a, otherwise we say $\vec{x}$ is 'bad'. Each node is associated with a random variable $\hat{\mathbf{a}}_{\vec{x}}$, denoting the output of the algorithm assuming we were at the $\vec{x}$ node at iteration $t = \dim(\vec{x})$. Correspondingly, at the root of the tree $\hat{\mathbf{a}}$ equals $\hat{\mathbf{a}}_{\vec{x}}$ where $\vec{x}$ is the 'empty' bit string with dimension 0.

We establish $|\mathbb{E}[\hat{\mathbf{a}}] - a| \leq \varepsilon\eta + \delta$ by proving the following fact. For all $t \in [0, 1, ..., T]$ and for all good $\vec{x} \in \{0,1\}^t$:

$$|\mathbb{E}[\hat{\mathbf{a}}_{\vec{x}}] - a| \leq \varepsilon\eta + \sum_{k=t}^{T-1} \delta_k \tag{104}$$

Then $|\mathbb{E}[\hat{\mathbf{a}}] - a| \leq \varepsilon\eta + \delta$ follows by plugging in $t = 0$ and noting that $\sum_{k=0}^{T-1} \delta_k \leq \delta$ as shown above.

We prove the statement recursively, starting with the leaves and working our way to the root. For a leaf, $t = T$ so the sum $\sum_{k=T}^{T-1} \delta_k$ vanishes. We have:

$$\mathbb{E}[\hat{\mathbf{a}}_{\vec{x}}] = |P_\eta^{(T)}(a)|^2 \cdot a_{\max} - (1 - |P_\eta^{(T)}(a)|^2) \cdot a_{\min} \tag{105}$$

$$= |P_\eta^{(T)}(a)|^2 \cdot \Delta_T + a_{\min} \tag{106}$$

By Lemma 22, we have $||P_\eta^{(T)}(a)|^2 - (a - a_{\min}^{(T)})/\Delta_T| \leq \eta$. So:

$$\eta\Delta_T \geq \left||P_\eta^{(T)}(a)|^2 - (a - a_{\min}^{(T)})/\Delta_T\right| \cdot \Delta_T \tag{107}$$

$$= \left|\left(|P_\eta(a)|^2 \cdot \Delta_T + a_{\min}^{(T)}\right) - a\right| = |\mathbb{E}[\hat{\mathbf{a}}_{\vec{x}}] - a| \tag{108}$$

At step 4 we have $\Delta_T \leq \varepsilon$, so we have established $|\mathbb{E}[\hat{\mathbf{a}}_{\vec{x}}] - a| \leq \varepsilon\eta$ as desired.

Now we recursively show the statement for t and all good $\vec{x} \in \{0,1\}^t$, assuming it holds for all good $\vec{x}y$ at $t+1$, with $y \in \{0,1\}$. There are two cases: either $\vec{x}y$ is good for both $y \in \{0,1\}$, or it is only good for one of them. It cannot be bad for both y, because then a would need to be both $\leq a_{\min}^{(t)} + 0.1\Delta_t$ and $\geq a_{\min}^{(t)} + 0.9\Delta_t$.

If $\vec{x}y$ is good for both y (like $\vec{x} = 01$ in Figure 9), then by assumption $|\mathbb{E}[\hat{\mathbf{a}}_{\vec{x}y}] - a| \leq \varepsilon\eta + \sum_{k=t+1}^{T} \delta_k$. For some probability p we have:

$$\mathbb{E}[\hat{\mathbf{a}}_{\vec{x}}] = p \cdot \mathbb{E}[\hat{\mathbf{a}}_{\vec{x}0}] + (1-p) \cdot \mathbb{E}[\hat{\mathbf{a}}_{\vec{x}1}] \tag{109}$$

$$|\mathbb{E}[\hat{\mathbf{a}}_{\vec{x}}] - a| \leq p \cdot |\mathbb{E}[\hat{\mathbf{a}}_{\vec{x}0}] - a| + (1-p) \cdot |\mathbb{E}[\hat{\mathbf{a}}_{\vec{x}1}] - a| \tag{110}$$

$$\leq \varepsilon\eta + \sum_{k=t+1}^{T} \delta_k \leq \varepsilon\eta + \sum_{k=t}^{T} \delta_k \tag{111}$$

Now assume without loss of generality that $\vec{x}0$ is bad, and $\vec{x}1$ is good (like $\vec{x} = 0$ in Figure 9). Now the assumption says nothing about $|\mathbb{E}[\hat{\mathbf{a}}_{\vec{x}0}] - a|$, but we at least know that $|\mathbb{E}[\hat{\mathbf{a}}_{\vec{x}0}] - a| \leq 1$. Furthermore, we know that the probability p with which we proceed down the $\vec{x}0$ path is at most δ_t. So:

$$|\mathbb{E}[\hat{\mathbf{a}}_{\vec{x}}] - a| \leq \delta_t \cdot |\mathbb{E}[\hat{\mathbf{a}}_{\vec{x}0}] - a| + |\mathbb{E}[\hat{\mathbf{a}}_{\vec{x}1}] - a| \tag{112}$$

$$\leq \varepsilon\eta + \delta_t + \sum_{k=t+1}^{T} \delta_k = \varepsilon\eta + \sum_{k=t}^{T} \delta_k. \tag{113}$$

□

5 Hybrid Classical-Quantum Estimation

In the future, we may have access to quantum computers that are capable of running Grover's algorithm but are still limited in the overall circuit depth. Amplitude estimation algorithms may become useful earlier if they can still accelerate estimation to accuracy ε without requiring depth $O(1/\varepsilon)$. This can be achieved by mixing classical and quantum resources.

In particular, [GKLPZ20] consider the following version of this problem: for any $0 \leq \beta < 1$ they show that there exists an amplitude estimation algorithm that runs in time $O(1/\varepsilon^{1+\beta})$, but requires a maximum depth of only $O(1/\varepsilon^{1-\beta})$. When $\beta = 0$ this reproduces the 'fully quantum' performance of other algorithms, and when $\beta \to 1$ this algorithm has the same asymptotic complexity as classical probability estimation.

We consider the construction of such hybrid algorithms a fascinating theoretical problem. [GKLPZ20] give two methods for achieving this: a 'Power law' algorithm relying on maximum likelihood estimation, and a 'QoPrime' algorithm relying on the Chinese remainder theorem. In this section we give another method for solving this problem based on quantum signal processing. Depending on the reader's background this new method may be more intuitive.

However, we remark that we consider it questionable if any of these methods are actually useful in practice. In particular, constructing the algorithms to achieve the nice theoretical scaling of $O(1/\varepsilon^{1\pm\beta})$ undercuts the simplicity of the underlying situation. A depth-limited quantum computer will be able to run some fixed number of Grover rotations r. A practical version of hybrid estimation would take as input such an r, and use that to extract as

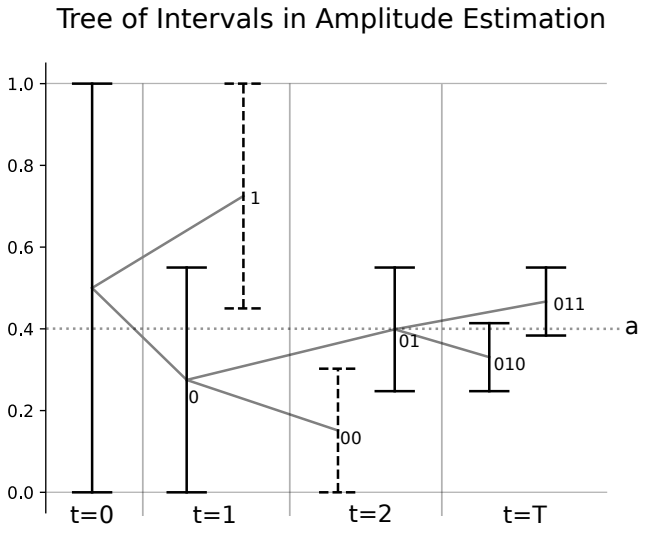


Figure 9: Sketch of the intervals considered in the proof of Theorem 23, labeled by the string $\vec{x} \in \{0,1\}^t$. At each iteration, we cut off either the right or left side of the interval which corresponds to adding either a 0 or 1 to $\vec{x}$. 'Bad' intervals not containing the true amplitude $a = 0.4$ are marked with dashed lines, and their children are not shown. The algorithm describes shrinking the interval by a factor of 0.9 with every step, but for visual clarity we use a factor of 0.55 here.

much information about a as possible. In the context of quantum signal processing, this r corresponds to some fixed 'degree budget'. We could then use machine learning to find whatever polynomial of degree $\leq r$ gives the most information about a. That is not to say that [GKLPZ20] does not consider practicality: they perform a noise analysis, and even implement their algorithm in hardware [Guirgica-Tiron&21] albeit with a trivial oracle. We merely invite the possibility of simpler algorithms with less elegant theoretical properties.

There is a reason to be skeptical of the $\beta \to 1$ limit in particular: we expect early quantum computers to have significantly slower clock speeds than classical computers. Thus, classical oracle evaluation is likely much faster than quantum oracle evaluation. So if one is going to put in all the effort of running the oracle on the quantum computer, one might as well do so as many times as possible. A perhaps more reasonable notation of hybrid quantum-classical is discussed by [Rosmanis22]: we have two different types of queries, quantum and classical, each with different cost. Unfortunately, [Rosmanis22] shows that there is no speedup for Grover's search in this setting. Is the same true for estimation?

Despite these observations, we consider the task of varying the complexity with β an excellent opportunity to display the power of the polynomial sampling framework. We now discuss the new algorithm for achieving depth $\mathbf{d} \in O(1/\varepsilon^{1-\beta})$ and overall complexity $\mathbf{D} \in O(1/\varepsilon^{1+\beta})$ based on quantum signal processing. The final result is presented in Theorem 31. While the polynomial constructions require cumbersome error analysis as usual, the core idea is quite simple. In light of the above discussion, we do not attempt a constant-factor performance analysis on this algorithm.

We take the same approach as in Theorem 23: we maintain an interval $[a_{\min}^{(t)}, a_{\max}^{(t)}]$ containing a with high probability, and shrink the interval with every iteration depending on what

side of the interval a is on. Letting $\Delta_t := a^{(t)}_{\max} - a^{(t)}_{\min}$, we test this by finding a polynomial $P(a)$ that is $\leq \frac{1}{2} - \gamma$ when a is $\leq a^{(t)}_{\min} + 0.1\Delta_t$, and is $\geq \frac{1}{2} + \gamma$ when a is $\geq a^{(t)}_{\max} - 0.1\Delta_t$. This can be achieved by making $P(a)$ approximate a line with slope $\propto \gamma/\Delta_t$, and the degree is largely determined by the slope. Then, $\propto 1/\gamma^2$ samples from the polynomial suffice to distinguish the two cases.

In the previous section we selected γ to be constant, so the degree is $\propto 1/\Delta_t$. This makes the depth and the total number of queries $\propto 1/\Delta_t$. To achieve hybrid estimation, we simply select $\gamma \propto \Delta_t^\beta$. Then the depth is $\propto \Delta_t^\beta/\Delta_t = 1/\Delta_t^{1-\beta}$. But, since we need to take $\propto 1/\gamma^2 \propto 1/\Delta_t^{2\beta}$ samples from the polynomial, the total complexity is $1/\Delta_t^{1+\beta}$. The runtime is dominated by the final iterations where $\Delta_t \propto \varepsilon$, so the final depth is $\propto 1/\varepsilon^{1-\beta}$ and the final complexity is $\propto 1/\varepsilon^{1+\beta}$ as desired.

Letting $a^{(t)}_{\text{mid}} := a^{(t)}_{\min} + 0.5\Delta_t$, we require a polynomial that approximates a line passing through $(a^{(t)}_{\text{mid}}, 1/2)$ with whatever slope we want. Once such a polynomial has been constructed the remainder of the analysis is extremely simple as shown above. However, such a polynomial is unfortunately not so easy to build because of two complications:

- Jackson's theorem (Theorem 21) is not precise enough. The line that we desire to approximate only creates a gap $\gamma \propto \Delta^\beta \in o(1)$. The approximation error better be be smaller than γ, but Jackson's theorem requires an additional factor of $1/\gamma$ in the degree to achieve this. This is too slow, so we need a more accurate polynomial approximation of a line. Fortunately, [LC17] constructed an extremely accurate approximation of $\text{erf}(kx) \approx kx$ via the Jacobi-Anger expansion.

- Semi-Pellian polynomials have fixed parity, so even though $\text{erf}(kx)$ is odd, the shifted version $\text{erf}(k(a - a_{\text{mid}}))$ has mixed parity. We can fix this by instead approximating the even function $\text{erf}(k(a - a_{\text{mid}})) + \text{erf}(-k(a + a_{\text{mid}})) + 1$, which is very similar to $\text{erf}(k(a - a_{\text{mid}}))$ for $a \geq 0$. But this only works when a_{mid} is not too close to the origin - otherwise the two erf functions interfere. This forces us to adopt special behavior for small a_{mid}, causing $O(a^{-1})$ terms in the final runtime. [GKLPZ20] treat a and β as constants in their analysis, so if we are allowed to do the same then we still reproduce their complexities.

Remark 24. *As pointed out by [MML22], it is sometimes possible to construct oracles for a shifted amplitude $\frac{a+1}{2}$. For some state $|0\rangle$ say we have a unitary U satisfying $U|0\rangle = |\Psi\rangle$ and another unitary V such that $\Pi = V|0\rangle\langle 0|V^\dagger$. Then redefine:*

$$|\bar{\Psi}\rangle := (I \oplus V^\dagger U)|+\rangle|0\rangle \tag{114}$$

$$\bar{\Pi} := |+\rangle\langle +| \otimes |0\rangle\langle 0| \tag{115}$$

Observe that $|\langle 0|V^\dagger U|0\rangle| = |\Pi|\Psi\rangle| = a$, but suppose furthermore that $\langle 0|V^\dagger U|0\rangle = a$. Then, we see that:

$$|\bar{\Pi}|\bar{\Psi}\rangle| = \left|\langle 0| \cdot (\langle +| \otimes I)(I \oplus V^\dagger U)(|+\rangle \otimes I) \cdot |0\rangle\right| \tag{116}$$

$$= \left|\langle 0| \cdot \frac{I + V^\dagger U}{2} \cdot |0\rangle\right| = \frac{1+a}{2} \tag{117}$$

If we use this oracle in our algorithm instead, then we have $a \geq 1/2$ so all the $O(a^{-1})$ terms in Theorem 31 become constant.

We begin by discussing the approximation of $\text{erf}(kx)$.

Lemma 25. *([LC17] Corollary 4). For all $k, \eta > 0$ there exists an odd polynomial $P_{erf,k,\eta}$ such that for all $x \in [-2, 2]$:*

$$|P_{erf,k,\eta}(x) - erf(kx)| \leq \eta. \tag{118}$$

Furthermore $deg(P_{erf,k,\eta}) \in O(\sqrt{(k^2 + \log(1/\eta)) \log(1/\eta)})$.

The construction in [LC17] actually obtains an approximation for $x \in [-1, 1]$, but this is easily expanded to $x \in [-2, 2]$ by dividing the input by two and then doubling k.

Now we leverage that $\text{erf}(kx)$ can be used to approximate the line kx, and is close to -1 or 1 when far from the origin. These properties are listed in the two facts below and also plotted in Figure 10. Note that the kind of approximation $\text{erf}(kx) \approx kx$ we achieve here is sufficient for interval reduction, but not sufficient for the sampling performed for the unbiased estimation algorithm from the previous section. If this had been the case, we could have exponentially improved the accuracy of the unbiased estimation algorithm.

Fact 26. *For $x \in [-1, 0]$ we have $erf(x) \leq 0.8x$. For $x \in [0, 1]$ we have $0.8x \leq erf(x)$.*

Fact 27. *For any $\tau > 0$, let $\kappa(\tau) := \frac{1}{2}\sqrt{2\ln(2/(\pi\tau^2))}$. Then:*

$$x \leq -\kappa \quad \rightarrow \quad -1 \leq erf(x) \leq -(1 - \tau) \tag{119}$$

$$\kappa \leq x \quad \rightarrow \quad 1 - \tau \leq erf(x) \leq 1 \tag{120}$$

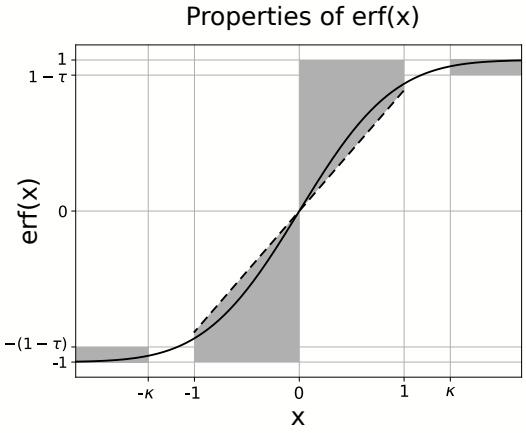


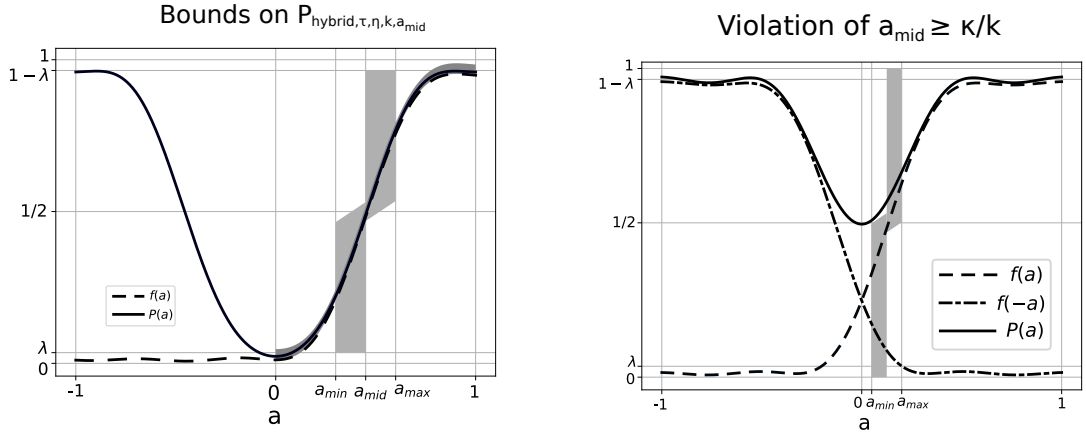
Figure 10: Sketch of the properties of $\text{erf}(x)$ stated in Facts 26 and 27. Here we selected $\tau = 0.1$, making $\kappa(\tau) \approx 1.44$. Fact 26 asserts that $\text{erf}(x)$ is either strictly above or below the dashed line $0.8x$ depending on the sign. Fact 27 asserts that $\text{erf}(x)$ is within τ of ± 1 when x is κ-far from the origin. These bounds correspond to the gray regions.

Now we discuss the trick we explained earlier that lets us approximate the mixed-parity function $\text{erf}(k(a - a_{\text{mid}}))$ with an even function on the interval $a \in [0, 1]$. Think of $f(a) \approx \text{erf}(k(a - a_{\text{mid}})) + 1$, so that $f(a)$ jumps from 0 to 1 on the interval $[a_{\min}, a_{\max}]$. Then $f_e(a) := f(a) + f(-a)$ approximates $f(a)$. We track the quality of the approximation via the parameter λ.

Fact 28. *Say there is a function that satisfies $f(a) \leq \lambda$ for $a \in [-1, 0]$, $f(a) \leq 1 - \lambda$ for $a \in [0, 1]$, and $0 \leq f(a)$ for all $a \in [-1, 1]$. Then $f_e(a) := f(a) + f(-a)$ is an even function satisfying $0 \leq f_e(a) \leq 1$ for $a \in [-1, 1]$. Furthermore, when $a \in [0, 1]$:*

$$f(a) \leq f_e(a) \leq f(a) + \lambda. \tag{121}$$

We have all the tools in place to construct the family of polynomials for hybrid estimation. A sketch of the construction is shown in Figure 11, which displays the critical property proven by the following lemma: that $P(a)$ is bounded away from $1/2$ for values close to $a_{\min}$ and $a_{\max}$. That bound will later become γ from the previous discussion. The criterion $a_{\text{mid}} \geq \kappa/k$ allows us to determine when a_{mid} is too close to the origin for the construction to work.



(a) $[a_{\min}, a_{\max}] = [0.3, 0.6]$ (b) $[a_{\min}, a_{\max}] = [0.05, 0.2]$

Figure 11: Sketch of the construction of $P_{\text{hybrid},\tau,\eta,k,a_{\text{mid}}}(a)$ or $P(a)$ for short from Lemma 29. Here we selected $\tau = \eta = 0.025$, $k = 4$, which make $\kappa \approx 1.87$, $\kappa/k \approx 0.47$ and $\deg(P_{\text{erf},k,\eta}) = 21$. In (a) we have $a_{\text{mid}} \geq \kappa/k$, so we see that $P(a)$ is contained in the gray regions asserted in (122, 123). The region at most λ above $f(a)$ is highlighted in dark gray. But in (b) where $a_{\text{mid}} < \kappa/k$, we see that $f(-a)$ is large on the interval $[a_{\min}, a_{\max}]$ causing the shape of $P(a)$ to be distorted and no longer be contained in the gray regions.

Lemma 29. *Consider any $\tau, \eta \in (0, 1)$, and let $\kappa = \kappa(\tau)$ be as in Fact 27. Consider any a_{min}, a_{max} that satisfy $0 < a_{min} < a_{max} < 1$. Furthermore, let $\Delta := a_{max} - a_{min}$ and consider any k satisfying $1 \leq k \leq 2/\Delta$. Finally, let $a_{mid} := \frac{1}{2}(a_{min} + a_{max})$ and suppose that $a_{mid} \geq \kappa/k$.*

Then there exists an even semi-Pellian polynomial $P_{hybrid,\tau,\eta,k,a_{mid}}(a)$ or $P(a)$ for short, that satisfies:

$$a_{min} \leq a \leq a_{mid} \quad \rightarrow \quad P(a) \leq \frac{1}{2} + 0.11k(a - a_{mid}) + \eta + 0.25\tau \tag{122}$$

$$a_{mid} \leq a \leq a_{max} \quad \rightarrow \quad P(a) \geq \frac{1}{2} + 0.11k(a - a_{mid}) - \eta - 0.25\tau. \tag{123}$$

This polynomial has the same degree as $P_{erf,k,\eta}(x)$ from Lemma 25.

Proof. We will obtain $P(x)$ by taking $P_{\text{erf},k,\eta}(x)$ and performing a series of degree-preserving transformations. Our first goal is to take $P_{\text{erf},k,\eta}(x)$, which is between $\pm(1 + \eta)$, and $\leq$

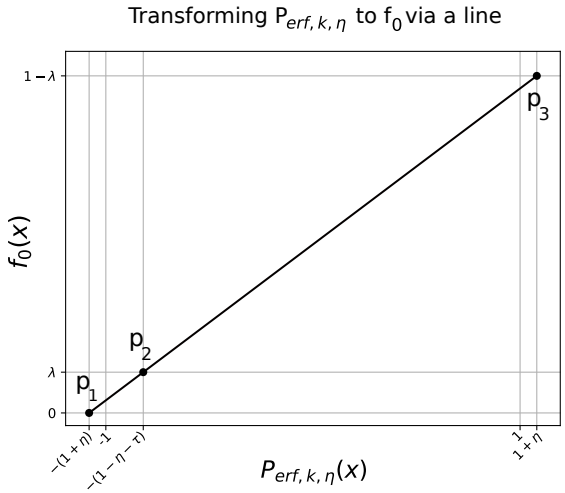


Figure 12: Line defined by the points in (124) used to transform $P_{\text{erf},k,\eta}$ to f_0 in the construction of $P_{\text{hybrid},\tau,\eta,k,a_{\text{mid}}}(a)$ from Lemma 29.

$-(1-\eta-\tau)$ for small x, and transform it to an $f_0(x)$ which is in $[0, 1-\lambda]$, and $\leq \lambda$ for small x as required by Fact 28. Let λ be chosen such that the following points are colinear:

$$\vec{p_1} := (-(1+\eta), 0); \quad \vec{p_2} := (-(1-\eta-\tau), \lambda); \quad \vec{p_2} := (1+\eta, 1-\lambda) \tag{124}$$

We find this is solved by:

$$\lambda := \frac{2\eta+\tau}{4\eta+\tau+2} \tag{125}$$

Furthermore, we find the line $y = (1+x+\eta)/(4\eta+\tau+2)$ contains these points. Accordingly, let:

$$f_0(x) := \frac{1+P_{\text{erf},k,\eta}(x)+\eta}{4\eta+\tau+2} \tag{126}$$

We want to plug a shifted f_0 into Fact 28, so we will first prove that f_0 meets some requirements for all $x \in [-2, 2]$. First, Lemma 25 implies that $P_{\text{erf},k,\eta}(x) \geq -(1+\eta)$, so since we pass through $\vec{p_1}$ we have $f_0(x) \geq 0$. Second, $P_{\text{erf},k,\eta}(x) \leq 1+\eta$, so since we pass through $\vec{p_3}$ we also have $f_0(x) \leq 1-\lambda$. Third, Fact 27 implies that for $kx \leq -\kappa$ we have $\text{erf}(kx) \leq -(1-\tau)$, so $P_{\text{erf},k,\eta}(x) \leq -(1-\eta-\tau)$, so since we pass through $\vec{p_2}$ we have $f_0 \leq \lambda$.

Now we are ready to use Fact 28. Let $f(a) := f_0(a - a_{\text{mid}})$ with $x = a - a_{\text{mid}}$. We know that $a \in [0, 1]$, but the polynomial construction must work for any $a \in [-1, 1]$. Since $a_{\text{mid}} \in [0, 1]$ it is ensured that $x \in [-2, 1] \subset [-2, 2]$ so the approximation holds. Since the function was shifted to the right, we have $f(a) \geq 0$ for $a \in [-1, 1]$, and we have $f(a) \leq 1-\lambda$ for $a \in [0, 1]$. Since we assumed $a_{\text{mid}} \geq \kappa/k$, we know that for $a \in [-1, 0]$ we have $kx = k(a - a_{\text{mid}}) \leq ka_{\text{mid}} \leq -\kappa$, which implies $f(a) \leq \lambda$. We meet the requirements of Fact 28, so let $P_{\text{hybrid},\tau,\eta,k,a_{\text{mid}}}(a) := f(a) + f(-a)$.

$P_{\text{hybrid},\tau,\eta,k,a_{\text{mid}}}(a)$, or $P(a)$ for short, is an even function by construction. Since it is a linear combination of polynomials it is itself a polynomial, and polynomials that are even

functions are even polynomials. From Fact 28, we know that $0 \leq P(a) \leq 1$, so $P(a)$ is semi-Pellian. The transformations from $P_{\mathrm{erf},k,\eta}$ to $P(a)$ were all linear, so the degree is unchanged.

It remains to show the desired bounds on $P(a)$. These stem from Fact 26. For $kx \in [-1, 0]$ we have $\mathrm{erf}(kx) \leq 0.8kx$, so $P_{\mathrm{erf},k,\eta}(x) \leq 0.8kx + \eta$, so:

$$f_0(x) \leq \frac{1 + (0.8kx + \eta) + \eta}{4\eta + \tau + 2} \tag{127}$$

$$= \frac{1}{2} + \left[\frac{1 + (0.8kx + \eta) + \eta}{4\eta + \tau + 2} - \frac{1}{2}\right] \tag{128}$$

$$= \frac{1}{2} + \frac{2(1 + 0.8kx + 2\eta) - (4\eta + \tau + 2)}{2(4\eta + \tau + 2)} \tag{129}$$

$$\leq \frac{1}{2} + \frac{\tau}{2(4 \cdot 0 + 0 + 2)} - \frac{2 \cdot 0.8(-x)}{2(4 \cdot 1 + 1 + 2)} \tag{130}$$

$$\leq \frac{1}{2} + 0.11kx - 0.25\tau \tag{131}$$

$$f_0(x) + \lambda \leq \frac{1}{2} + 0.11kx + \eta + 0.25\tau \tag{132}$$

where we plugged in $0 \leq \eta, \tau \leq 1$ to obtain a very loose but workable bound, and observed $\lambda \leq \eta + \tau/2$. Similarly, for $kx \in [0, 1]$ we have $\mathrm{erf}(kx) \geq 0.8kx - \eta$, so we derive:

$$f_0(x) \geq \frac{1 + (0.8kx - \eta) + \eta}{4\eta + \tau + 2} \tag{133}$$

$$= \frac{1}{2} + \frac{2(1 + 0.8kx) - (4\eta + \tau + 2)}{2(4\eta + \tau + 2)} \tag{134}$$

$$\geq \frac{1}{2} + \frac{2 \cdot 0.8kx}{2(4 \cdot 1 + 1 + 2)} - \frac{4\eta + \tau}{2(4 \cdot 0 + 0 + 2)} \tag{135}$$

$$\geq \frac{1}{2} + 0.11kx - \eta - 0.25\tau \tag{136}$$

The transformation $x = a - a_{\mathrm{mid}}$ makes the intervals $kx \in [-1, 0]$ and $kx \in [0, 1]$ correspond to $a \in [a_{\mathrm{mid}} - 1/k, a_{\mathrm{mid}}]$ and $a \in [a_{\mathrm{mid}}, a_{\mathrm{mid}} + 1/k]$ respectively. Since we assumed $k \leq 2/\Delta$, the intervals $[a_{\mathrm{min}}, a_{\mathrm{mid}}]$ and $[a_{\mathrm{mid}}, a_{\mathrm{max}}]$ are subintervals. From Fact 28, we thus derive the desired bounds via $f_0(x) \leq P(a) \leq f_0(x) + \lambda$. □

We have yet to take into account the Born rule: when we sample from $P(a)$, we actually toss a coin that comes up heads with probability $|P(a)|^2$. Fortunately, as the following lemma shows, it suffices to bound x away from $1/2$ even when tossing a coin with bias x^2 in order to decide if x is big or small.

Lemma 30. *Say there is a coin that comes up heads with probability x^2, and for any $\gamma, \delta > 0$ let $A_{t,\gamma,\delta}(x)$ be the probability that more than $\frac{1}{4} + \gamma^2$ of $\lceil \frac{1}{2}\gamma^{-2} \ln(\delta^{-1}) \rceil$ tosses come up heads. Then:*

$$\textit{if } x \geq \frac{1}{2} + \gamma \textit{ then } A_{\gamma,\delta}(x) \geq 1 - \delta, \tag{137}$$

$$\textit{if } x \leq \frac{1}{2} - \gamma \textit{ then } A_{\gamma,\delta}(x) \leq \delta. \tag{138}$$

Proof. We can derive $x \geq \frac{1}{2}+\gamma$ implies $x^2 \geq \frac{1}{4}+\gamma^2+\gamma$, and $x \leq t-\gamma$ implies $x^2 \leq \frac{1}{4}+\gamma^2-\gamma$. The claims on $A_{t,\gamma,\delta}$ follow from two applications of the Chernoff-Hoeffding theorem. Let $\mathbf{S}$ be the random variable denoting the number of heads, and let m be the number of tosses. Then, if $x^2 \geq \frac{1}{4}+\gamma^2+\gamma$:

$$\Pr\left[\mathbf{S} \geq m\left(\frac{1}{4}+\gamma^2\right)\right] \leq \Pr[\mathbf{S} \geq \mathbb{E}(\mathbf{S})+\gamma m] \leq e^{-2\gamma^2 m} \tag{139}$$

Demanding $e^{-2\gamma^2 m} \leq \delta$ and solving for m completes the proof. A similar argument follows for the bound when $x^2 \leq \frac{1}{4}+\gamma^2-\gamma$. □

Having constructed the polynomial and shown how to use it to distinguish $a \leq a_{\min}^{(t)}+0.1\Delta_t$ and $a \geq a_{\max}^{(t)}-0.1\Delta_t$, all that remains to show is how to deal with the restriction that we must have $a_{\text{mid}}^{(t)} \geq \kappa/k$ for the polynomial construction to work properly. Recall that we would ideally like to pick the slope $k \propto 1/\Delta_t^{1-\beta}$. Since κ is roughly constant, this essentially translates the condition $a_{\text{mid}}^{(t)} \geq \kappa/k$ to $a_{\text{mid}}^{(t)} \geq \frac{1}{2}\Delta_t^{1-\beta}$. If this is not the case, then we must go 'full quantum' ($\beta = 0$) and set the slope to be $k \propto 1/\Delta_t$ instead. Then the condition is $a_{\text{mid}}^{(t)} \geq \frac{1}{2}\Delta_t$ which is always true.

We find that the consequences of this behavior are to knock out an extra $O(a^{-1})$ term in the total query complexity and an extra $O(a^{-1/(1-\beta)})$ term in the depth. Intuitively, these arise as follows. At iteration t we have $a \in [a_{\min}^{(t)}, a_{\max}^{(t)}]$, so, implying that a_{mid} is lower bounded by $a-\Delta_t/2 \geq a-\Delta_t^{1-\beta}/2$. At a certain time t^*, Δ_t becomes small enough ($\Delta_t^{1-\beta} \leq a$) so that the bound $a_{\text{mid}}^{(t)} \geq \frac{1}{2}\Delta_t^{1-\beta}$ is assured. So, the algorithm divides into two phases. In the first phase ($t \leq t^*$) we have $\Delta_t^{1-\beta} \geq a$ and the algorithm is essentially fully quantum. The complexity and depth are then given by $O(a^{-1})$ and $O(a^{-1/(1-\beta)})$ respectively. In the second phase ($t > t^*$) we are guaranteed to have $a_{\text{mid}}^{(t)} \geq \kappa/k$, so the algorithm has the desired complexity and depth of $O(1/\varepsilon^{1+\beta})$ and $O(1/\varepsilon^{1-\beta})$ respectively.

Recall that $\mathbf{d}$ is the maximum degree of any particular polynomial, and that $\mathbf{D}$ is the sum of the degrees of all the polynomials. These correspond to query depth and query complexity respectively.

Theorem 31. ***Hybrid quantum-classical estimation via polynomial sampling.*** *For any $\varepsilon, \delta > 0$ and $0 \leq \beta < 1$, there exists an amplitude estimation algorithm satisfying:*

$$Pr[|\hat{\mathbf{a}}-a| \geq \varepsilon] \leq \delta \tag{140}$$

$$\exists f(\varepsilon,a,\delta) \in \tilde{O}((a^{-1}+\varepsilon^{-(1+\beta)})\log(\delta^{-1})) \text{ such that } \Pr[\mathbf{D} \geq f(\varepsilon,a,\delta)] \leq \delta \tag{141}$$

$$\exists g(\varepsilon,a) \in \tilde{O}(a^{-1/(1-\beta)}+\varepsilon^{-(1-\beta)}) \text{ such that } \Pr[\mathbf{d} \geq g(\varepsilon,a)] \leq \delta \tag{142}$$

Here, $\tilde{O}$ hides log factors in ε.

Proof. The algorithm proceeds as follows:

1. Initialize $a_{\min}^{(0)} \leftarrow 0$ and $a_{\max}^{(0)} \leftarrow 1$. Let $\Delta_t := a_{\max}^{(t)} - a_{\min}^{(t)}$ and $a_{\text{mid}}^{(t)} = a_{\min}^{(t)} + 0.5\Delta_t$ throughout.

2. Let $T := \lceil \log_{0.9}(\varepsilon) \rceil$. For $t = 0, 1, 2, ..., T-1$, do:

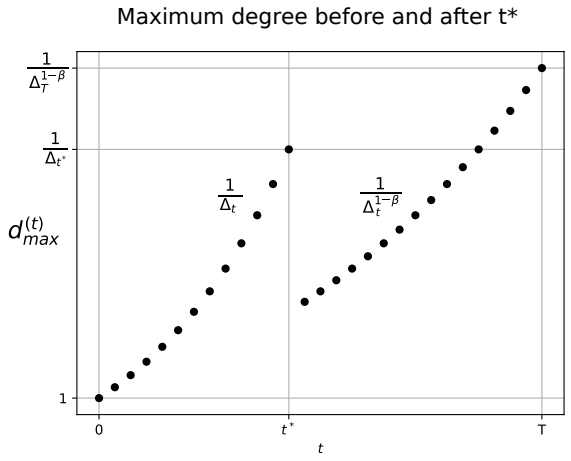


Figure 13: Sketch of an argument for bounding $\mathbf{d} := \max_t d_{\max}^{(t)}$ in Theorem 31 in the case when $T > t^*$. When $t \leq t^*$ then there is a chance that $a_{\text{mid}}^{(t)} < \frac{1}{2}\Delta_t^{1-\beta}$ so we cannot rule out the more expensive scaling of $d_{\max}^{(t)} \in O(\Delta_t^{-1})$. But after $t > t*$, we have the cheaper scaling of $\tilde{O}(\Delta_t^{-(1-\beta)})$. So the maximum must occur at either $t = t^*$ or $t = T$.

(a) If $a_{\text{mid}}^{(t)} \geq \frac{1}{2}\Delta_t^{1-\beta}$, set:

$$\eta_t, \tau_t := 0.01 \cdot \Delta_t^{\beta}, \tag{143}$$

$$k_t := \frac{\kappa(\tau_t)}{2} \frac{1}{\Delta_t^{1-\beta}}, \tag{144}$$

$$\gamma_t := 0.01 \cdot \Delta_t^{\beta} \tag{145}$$

Otherwise, set:

$$\eta_t, \tau_t := 0.01, \tag{146}$$

$$k_t := \frac{\kappa(\tau_t)}{2} \frac{1}{\Delta_t}, \tag{147}$$

$$\gamma_t := 0.01 \tag{148}$$

(b) Obtain $P_t(a)$ from Lemma 29. Set $\delta_t := \delta/T$ and $m_t := \lceil \frac{1}{2}\gamma_t^{-2} \ln(\delta_t^{-1}) \rceil$ and sample from $\mathbf{b}_{P_t}$ a total of m_t times and let $\mathbf{S}^{(t)}$ be the total.

(c) If $\mathbf{S}^{(t)} > m_t \left(\frac{1}{4} + \gamma_t^2\right)$, then set:

$$[a_{\min}^{(t+1)},\ a_{\max}^{(t+1)}] \leftarrow [a_{\min}^{(t)} + 0.1\Delta_t,\ a_{\max}^{(t)}] \tag{149}$$

Otherwise, set:

$$[a_{\min}^{(t+1)},\ a_{\max}^{(t+1)}] \leftarrow [a_{\min}^{(t)},\ a_{\max}^{(t)} - 0.1\Delta_t] \tag{150}$$

3. Let $\hat{\mathbf{a}} \leftarrow a_{\text{mid}}^{(T)}$ (or anything else in the interval $[a_{\min}^{(T)}, a_{\max}^{(T)}]$).

We begin by proving correctness. At the beginning we have $\Delta_0 = 1$, and each of the sub-steps of step 2 sets $\Delta_{t+1} = 0.9\Delta_t$, so $\Delta_t = 0.9^t$. At the end we have $\Delta_T = 0.9^T \leq \varepsilon$, so, provided $a \in [a_{\min}^{(T)}, a_{\max}^{(T)}]$, anything inside the interval $[a_{\min}^{(T)}, a_{\max}^{(T)}]$ is within ε of a. So the algorithm returns correctly if at all iterations of step 2 we have $a \in [a_{\min}^{(t)}, a_{\max}^{(t)}]$.

We will show that the probability step 2c) adjusts the interval in a way to make $a \notin [a_{\min}^{(t+1)}, a_{\max}^{(t+1)}]$ is at most δ_t. Then, by the union bound, the probability any of the steps fail is δ.

There are two ways we could fail to have $a \notin [a_{\min}^{(t+1)}, a_{\max}^{(t+1)}]$ after step 2c): either we have $a \in [a_{\min}^{(t)}, a_{\min}^{(t)} + 0.1\Delta_t]$ and $\mathbf{S}^{(t)} > m_t(\frac{1}{4} + \gamma_t^2)$, or we have $a \in [a_{\max}^{(t)} - 0.1\Delta_t, a_{\max}^{(t)}]$ and $\mathbf{S}^{(t)} < m_t(\frac{1}{4} + \gamma_t^2)$. To show that these are both unlikely, we invoke Lemma 30. Say $a \in [a_{\min}^{(t)}, a_{\min}^{(t)} + 0.1\Delta_t]$ implies $P_t(a) \leq \frac{1}{2} - \gamma_t$, and $a \in [a_{\max}^{(t)} - 0.1\Delta_t, a_{\max}^{(t)}]$ implies $P_t(a) \geq \frac{1}{2} + \gamma_t$. Then Lemma 30 asserts that any of the events causing $a \notin [a_{\min}^{(t+1)}, a_{\max}^{(t+1)}]$ happen with probability at most δ_t.

So all that remains is to show the bounds on $P_t(a)$ in terms of γ_t. We can show this using Lemma 29, provided we meet its requirements. The requirement that $1 \leq k_t \leq 2/\Delta_t$ is always satisfied. But the requirement that $a_{\text{mid}}^{(t)} \geq \kappa(\tau_t)/k_t$, demanded that in step 2a) we select the parameters conditionally.

We begin with the case where $a_{\text{mid}}^{(t)} \geq \frac{1}{2}\Delta_t^{1-\beta}$. Here, we see that we selected $k_t := \frac{1}{\Delta_t^{1-\beta}}\frac{\kappa(\tau_t)}{2}$, implying $\kappa(\tau_t)/k_t = \frac{1}{2}\Delta_t^{1-\beta}$. So we have $a_{\text{mid}}^{(t)} \geq \kappa(\tau_t)/k_t$ as desired. Also observe that $\tau_t < 1/4$, so $\kappa(\tau_t) > 1$, so $k_t \geq \frac{1}{2}\frac{1}{\Delta_t^{1-\beta}}$. Then, Lemma 29 demonstrates:

$$a \in [a_{\min}^{(t)}, a_{\min}^{(t)} + 0.1\Delta_t] \quad \rightarrow \quad P_t(a) \leq \frac{1}{2} - 0.11k_t(0.4\Delta_t) + \eta_t + 0.25\tau_t \tag{151}$$

$$\leq \frac{1}{2} - \left(0.11 \cdot \frac{1}{2} \cdot 0.4 - 0.01 - 0.25 \cdot 0.01\right) \cdot \Delta_t^{\beta} \tag{152}$$

$$\leq \frac{1}{2} - 0.01 \cdot \Delta_t^{\beta} = \frac{1}{2} - \gamma_t \tag{153}$$

$$a \in [a_{\max}^{(t)} - 0.1\Delta_t, a_{\max}^{(t)}] \quad \rightarrow \quad P_t(a) \geq \frac{1}{2} + 0.11k_t(0.4\Delta_t) - \eta_t - 0.25\tau_t \tag{154}$$

$$\geq \frac{1}{2} + \gamma_t \tag{155}$$

In the case where $a_{\text{mid}}^{(t)} \geq \frac{1}{2}\Delta_t^{1-\beta}$, we instead select $\eta_t, \tau_t = 0.001$, implying $\kappa(\tau_t) > 1$ as before. That way $\frac{\kappa(\tau_t)}{k_t} \leq \frac{1}{2}\Delta_t$. We always have $\frac{1}{2}\Delta_t \leq a_{\text{mid}}^{(t)}$, so the conditions of Lemma 29

are satisfied. So, we derive:

$$a \in [a^{(t)}_{\min}, a^{(t)}_{\min} + 0.1\Delta_t] \quad \rightarrow \quad P_t(a) \leq \frac{1}{2} - 0.11\left(\frac{1}{2}\cdot\frac{1}{\Delta_t}\right)(0.4\Delta_t) + \eta_t + 0.25\tau_t \tag{156}$$

$$\leq \frac{1}{2} - \left(0.11\cdot\frac{1}{2}\cdot 0.4 - 0.01 - 0.25\cdot 0.01\right) \tag{157}$$

$$\leq \frac{1}{2} - 0.01 \geq \frac{1}{2} - \gamma_t \tag{158}$$

$$a \in [a^{(t)}_{\max} - 0.1\Delta_t, a^{(t)}_{\max}] \quad \rightarrow \quad P_t(a) \geq \frac{1}{2} + 0.11\left(\frac{1}{2}\cdot\frac{1}{\Delta_t}\right)(0.4\Delta_t) - \eta_t - 0.25\tau_t \tag{159}$$

$$\geq \frac{1}{2} + 0.01 \geq \frac{1}{2} + \gamma_t \tag{160}$$

So in both cases we meet the assumptions of Lemma 30, completing the correctness portion of the proof.

Now we prove the properties of $\mathbf{D}$ and $\mathbf{d}$. To do so, we bound the maximum degree $d^{(t)}_{\max}$ and the sum of all the degrees $d^{(t)}$ at each iteration of step 2. The following bounds rely on the guarantee that $a \in [a^{(t)}_{\min}, a^{(t)}_{\max}]$ for all t, which is only true probabilistically. Hence the bounds are themselves probabilistic.

We start with $\mathbf{d}$ via $d^{(t)}_{\max}$. The degree $d^{(t)}_{\max}$ is given by Lemma 25, and since we always have $\log(1/\eta_t) \leq k_t$ we have $d^{(t)}_{\max} \in O(k_t\sqrt{\log(1/\eta_t)})$. If $a^{(t)}_{\text{mid}} \geq \frac{1}{2}\Delta_t^x$, then we have $d^{(t)}_{\max} \in O(\Delta_t^{-(1-\beta)}\sqrt{\log(\Delta_t^{-2\beta})}\cdot\sqrt{\log(\Delta_t^{-\beta})}) \leq \tilde{O}(\Delta_t^{-(1-\beta)})$ where $\tilde{O}$ hides log factors in Δ_t. Otherwise if $a^{(t)}_{\text{mid}} < \frac{1}{2}\Delta_t^{1-\beta}$, we have $d^{(t)}_{\max} \leq O(\Delta_t^{-1})$.

We have $\mathbf{d} = \max_t d^{(t)}_{\max}$. To maximize over all t, we must analyze the condition $a_{\text{mid}} \geq \frac{1}{2}\Delta_t^{1-\beta}$, since when this is the case we have the cheaper degree of $O(\Delta_t^{-(1-\beta)})$. We argue that there is some t^* such that for $t > t^*$ the algorithm is guaranteed to satisfy $a^{(t)}_{\text{mid}} \geq \frac{1}{2}\Delta_t^{1-\beta}$.

Observe that $a - \Delta_t/2 \leq a^{(t)}_{\text{mid}} \leq a + \Delta_t/2$, and since $\Delta_t \leq \Delta_t^{1-\beta}$ we also have $a^{(t)}_{\text{mid}} \geq a - \Delta_t^{1-\beta}/2$. That means that as soon as $\frac{1}{2}\Delta_t^{1-\beta} \geq a - \Delta_t^{1-\beta}/2$, we are guaranteed that $a^{(t)}_{\text{mid}} \geq \frac{1}{2}\Delta_t^{1-\beta}$. This happens when $a \geq \Delta_t^{1-\beta}$, that is, when $t > t^* := \lceil \log_{0.9}(a) \rceil$.

Let's assume $T > t^*$. Before t^* we have $d^{(t)}_{\max} \in O(\Delta_t^{-1})$, and afterwards we have $d^{(t)}_{\max} \in \tilde{O}(\Delta_t^{-(1-\beta)})$. Both of these functions are increasing in t, so we maximize $d^{(t)}_{\max}$ over all t by just considering their values at t^* and T. See Figure 13. We saw above that at t^* we have $\Delta_t \geq a^{1/(1-\beta)}$, and at T we have $\Delta_t \leq \varepsilon$. So we have established:

$$\max_t d^{(t)}_{\max} \in \max\left[O(a^{-1/(1-\beta)}), \tilde{O}(\varepsilon^{-(1-\beta)})\right] = \tilde{O}\left(a^{-1/(1-\beta)} + \varepsilon^{-(1-\beta)}\right) \tag{161}$$

where the final $\tilde{O}$ hides log factors in ε, but not log factors in $a^{1/(1-\beta)}$.

Otherwise, assume $T \leq t^*$. Then we have $d^{(t)}_{\max} \in O(\Delta_t^{-1})$ always, and since this function is increasing the maximum occurs at $t = T$. But at $t \leq t^*$ we have $a < \Delta_t^{1-\beta}$, which implies $\Delta_t^{-1} \leq a^{-1/(1-\beta)}$. So in this case we again have $\max_t d^{(t)}_{\max} \in O(a^{-1/(1-\beta)})$. This establishes the bound on $\mathbf{d}$ for both cases.

Now we bound the sum of the degrees $\mathbf{D}$ via the sum of all degrees at a particular iteration $d^{(t)}$. We simply have $d^{(t)} = m_t d^{(t)}_{\max}$, so in the $t \leq t^*$ case we have $d^{(t)} \in O(\Delta_t^{-1}\log(\delta_t^{-1}))$, and in the $t > t^*$ case we have $d^{(t)} \in \tilde{O}(\Delta_t^{-(1-\beta)}\cdot\Delta_t^{-2\beta}\log(\delta_t^{-1})) = \tilde{O}(\Delta_t^{-(1+\beta)}\log(\delta_t^{-1}))$.

Since δ_t is independent of t, we can just bound the first factor in the products, recalling that $\Delta_t = 0.9^t$:

$$\sum_{t=0}^{t^*} \Delta_t^{-1} + \sum_{t=t^*+1}^{T-1} \Delta_t^{-(1+\beta)} \leq \frac{1}{0.9} \sum_{t=0}^{t^*-1} \Delta_t^{-1} + \sum_{t=0}^{T-1} \Delta_t^{-(1+\beta)} \tag{162}$$

$$= \frac{1}{0.9} \frac{0.9^{-t*} - 1}{0.9^{-1} - 1} + \frac{(0.9^T)^{-(1+\beta)} - 1}{0.9^{-(1+\beta)} - 1} \tag{163}$$

$$\in O(a^{-1} + \varepsilon^{-(1+\beta)}) \tag{164}$$

Looking at the $\log(\delta_t^{-1})$ term, we have $T \in O(\log(1/\varepsilon))$, so we have $\log(\delta_t^{-1}) \in \tilde{O}(\log(\delta^{-1}))$ where the $\tilde{O}$ hides log factors in ε. This establishes the bound on $\mathbf{D}$. □

6 Acknowledgements

The authors are grateful for fruitful discussions with Stefan Woerner, Julien Gacon, Giacomo Nannicini, John Martyn, Nikitas Stamatopoulos, and Pawel Wocjan. Part of this work was completed while PR was supported by Scott Aaronson's Vannevar Bush Faculty Fellowship, although the majority of the work was performed at IBM Quantum.

References

[CH22] Arjan Cornelissen, Yassine Hamoudi **A Sublinear-Time Quantum Algorithm for Approximating Partition Functions** arXiv:2207.08643 Proceedings of the 34th Symposium on Discrete Algorithms (SODA) (2022)

[vACGN22] Joran van Apeldoorn, Arjan Cornelissen, András Gilyén, Giacomo Nannicini **Quantum tomography using state-preparation unitaries** arXiv:2207.08800 Proceedings of the 34th Symposium on Discrete Algorithms (SODA) (2022)

[Zhao&22] Yunpeng Zhao, Haiyan Wang, Kuai Xu, Yue Wang, Ji Zhu, Feng Wang **Adaptive Algorithm for Quantum Amplitude Estimation** arXiv:2206.08449 (2022)

[MML22] Alberto Manzano, Daniele Musso, Álvaro Leitao **Real Quantum Amplitude Estimation** arXiv:2204.13641 EPJ Quantum Technol. 10, 2 (2022)

[Rosmanis22] Ansis Rosmanis **Hybrid Quantum-Classical Search Algorithms** arXiv:2202.11443 (2022)

[WDL21] Jiasu Wang, Yulong Dong, Lin Lin **On the energy landscape of symmetric quantum signal processing** arXiv:2110.04993 Quantum 6, 850 (2021)

[LdW21] Noah Linden, Ronald de Wolf **Average-Case Verification of the Quantum Fourier Transform Enables Worst-Case Phase Estimation** arXiv:2109.10215 Quantum 6, 872 (2021)

[Guirgica-Tiron&21] Tudor Giurgica-Tiron, Sonika Johri, Iordanis Kerenidis, Jason Nguyen, Neal Pisenti, Anupam Prakash, Ksenia Sosnova, Ken Wright, William Zeng **Low depth amplitude estimation on a trapped ion quantum computer** arXiv:2109.09685 Physical Review Research 4, 033034 (2021)

[MRTC21] John M. Martyn, Zane M. Rossi, Andrew K. Tan, Isaac L. Chuang **A Grand Unification of Quantum Algorithms** arXiv:2105.02859 PRX Quantum 2, 040203 (2021)

[Rall21] Patrick Rall **Faster Coherent Quantum Algorithms for Phase, Energy, and Amplitude Estimation** arXiv:2103.09717 Quantum 5, 566 (2021)

[GKLPZ20] Tudor Giurgica-Tironc, Iordanis Kerenidisa, Farrokh Labibd, Anupam Prakash, and William Zeng **Low depth algorithms for quantum amplitude estimation** arXiv:2012.03348 Quantum 6, 745 (2020)

[VO20] Ramgopal Venkateswaran, Ryan O'Donnell **Quantum Approximate Counting with Nonadaptive Grover Iterations** arXiv:2010.04370 38th International Symposium on Theoretical Aspects of Computer Science (STACS) (2020)

[AHNTW20] Srinivasan Arunachalam, Vojtech Havlicek, Giacomo Nannicini, Kristan Temme, Pawel Wocjan **Simpler (classical) and faster (quantum) algorithms for Gibbs partition functions** arXiv:2009.11270 Quantum 6, 789 (2020)

[YLRJ20] Kwangmin Yu, Hyunkyung Lim, Pooja Rao, Dasol Jin **Comparison of Amplitude Estimation Algorithms by Implementation** arXiv:2005.05300 (2020)

[Chao&20] Rui Chao, Dawei Ding, Andras Gilyen, Cupjin Huang, Mario Szegedy **Finding Angles for Quantum Signal Processing with Machine Precision** arXiv:2003.02831 (2020)

[Nakaji20] Kouhei Nakaji **Faster Amplitude Estimation** arXiv:2003.02417 QIC20.13-14-2 (2020)

[LT20] Lin Lin, Yu Tong. **Near-optimal ground state preparation** Quantum 4, 372 arXiv:2002.12508 (2020)

[GGZW19] Dmitry Grinko, Julien Gacon, Christa Zoufal, Stefan Woerner **Iterative Quantum Amplitude Estimation** npj Quantum Inf 7, 52 arXiv:1912.05559 (2019)

[AR19] Scott Aaronson, Patrick Rall. **Quantum Approximate Counting, Simplified** Symposium on Simplicity in Algorithms. 2020, 24-32 arXiv:1908.10846 (2019)

[HW19] Aram W. Harrow, Annie Y. Wei. **Adaptive Quantum Simulated Annealing for Bayesian Inference and Estimating Partition Functions** Proc. of SODA 2020 arXiv:1907.09965 (2019)

[Suzuki&19] Yohichi Suzuki, Shumpei Uno, Rudy Raymond, Tomoki Tanaka, Tamiya Onodera, Naoki Yamamoto **Amplitude estimation without phase estimation** arXiv:1904.10246 Quantum Information Processing, 19, 75 (2019)

[Haah18] Jeongwan Haah **Product Decomposition of Periodic Functions in Quantum Signal Processing** Quantum 3, 190. arXiv:1806.10236 (2018)

[GSLW18] András Gilyén, Yuan Su, Guang Hao Low, Nathan Wiebe **Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics** arXiv:1806.01838 (2018)

[GSLW19] András Gilyén, Yuan Su, Guang Hao Low, Nathan Wiebe **Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics** Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing (STOC 2019) Pages 193–204 (2019)

[LC17] Guang Hao Low, Isaac L. Chuang **Hamiltonian Simulation by Uniform Spectral Amplification** arXiv:1707.05391 (2017)

[LC1610] Guang Hao Low, Isaac L. Chuang **Hamiltonian Simulation by Qubitization** Quantum 3, 163 arXiv:1610.06546 (2016)

[LC1606] Guang Hao Low, Isaac L. Chuang **Optimal Hamiltonian Simulation by Quantum Signal Processing** Phys. Rev. Lett. 118, 010501 arXiv:1606.02685 (2016)

[CO16] Earl T. Campbell, Joe O'Gorman **An efficient magic state approach to small angle rotations** Quantum Science and Technology, 1, 015007 arXiv:1603.04230 (2016)

[LYC16] Guang Hao Low, Theodore J. Yoder, Isaac L. Chuang **The methodology of resonant equiangular composite quantum gates** arXiv:1603.03996 Phys. Rev. X 6, 041067 (2016)

[Montanaro15] Ashley Montanaro **Quantum speedup of Monte Carlo methods** Proc. Roy. Soc. Ser. A, vol. 471 no. 2181 arXiv:1504.06987 (2015)

[YLC14] Theodore J. Yoder, Guang Hao Low, Isaac L. Chuang **Fixed-point quantum search with an optimal number of queries** arXiv:1409.3305 Phys. Rev. Lett. 113, 210501 (2014)

[AKLV13] Itai Arad, Alexei Kitaev, Zeph Landau, Umesh Vazirani "An area law and subexponential algorithm for 1D systems" arXiv:1301.1162

[Demeyer07] J. Demeyer "Diophantine Sets over Polynomial Rings and Hilbert's Tenth Problem for Function Fields" PhD Thesis at Universiteit Gent. (2007)

[BHMT00] Gilles Brassard, Peter Hoyer, Michele Mosca, Alain Tapp **Quantum Amplitude Amplification and Estimation** Quantum Computation and Quantum Information, 305:53-74 arXiv:quant-ph/0005055 (2000)

[NW98] Ashwin Nayak, Felix Wu. **The quantum query complexity of approximating the median and related statistics** arXiv:quant-ph/9804066 Proceedings of the 31st Annual ACM SIGACT Symposium on Theory of Computing (STOC 1999) Pages 384-393 (1998)

[Rivlin69] Theodore Rivlin **An Introduction to the Approximation of Functions** SIAM Review Vol. 12, Iss. 2 Dover Publications, Inc. New York. (1969)

[CP34] C. Clopper, E. Pearson **The use of confidence or fiducial limits illustrated in the case of the binomial,** Biometrika, vol. 26, no. 4, pp. 404–413. (1934)