The Dissertation Committee for Patrick Julian Tassilo Rall
certifies that this is the approved version of the following dissertation:

Quantum Computers and Monte Carlo Estimation

Committee:

Scott Aaronson, Supervisor

Elena Caceres

Allan MacDonald

Eric Price

Quantum Computers and Monte Carlo Estimation

by

Patrick Julian Tassilo Rall

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

December 2021

This dissertation is dedicated to my mother Anna Maria Rall.

Acknowledgments

My journey through graduate school was an amazing experience. I was brought into contact with fascinating and challenging research problems, and grew both as a scientist and as a person. This would not have been possible without the many people in my life that supported me along the way.

First and foremost, I want to thank my wife Stella Wang for going on this journey with me. As we progressed through our studies, we shared the ideas we were working on and consulted each other's opinions on what to do next. She always took the time to listen to my ramblings and deeply consider any tough decisions, which must have taken an enormous amount of patience. My gratitude for her support and love is immeasurable. The same goes to the rest of my family, especially my mother Anna Maria, who was always there to listen and give me advice.

Dr. Scott Aaronson is, of course, the reason any of this was possible in the first place. After joining the UT Physics department's graduate program, I really wanted to work on quantum computation. But back then there were very few professors there that made the subject their focus. So, I was absolutely ecstatic when Dr. Aaronson took me as a student, despite being a professor in the computer science department. Over the last six years, Dr. Aaronson's wisdom and intuition were an invaluable guide to my research projects and illuminated my perspective of the field as a whole. I also had the joy of being a teaching assistant for both of his courses on quantum computing, which boosted the stability of my understanding of the fundamentals. Being a graduate student of Dr. Aaronson is an enormous and unforgettable privilege.

With one foot in the physics program and another in the CS program, I had the additional privilege of being taught the fundamentals of both areas. The UT Physics graduate program provides each and every student with a world-class baseline of knowledge, and I want to thank Dr. Richard Hazeltine, Dr. Gregory Fiete and Dr. Allan MacDonald for teaching courses of excellent quality. But easily the most valuable course I took at UT was Dr. Eric Price's course on randomized algorithms. Quantum algorithms are themselves randomized algorithms after all, and the techniques I learned during that intense semester form the backbone of basically every paper in this dissertation. The analysis of these algorithms quickly becomes a maze of coin tosses and branching paths, and demands a strong intuition to not get lost. I got lost countless times, but Dr. Price and his student Dr. John Kallaugher were always there to help untie my knotted thoughts.

I still consider myself a physicist. But without the help of the professors at the UT Physics department, I might have floated off to somewhere else. Dr. Elena Caceres graciously let me attend (and even present at) her reading group on quantum gravity, letting me learn about the enormously exciting connections between theoretical physics and quantum information theory. Feynman's vision for quantum computers was that they would be an tool for studying physics. But it is amazing that even just thinking about quantum computers can already help us learn more about our universe!

We would also hope that quantum computers can actually run programs that are useful for studying physics. But somehow, so many of the problems we consider in quantum algorithms are drawn from the computer science literature. It turns out that identify-

ing questions in other fields of physics and translating them into a format that quantum computers can understand is already a significant challenge. Deciding to take on this interdisciplinary challenge for one of my papers, I interviewed Dr. Allan MacDonald and Dr. Elaine Li. The two of them very patiently explained the computational tools of their research areas, and helped me connect back to the challenges for which a quantum computer was originally envisioned. Without their help, this project would not have had any ground to stand on.

At the beginning of my time at UT Austin, when I was still finding my footing as a scientist, I needed a lot of support from other scientists in order to try out different areas of research and develop some essential skills. Dr. Brian La Cour and Dr. Antia Lamas-Linares were extremely generous in giving me opportunities to wet my feet. Dr. La Cour gave me access to his optics research lab at the Applied Research Laboratory, where I could study quantum state tomography. Dr. Antia Lamas-Linares helped me operate the Stampede 2 supercomputer at the Texas Advanced Computing Center to simulate quantum circuits. These experiences were absolutely vital to my journey.

I recall the early days in 2017 when all of Dr. Aaronson's research group fit into that tiny conference room next to my desk at the Gates Dell Complex. Since then, the group has grown into a vibrant research environment with many brilliant graduate students and postdoctoral scholars for me to pester with my questions. My fellow students gave me enormous amounts of their time and patience, and many even took the time to carefully proof read my manuscripts. My first published research paper grew out of a collaboration with Dr. Aaronson's students Daniel Liang and William Kretschmer, and without their brilliance that project would not have been possible. Other students and postdocs that

were of great help with the work presented in this dissertation were Bryce Fuller, Justin Yirka, Chunhao Wang, Yosi Atia, Andrew Tan, Adrian Trejo Nuñes, Sabee Grewal, Devon Ingram, and David McKnight.

Finally, it is an enormous privilege to work in such a supportive and welcoming research area. After a while, I realized that so many scientists in quantum information are incredibly friendly and willing to dedicate you their time—all you must do is ask! The connections and insights drawn from the resulting conversations eventually turned into the core ideas behind many of the results presented in this dissertation. In particular, I would like to thank Dr. David Gross, Dr. Earl Campbell, Dr. Mark Howard, James Seddon, Dr. Paul Burchard, Dr. C. R. Wie, Dr. Ronald de Wolf, Dr. Naoki Yamamoto, Xu Guoliang, Senguoan Song, Hanqing Wu, Dr. Andras Gilén, Dr. Justin Thaler, Dr. Andrew Potter, Dr. Alexander Weisse, Dr. Alesandro Roggero, Philip Jensen, Lasse Bjørn Kristensen, Dr. Alán Aspuru-Guzik, Dr. Pawel Wocjan, and Dr. William Zeng for their time and insight.

Quantum Computers and Monte Carlo Estimation

Publication No. ____________________

Patrick Julian Tassilo Rall, Ph.D.
The University of Texas at Austin, 2021

Supervisor: Scott Aaronson

Quantum algorithms have a fascinating relationship with randomness. On the one hand, the dynamics of quantum circuits and quantum physics defy description with classical probability. This is, in some sense, because quantum mechanics requires the use of 'negative probabilities' to describe what is going on. On the other hand, it is these very same negative probabilities that lend quantum computers their power. While classical Monte Carlo strategies fail to efficiently simulate arbitrary quantum computations, quantum computers can be used to speed up the performance of Monte Carlo algorithms.

In this dissertation we explore the connections between Monte Carlo estimation and quantum computation. We present several quantum algorithms for Monte Carlo estimation tasks in physics, such as estimating physical quantities and measuring observables. We also present classical algorithms that simulate quantum computers using Monte Carlo strategies that succeed in avoiding a sign problem for an interesting family of circuits.

We begin by deepening our understanding of the seminal result by Brassard, Høyer, Mosca, and Tapp [BHMT00]: while classical Monte Carlo estimation requires $O(\varepsilon^{-2})$ sam-

ples to estimate a quantity to accuracy ε, quantum computers merely require $O(\varepsilon^{-1})$. While this algorithm is extremely widely used, it is also very complicated. We give a new, simpler quantum algorithm for quantum estimation. We find that the problem reduces to estimating a parameter θ, given access to a coin with bias $\sin^2(n\theta)$.

Next, we present a review of block encodings. Block encodings are a recently developed technique for designing quantum algorithms that permit us to use non-unitary matrices. We also review singular value transformation, a powerful technique for applying functions to the singular values of block encoded matrices. While [GSLW18] presents a comprehensive analysis of these techniques, we find that there is a need in the literature for a more accessible introduction to the subject. Several results in this dissertation rely on block encodings and singular value transformation.

We proceed to use the power of block encodings to construct quantum algorithms for the estimation of some physical quantities. We give a subroutine that, given access to a block encoding of O can estimate the expectation $\langle O\rangle$. This subroutine harnesses the quadratic speedup for Monte Carlo estimation presented earlier. We then build algorithms for computing n-time correlation functions, the density of states of a Hamiltonian, and linear response functions.

Another fundamental task in simulating physical systems is measuring in the eigenbasis of an observable O and extracting an eigenvalue. This is usually performed via a very complicated method: we pretend that O is a Hamiltonian, perform Hamiltonian simulation e^{iOt}, and then use phase estimation to extract the eigenvalues of O. We find that block encodings and singular value transformation let us build a much simpler algorithm for this task. The idea is to directly construct block encodings of matrices encoding the bits of a binary expansion of the eigenvalues of O, and then the extract the estimate one bit at a time. We find that this method is about 20x faster than the one based on phase estimation.

Finally, we turn to classical Monte Carlo strategies for simulating quantum circuits. A famous early result of quantum information was that classical computers can simulate Clifford circuits [Gottesman98]. We combine this result with quantum Monte Carlo to obtain a classical algorithm for simulating all quantum circuits. However, the further away the circuit is from the Clifford group, the worse the impact of the sign problem becomes. These algorithms have several interesting properties. They can simulate Clifford circuits in linear time, which is quadratically faster than the best known deterministic algorithm [AG04]. They can also simulate a large family of input states that cannot be made using Clifford circuits.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

The world has many possible states. The number of states an object can be in tends to grow exponentially in its size. Describing any particular state might not be so challenging: if a physical system has N many possible states, then we only need about $\log_2 N$ many bits to write it down - the logarithm cancels the exponent, and we are back in business. But what if we are uncertain about what state the system is in? To even describe our own uncertainty, we need to write down a probability for each of the possible states. This is clearly intractable in general.

A key feature of quantum mechanics is that uncertainty is inextricable from the physical state itself. There is no way to deterministically predict the outcomes of all experiments. This fact frustrated many scientists in the early days of quantum mechanics: for example, Albert Einstein famously resisted the idea [Isaacson07], and considered its implications paradoxical [EPR35]. But another consequence is that it makes the study of quantum systems much harder computationally. The behavior of stochastic processes is very difficult to study with deterministic means. For this reason, computational approaches often adopt randomness themselves.

However, quantum mechanical systems also resist a stochastic description. This might have been very pleasing to some scientists: from a Bayesian perspective we can just think of randomness as an uncertainty in the mind of a human being, while Nature herself refuses to 'play dice'. However, Bell inequality violations [Bell64] show us that such a perspective does not work when we consider only probability distributions of positive real numbers. Quantum mechanics implies some form of 'negativity' in the distribution. This could manifest as complex amplitudes in state vectors or complex off-diagonal entries in density matrices, or as negative quasiprobabilities in some hidden variable theory.

The fact that a stochastic hidden variable theory cannot predict the outcomes of quantum mechanics has obvious philosophical consequences: it seems to suggest that randomness is inherent to the natural world, and sometimes has nothing to do with the uncertainty or ignorance of a human being. But it has computational consequences too: as far as we know, it prohibits randomized classical algorithms from efficiently predicting the outcomes of quantum systems. Richard Feynman was one of the first to observe this. In his famous 1981 lecture he insists that Nature has too many variables to allow for an efficient description on a normal computer. Of course, probabilistic processes also have too many variables to allow an efficient deterministic description. The key observation which Feynman also alludes to is that because of the limitations of hidden variable theories, quantum mechanics does not even permit an efficient description in terms of probability distributions.

Consequently, Feynman proposes a computer where superpositions over program states are as natural as tossing a coin as a part of a classical algorithm. The promise of such 'quantum computers' is that, since they obey the same rules as Nature herself, they can help us understand physical phenomena in ways previously not possible. Since quantum technologies have already had an astounding impact on our daily lives, from computer chips to solar cells, quantum computers also promise vast practical applications. This subject was once a curiosity at the fringes of physics and computer science, only considered a radical and impractical academic pastime. But since then, thanks to breakthroughs in the 1990s, quantum computation has blossomed into an enormously rich and exciting field.

As such, our understanding of the capabilities of quantum computers has advanced significantly since Feynman's time. We can be more concrete about exactly how useful we expect quantum computers to be for solving physics problems. We can crudely divide calculations in physics into two main categories: statics and dynamics. A statics question is usually of the form: In what state is a physical system likely to be found? For example, what portion of a floating body will be submerged underwater? What is the color of a black body at a certain temperature? On the other hand, a dynamics question assumes you already know the current state and asks: What will the state be like after some amount of time? Given that the weather was sunny today, will it be sunny tomorrow? When two protons at high velocity collide, what particles do they produce? A significant finding of the past few decades it that quantum computers are very good at answering 'dynamics questions' about quantum physics. This stems from very efficient quantum algorithms for simulating the dynamics of Hamiltonians, either via the Trotter approximation [KKR04] or via qubitization [LC1606, LC1610].

But for 'statics questions', the exponential growth of possibility spaces seems to prevent quantum computers from being a silver bullet [KKR04]. Just because a quantum computer can easily write a quantum state down does not mean that it can easily find a quantum state that matches a certain set of constraints. For example, in ground state finding we are interested in the quantum state that minimizes the energy. For thermal state preparation we want to find the state that maximizes entropy given a certain average energy. Of course, the challenge of searching large possibility spaces does not stem from quantum mechanics! Any classical optimization problem, like finding the shortest Hamiltonian cycle on a weighted graph, can be phrased an energy estimation problem by simply encoding the cost function into the energy. Such general classical optimization problems are widely believed to not be tractable even by a quantum computer. So why should searching over a quantum possibility space be any easier?

Optimization problems remain ubiquitous in science and engineering despite their hardness in the general case. Randomness becomes an extremely valuable asset in tackling these problems. Instead of painstakingly iterating and eliminating possibilities, we can simply randomly guess until we find a good answer. This is called a 'Monte Carlo' algorithm. A big part of their practicality stems from the fact that it is usually not necessary to obtain an answer that is exactly correct with certainty. An answer that is probably roughly right is usually good enough. Many randomized algorithms end up being absurdly simple, because the majority of the challenge lies in making a rigorous probabilistic argument that the method works. In practice, it is often sufficient simply to experimentally demonstrate that the Monte Carlo method is effective.

A surprising discovery that was not anticipated by the field's earliest visionaries is that quantum computers can improve the performance of Monte Carlo algorithms. Just as how classical randomized algorithms can sometimes give a polynomial advantage over deterministic computation, quantum algorithms can improve certain randomized algorithms. The most famous example of this is Grover's search algorithm [Grover96]: say there are N possibilities, and each one is easily verified to be 'correct' or 'incorrect'. The goal of 'search' is to find any correct possibility. Both deterministic and randomized algorithms require at least $\sim N$ verifications to solve the problem. But Grover showed that a quantum computer merely needs $\sim \sqrt{N}$. The fact that the search problem solved by Grover's algorithm is so abstract is precisely what lends it its power. Many problems in computer science and in physics can be put into a form where they can benefit from this quantum speedup. This includes minimizing functions [Grover97], the traveling salesman problem and techniques involving dynamic programming [Ambainis&18], as well as ground state finding [LT20] and thermal state preparation [CS16]. Quantum computers do not make the difficulty of unstructured optimization problems magically disappear. For some important problems we cannot expect exponential speedups, even if we leverage a quantum computer's ability to efficiently write down quantum states. On the other hand, the value of a significant polynomial speedup is not to be understated: a runtime of $\sim N^4$ is impossibly impractical compared to $\sim N$. If one understands a problem well, carefully crafted quantum algorithms can make previously unattainable solutions reachable.

In this dissertation we explore the relationship of quantum computers with Monte Carlo estimation. We present nine algorithms, two classical and seven quantum. These solve both abstract problems like estimation and circuit simulation, as well as more concrete problems like energy measurement and determining the density of states of a physical system. In sections 1.1, 1.2 and 1.3 of this introduction we informally discuss the capabilities of classical and quantum computers, and allude to how this connects to the contributions presented in this dissertation. In section 1.4 we directly summarize the remaining chapters.

1.1 Classical Estimation and Quasiprobability

Many quantities in physics and other fields are prohibitively difficult to compute exactly. However, we are usually satisfied with an estimate that is probably close to the answer. Such estimates can be obtained through Monte Carlo estimation, which is often as simple as sampling a bunch of random numbers from some probability distribution and then taking the average.

One example of Monte Carlo estimation that is common in physics is ‘quantum Monte Carlo’. This is actually a large family of classical methods for studying quantum systems. We give a simple example here to illustrate the basic idea. Say ρ is a quantum state and O is some observable, and we want to compute the expected value $\langle O \rangle = \mathrm{Tr}(\rho O)$. Say there is also a family of quantum states $\{|\phi_i\rangle\}$ (e.g., Gaussian states or stabilizer states) that makes expectations $\langle\phi_i| O |\phi_i\rangle$ easy to compute, and we know of real q_i such that:

$$\rho = \sum_i q_i |\phi_i\rangle \langle\phi_i| \,. \tag{1.1}$$

We could try to compute the expectation of the observable via

$$\mathrm{Tr}(\rho O) = \sum_i q_i \langle\phi_i| O |\phi_i\rangle \,, \tag{1.2}$$

but there could be very many nonzero q_i, so summing over all of them is computationally expensive. Instead, we could take a Monte Carlo approach. Since ρ is normalized, we see that the q_i sum to 1. If furthermore they all are non-negative then the q_i form a probability distribution. Otherwise, we can define $|\vec{q}|_1$ to be the sum of the magnitudes of the q_i. Then we define a random variable:

$$X = |\vec{q}|_1 \cdot \frac{q_i}{|q_i|} \cdot \langle\phi_i| O |\phi_i\rangle \text{ with probability } \frac{|q_i|}{|\vec{q}|_1} \tag{1.3}$$

If we can sample from this random variable, then we can take the average of many samples to estimate its expectation. We see that the expectation is the desired quantity:

$$\mathbb{E}[X] = \sum_i |\vec{q}|_1 \cdot \frac{q_i}{|q_i|} \cdot \langle\phi_i| O |\phi_i\rangle \cdot \frac{|q_i|}{|\vec{q}|_1} = \sum_i q_i \langle\phi_i| O |\phi_i\rangle = \mathrm{Tr}(\rho O) \tag{1.4}$$

How many samples do we need in order to achieve an estimate with decent accuracy? Fortunately this statistics problem is well studied, and the answer to this question is given by the Chernoff-Hoeffding theorem. Say $\hat{X}$ is the average of N samples, ε is the desired accuracy, and $1-\delta$ is the desired success probability. Then:

$$N \geq \frac{1}{2\varepsilon^2} \cdot \ln \frac{2}{\delta} \cdot \mathrm{Var}(X) \qquad \to \qquad \Pr[|\hat{X} - \mathbb{E}[X]| \geq \varepsilon] \leq \delta. \tag{1.5}$$

While the variance itself may be hard to obtain, we can observe that $|X| \leq |\vec{q}|_1 \cdot |O|$ (here $|O|$ is the spectral norm of O) and obtain the bound $\mathrm{Var}(X) = \mathbb{E}[(X - \mathbb{E}(X))^2] \leq (2 \cdot |\vec{q}|_1 \cdot |O|)^2$. Then our classical algorithm requires $N \in O(\varepsilon^{-2} \log(\delta^{-1}) \cdot |\vec{q}|_1^2 \cdot |O|^2)$ many samples.

The $|\vec{q}|_1^2$ term in the runtime is a manifestation of the famous 'sign problem'. Recall that the q_i sum to 1, so if the q_i are all non-negative then $|\vec{q}|_1^2 = 1$. So if there are no minus signs on the q_i the simulation is efficient. Notice that in this case the state ρ is in the convex hull of the $\{|\phi_i\rangle\langle\phi_i|\}$. However, for any family of states $\{|\phi_i\rangle\langle\phi_i|\}$ (that is not just 'all states') there will exist ρ outside their convex hull, meaning that some of the q_i must be negative and we have $|\vec{q}|_1 > 1$. While the magnitudes of the $|q_i|$ are bounded by a constant because ρ is positive semi-definite, the number of q_i scales quadratically with the Hilbert space dimension. So for many quantum states $|\vec{q}|_1$ will be prohibitively large, making the simulation slow. Nonetheless, there exist families of quantum circuits these classical simulation techniques can remain effective. We explore such a situation in Chapter 6.

1.2 Capabilities of Quantum Computers

This section is intended for readers who are already familiar with quantum mechanics, but might not be so familiar with how quantum mechanics is used in quantum computation. Before we go into technical detail, we emphasize some differences between the dynamics of quantum computers and the dynamics of similar systems in physics.

Quantum computers consist of many two-level systems called qubits that are coupled together by a highly programmable time-dependent Hamiltonian. Initially, we can think of a quantum computer like a quantum many body system, or a system of coupled spins. In fact, there are many different physical systems, such as some cold atoms in an electric field or superconducting Josephson-Junctions, that can act like a quantum computer. However, the dynamics of a quantum computer are radically different from how such systems would usually behave. First, we usually expect to find the quantum computer in a pure, non-thermal quantum state. This is because quantum computers can be 'initialized' to a state where all all the qubits are in a tensor product state $|0\rangle^{\otimes n}$. Second, we think of time evolution on quantum computers as discrete, and individual degrees of freedom are sometimes extremely strongly coupled and sometimes completely decoupled. This is because the underlying Hamiltonian is so programmable that we have full control over the strength and nature of the coupling between the qubits. Finally, as time passes, the quantum state remains pure and does not thermalize or decohere. This is because quantum computers are designed to almost completely insulate the state of the qubits from the environment. Decoherence only occurs at the very end of the computation when the qubits are measured and the qubits are reset.

Making a physical system behave in this highly unnatural way is an extremely difficult engineering challenge. However, there has been remarkable theoretical and experimental progress towards this goal in the past few decades. On the theoretical side, quantum error correction gives sound theoretical evidence that insulating a quantum computer from decoherence is possible [ABO99]. On the experimental side, systems of coupled superconducting Josephson-Junctions [Google&19, Wu&21] and squeezed photons [Zhong&20, Zhu&21] have demonstrated significant programmable control over Hilbert spaces so large that they might be impractical to simulate even with the world's largest supercomputers.

There are many models of quantum computation, but the most popular is the 'quantum circuit'. In this dissertation we think of a quantum circuit as a symbolic expression with the following structure:

$$\begin{aligned} C :=\ & U && \text{(a } 2^k \times 2^k \text{ complex unitary matrix)} && (1.6)\\ \text{or}\ & |0\rangle && \left(\text{the column vector } \begin{bmatrix}1\\0\end{bmatrix}\right) && (1.7)\\ \text{or}\ & C \cdot C && \text{(matrix multiplication)} && (1.8)\\ \text{or}\ & C \otimes C && \text{(tensor product)} && (1.9)\\ \text{or}\ & C^\dagger && \text{(adjoint / conjugate transpose).} && (1.10) \end{aligned}$$

The components U and $|0\rangle$ are the primitive building blocks for a circuit. These can then be chained together into a more complex expression using multiplication, tensor product and adjoint. The 'size' of a circuit C is the number of classical bits required to write the expression down. A quantum computer can 'evaluate' a quantum circuit of size s in time $\sim \text{poly}(s)$. To illustrate what we mean by 'evaluate' we provide some simple examples.

First, we define some simple unitary matrices. A matrix is unitary if it is square and satisfies $UU^\dagger = I$.

$$X := \begin{bmatrix}0 & 1\\1 & 0\end{bmatrix} \qquad Z := \begin{bmatrix}1 & 0\\0 & -1\end{bmatrix} \qquad H := \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\1 & -1\end{bmatrix} \qquad (1.11)$$

Here X, Z are the familiar Pauli matrices. The X matrix is very useful in quantum computation because it can be used to flip a quantum bit from the $|0\rangle$ state encoding a '0' to the $|1\rangle$ state encoding a '1'. When we measure the final state of the quantum computer evaluating the circuit $X \cdot |0\rangle$ the output will always be '1' . Here, 'w.p.' stands for 'with probability':

$$X \cdot |0\rangle \qquad \text{evaluates to} \qquad \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} =: |1\rangle \,. \qquad \text{Output: 1 w.p. 100\%} \tag{1.12}$$

H is called the 'Hadamard gate', and it is useful because it lets us create superposition. The circuit $H \cdot |0\rangle$ evaluates to a quantum state with amplitude $1/\sqrt{2}$ for the '0' possibility, and $1/\sqrt{2}$ for the '1' possibility. When the final state of the quantum computer is in superposition, then the quantum computer outputs a random answer according to the Born rule [Born26]: if the amplitude of a possibility is α then the probability of observing it is $|\alpha|^2$.

$$H \cdot |0\rangle \qquad \text{evaluates to} \qquad \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix} . \qquad \text{Output:} \begin{cases} 0 \text{ w.p. } 50\% = |1/\sqrt{2}|^2 \\ 1 \text{ w.p. } 50\% = |1/\sqrt{2}|^2 \end{cases} \tag{1.13}$$

Quantum circuits can easily model the behavior of physical systems. Just as bits can be used to represent any kind of information, the state of a quantum system can be represented using qubits. This is particularly simple for spin systems, so as an example we show how to simulate the dynamics of a transverse-field Ising model. Say we have a collection of n spins arranged in a line, and we encode $|{\downarrow}\rangle = |0\rangle$ and $|{\uparrow}\rangle = |1\rangle$. The Hamiltonian is then easily written in terms of the Pauli matrices:

$$P_k := I^{\otimes(k-1)} \otimes P \otimes I^{\otimes(n-k-1)} \tag{1.14}$$

$$H := -J \sum_{k=1}^{n-1} Z_k Z_{k+1} - h \sum_{k=1}^{n} X_k \tag{1.15}$$

where J is the coupling strength that encourages spins to point the same or opposite direction depending on its sign, and h is the strength of the magnetic field. To simulate time evolution with respect to this Hamiltonian for a time t we use the Suzuki-Trotter approximation [Lloyd96, Childs&19]. This splits split the n-qubit unitary $e^{-iHt/\hbar}$ into a product of many 1- or 2-qubit unitaries:

$$e^{-iHt/\hbar} \approx \prod_{j=1}^{m} \left(\prod_{k=1}^{n-1} \exp\left(\frac{iJt}{m\hbar} Z_k Z_{k+1} \right) \cdot \prod_{k=1}^{n} \exp\left(\frac{iht}{m\hbar} X_k \right) \right) \tag{1.16}$$

where m is a positive integer sufficiently large to make the approximation hold. The idea is essentially to split the time evolution over time t into lots of tiny steps of length t/m. Then the coupling term and magnetic field term in the Hamiltonian approximately commute since they are then multiplied by a small prefactor. If two matrices A, B commute then $e^{A+B} = e^A \cdot e^B$.

Say we want to sample from the distribution over spin configurations obtained from starting in the $|{\downarrow}\rangle^{\otimes n}$ configuration and then evolving for time t under H. Then we have a quantum computer evaluate the following circuit:

$$U_k^{\text{coupl}} := I^{\otimes(k-1)} \otimes \exp\left(\frac{iJt}{m\hbar} \cdot Z \otimes Z \right) \otimes I^{\otimes(n-k-2)} \tag{1.17}$$

$$U_k^{\text{mag}} := I^{\otimes(k-1)} \otimes \exp\left(\frac{iJt}{m\hbar} X \right) \otimes I^{\otimes(n-k-1)} \tag{1.18}$$

$$\prod_{j=1}^{m} \left(\prod_{k=1}^{n-1} U_k^{\text{coupl}} \cdot \prod_{k=1}^{n} U_k^{\text{mag}} \right) |0\rangle^{\otimes n} \text{ approximately evaluates to } e^{-iHt/\hbar} |{\downarrow}\rangle^{\otimes n} \tag{1.19}$$

We observed that for a spin system it was relatively easy to encode the time evolution into a quantum circuit acting on qubits. We can then use the quantum computer to probe the dynamics of the physical system. Quantum computers are quite good at solving dynamics problems in general. Tools like the Jordan-Wigner transformation [SOKG03] can also encode

Hamiltonians with annihilation and creation operators into qubits. This can be used to study Fermi-Hubbard models, molecular Hamiltonians, and even quantum field theory [JLP11].

Say we are interested in obtaining the ground state of a Hamiltonian H, call it $|\psi_0\rangle$. This is known to be intractable for quantum computers in the general case [KKR04]. However, we could be in a special situation such as the one considered by [LT21] where there is a quantum state $|\phi\rangle$ with good overlap with $|\psi_0\rangle$, and a projector Π that projects onto the ground state. We are given quantum circuits that implement the unitary matrices:

$$C_\Pi \text{ implements } (I - 2\Pi) \tag{1.20}$$

$$C_\phi \text{ implements } (I - 2\,|\phi\rangle\langle\phi|) \tag{1.21}$$

$$C_\phi^{\text{prep}} \text{ satisfies } C_\phi^{\text{prep}}\,|0\rangle^{\otimes n} = |\phi\rangle \tag{1.22}$$

Then, we can follow a generalization of Grover's algorithm called 'amplitude amplification'. Assume $\langle\psi_0|\phi\rangle$ is a positive real number, and define θ by $\sin(\theta) := \langle\psi_0|\phi\rangle$. Then there exists a state $|\psi_0^\perp\rangle$ such that:

$$|\phi\rangle = \sin(\theta)\,|\phi_0\rangle + \cos(\theta)\,|\psi_0^\perp\rangle \tag{1.23}$$

With some algebra we can show that the quantum circuit $(C_\phi C_\Pi)^k\, C_\phi^{\text{prep}}\,|0\rangle^{\otimes n}$ satisfies:

$$(C_\phi C_\Pi)^k\, C_\phi^{\text{prep}}\,|0\rangle^{\otimes n} \text{ evaluates to } \sin((2k+1)\theta)\,|\psi_0\rangle + \cos((2k+1)\theta)\,|\psi_0^\perp\rangle \tag{1.24}$$

So by selecting n such that $(2k+1)\theta \approx \pi/2$, we have a quantum circuit that approximately prepares the ground state $|\phi_0\rangle$. We have just sketched a very general method for preparing quantum states of interest.

In the above we have outlined some of the basic capabilities of a quantum computer. The most important observation is that quantum mechanical systems can be translated into systems many qubits. Then, quantum circuits can be used to probe various aspects of the system of interest. We can rewrite time evolution under a Hamiltonian as a sequence of local unitary quantum operations, allowing quantum circuits to answer questions about the dynamics of quantum systems. Also, if some assumptions are met, then we can prepare the ground states of interest using a generalized version of Grover's algorithm.

These tools put us in a good position to study the nature of quantum systems using quantum computers. In the next section we apply these tools to the problem of observable estimation we considered in the previous section.

1.3 Quantum Algorithms for Observable Estimation

While many ideas from quantum physics map very cleanly onto quantum circuits, observables require a little more work. Many of the contributions presented in this dissertation are specifically about providing better tools for studying observables on a quantum computer.

In quantum physics, an observable is denoted by a Hermitian matrix O. This matrix has an eigendecomposition into eigenvectors λ_i and eigenstates $|\lambda_i\rangle$:

$$O = \sum_i \lambda_i |\lambda_i\rangle \langle\lambda_i| \tag{1.25}$$

Say we are interested in evaluating $\langle\psi| O |\psi\rangle$. Since O is Hermitian, its eigenstates $|\lambda_i\rangle$ form a basis. We can expand $|\psi\rangle$:

$$|\psi\rangle = \sum_i |\lambda_i\rangle \cdot \langle\lambda_i|\psi\rangle \tag{1.26}$$

We see that the expectation of the observable is:

$$\langle\psi| O |\psi\rangle = \sum_i \lambda_i \cdot \langle\psi|\lambda_i\rangle \langle\lambda_i|\psi\rangle = \sum_i \lambda_i \cdot |\langle\lambda_i|\psi\rangle|^2 \tag{1.27}$$

Following the Born rule [Born26], quantum mechanics lets us sample from the random variable:

$$Y = \lambda_i \text{ with probability } |\langle\lambda_i|\psi\rangle|^2 \tag{1.28}$$

so we see immediately that $\mathbb{E}[Y] = \langle\psi| O |\psi\rangle$. We also see that $|Y| \leq |O|$, so following the same analysis as above we require $O(\varepsilon^{-2}\log(\delta^{-1}) \cdot |O|^2)$ samples. The $|\vec{q}|_1^2$ term from the runtime of the classical method $O(\varepsilon^{-2}\log(\delta^{-1}) \cdot |\vec{q}|_1^2 \cdot |O|^2)$ is no longer present. This means we have sidestepped the sign problem entirely.

However, as we saw in the previous section, measurement on a quantum computer works quite differently from how it works in quantum physics. In the model we described above, quantum circuits are limited to measuring the state of the quantum system in a fixed basis called the 'computational basis'. The problem of sampling from the random variable Y is related to producing a quantum circuit that can give access to the $|\lambda_i\rangle$ basis from the computational basis. In particular, we would like to perform the transformation:

$$\sum_i \langle\lambda_i|\psi\rangle \cdot |\lambda_i\rangle |0\rangle^{\otimes n} \quad \rightarrow \quad \sum_i \langle\lambda_i|\psi\rangle \cdot |\lambda_i\rangle |\hat{\lambda}_i\rangle \tag{1.29}$$

where $|\hat{\lambda}_i\rangle$ is an n-bit computational basis state that encodes an estimate of the eigenvalue $\hat{\lambda}_i$. For example, if $\lambda_i = 0.101$ then $|\hat{\lambda}_i\rangle$ could be $|1\rangle \otimes |0\rangle \otimes |1\rangle$. Then, to sample from the random variable Y we simply look at the output state in the computational basis and read off a randomly sampled eigenvalue.

Performing the above transformation is easy when O is local - that is, supported on a small subset of the qubits. However, there is a particularly important example of an observable that is usually not local in this sense: the Hamiltonian. The problem of energy estimation is famously solved via a complicated strategy called ‘phase estimation’ [NC00], which involves many sophisticated tools such as Hamiltonian simulation and the quantum Fourier transform. In Chapter 5 we outline a new algorithm for energy estimation (that can also be applied to arbitrary observables) that significantly outperforms the traditional method in terms of circuit size.

Above we showed how quantum computers can avoid the sign problem by avoiding quasiprobabilities and directly sampling in the eigenbasis of an observable. But the runtime of estimating expectations of observables can be improved even further. In their seminal work, [BHMT00] proposed a quantum algorithm called ‘amplitude estimation’ that performs the following task. The premise is extremely similar to amplitude amplification: say $|\phi\rangle$ is a quantum state and Π is a projector. Then let $a := |\Pi|\phi\rangle|$ be the ‘amplitude’. Given sub-circuits C_Π, C_ϕ, and C_ϕ^{prep} as above, their algorithm outputs an estimate $\hat{a}$ such that:

$$\Pr[|\hat{a} - a| \geq \varepsilon] \leq \delta \tag{1.30}$$

The algorithm makes use of merely $O(\varepsilon^{-1}\log(\delta^{-1}))$ many instances of the sub-circuits. Notice that the ε^{-2} we obtained from the Chernoff-Hoeffding theorem has been improved to ε^{-1}. This improvement in the accuracy of estimation is the source of many polynomial quantum speedups. While their algorithm is quite complicated, one of our findings is that a significantly simpler algorithm for amplitude estimation exists. We discuss these techniques in Chapter 2.

At first glance, amplitude estimation may seem like a very restricted task since Π must be a projector. However, it turns out that pretty much any Monte Carlo estimation task can be mapped to and accelerated by amplitude estimation. There is a large body of literature that showcases this fact: there are quantum algorithms for estimating classical partition functions [Mon15], performing Bayesian inference and preparation of thermal states [HW19, AHNTW20], estimation of the volumes of convex bodies [Chak&19], and solving differential equations with applications in finance [An&20]. There are also more general tools for quantum mean estimation [HM18], quantum median estimation [BDGT11], and quantum gradient estimation [Jordan04, GAW17]. We find that there is a quadratic quantum speedup for a very generic notion of estimation, just like how Grover's search algorithm works for a very broad idea of search. In fact, many of the works listed make use of a combination of quantum search and estimation in order to combine multiple quadratic speedups into a more significant polynomial speedup.

Here we show how estimation of expectations of observables $\langle\psi| O |\psi\rangle$ can be brought into the amplitude estimation framework using block encodings. Block encodings are a fairly recent tool for designing quantum algorithms that enable manipulation of non-unitary matrices. If O is a observable, then a block encoding of O is a unitary matrix U_O that puts O into the top left submatrix:

$$U_O := \begin{bmatrix} O & \cdot \\ \cdot & \cdot \end{bmatrix} \tag{1.31}$$

Block encodings can be synthesized in several ways. For example, it was shown in [BCK15] that if O is a linear combination of unitaries, then there exists a block encoding of O. Since Pauli matrices are simultaneously unitary and a basis for all hermitian matrices, we can use linear combinations of Pauli matrices to synthesize any observable. In Chapter 3 we give an accessible review on how to construct quantum circuits with block encodings.

We can rephrase the defining property of U_O as a property of a quantum circuit:

$$(\langle 0| \otimes I)U_O(|0\rangle \otimes I) = O \tag{1.32}$$

Now we can construct a state $|\phi\rangle$ and a projector Π so that $a = |\Pi\,|\phi\rangle\,| = |\,\langle\psi|\,O\,|\psi\rangle\,|$.

$$|\phi\rangle = U_O(|0\rangle \otimes |\psi\rangle) \qquad \Pi = |0\rangle\,\langle 0| \otimes |\psi\rangle\,\langle\psi| \tag{1.33}$$

$$|\Pi\,|\phi\rangle\,| = |(\langle 0| \otimes \langle\psi|)U_O(|0\rangle \otimes |\psi\rangle)| = |\,\langle\psi|\,O\,|\psi\rangle\,| \tag{1.34}$$

We can the use amplitude estimation to obtain an estimate $\hat{a}$ of $|\,\langle\psi|\,O\,|\psi\rangle\,|$ with accuracy ε. The runtime is $O(\varepsilon^{-1}\log(\delta^{-1}) \cdot |O|)$. This quadratically improves over the complexity of the classical algorithm in both ε and $|O|$.

In Chapter 4 we show how to generalize the above technique to estimate $\langle\psi|\,O\,|\psi\rangle$ rather than its magnitude, and show how to estimate the expectation $\mathrm{Tr}(\rho O)$ of mixed states ρ. The ability to estimate observables on a quantum computer enables us to compute many interesting quantities in physics. This includes n-time correlation functions, the density of states, and a broad family of linear response functions.

1.4 Summary of Chapters

The above discussion outlined many of the general ideas that appear in this dissertation, and alluded to some of the contributions. To make these contributions more explicit, we give a short summary of each chapter of the dissertation here.

Chapter 2: Simplified Amplitude Estimation

As discussed above, amplitude estimation is a powerful quantum subroutine that quadratically improves the accuracy-runtime tradeoff for Monte Carlo estimation. Above, we applied it to the quantum mechanical problem of observable estimation, but it can also be used for entirely classical problems. The general 'black-box' estimation problem considered in the classical algorithms and complexity literature is 'approximate counting': we are given a set of N items, and an unknown number $K > 0$ of these are 'marked'. We are to output an estimate $\hat{K}$ of K that satisfies:

$$\Pr[(1-\varepsilon)K \leq \hat{K} \leq (1+\varepsilon)K] \geq 1-\delta \tag{1.35}$$

for some failure probability δ. This algorithm is given an 'oracle' that checks if a particular item is marked, as should query this oracle as few times as possible. The power of approximate counting comes from its abstract nature: the items could be graph colorings that are valid whenever the coloring is valid, or configurations of a classical spin system that are marked whenever the total energy is below some threshold.

In 2000, Brassard, Høyer, Mosca, and Tapp [BHMT00] presented a quantum algorithm that solves approximate counting as presented above in $O\left(\sqrt{N/K} \cdot \varepsilon^{-1} \log\left(\delta^{-1}\right)\right)$ queries. This is a quadratic improvement over the best classical algorithm which takes $\Theta\left((N/K) \cdot \varepsilon^{-2} \log(\delta^{-1})\right)$ queries.

However, this algorithm is very complicated. First, it constructs a state $|\phi\rangle$ and a projector Π such that $|\Pi|\phi\rangle| = \sqrt{N/K} := \sin(\theta)$. Then, following our discussion of amplitude amplification above, it constructs a quantum circuit:

$$C_\phi C_\Pi = (I - 2\Pi)(I - 2|\phi\rangle\langle\phi|) \tag{1.36}$$

and observes that the complex eigenvalues of this matrix are actually $e^{\pm 2i\theta}$. Then, it uses an application of the quantum phase estimation algorithm [NC00] to extract $\pm 2\theta$ to sufficient precision, and translates the result into an estimate $\hat{K}$.

In 2019, [AR19] observed that it is possible to perform approximate counting without invoking the quantum phase estimation algorithm. In fact, all the quantum parts of this algorithm can be packed into a single black-box, after which the algorithm becomes entirely classical. This black-box is called a 'Grover coin' - a coin that comes up heads with probability $\sin^2(n\theta)$ for any odd integer n. We construct such a coin via a circuit we have already discussed: we select k such that $2k+1=n$, and we evaluate the quantum circuit $(C_\phi C_\Pi)^k C_\phi^{\text{prep}} |0\rangle$ to sample an item that is marked with probability $\sin^2((2k+1)\theta)$. Each toss of the Grover coin uses $O(n)$ queries to the oracle.

Chapter 2 showcases the algorithm presented in [AR19] that uses repeated tosses of the Grover coin to obtain an estimate of θ. The estimate of θ then gives an estimate of K. The algorithm neatly splits into two parts. First, a pre-processing step obtains bounds $\theta_{\min}$ and $\theta_{\max}$ such that $\theta_{\min} \leq \theta \leq \theta_{\max}$ and $\theta_{\max}/\theta_{\min} \leq 1.65$. This is achieved by tossing Grover coins with exponentially increasing n until enough heads are observed. Second, the algorithm iteratively shrinks the interval $[\theta_{\min}, \theta_{\max}]$ until $\theta_{\max}/\theta_{\min} \leq 1+\varepsilon$. If the interval is this small, then any value inside the interval is a good estimate of θ.

Actually, several simplifications of [BHMT00] appeared in 2019. The first of these was [Suzuki&19], which presented an algorithm based on maximum likelihood estimation. This was followed by [Wie19], who discovered an algorithm similar to Kitaev's iterative phase estimation algorithm [Kit95]. However, both of these algorithms claimed accuracy solely based on numerical experiments and do not keep track of the failure probability throughout the algorithm. In that regard, the primary advancement over the state of the art contributed by [AR19] is that it features a rigorous proof that the presented algorithm is correct. Furthermore, it presents an algorithm for relative error estimation rather than additive error estimation. On the other hand, the analysis of the algorithm makes no attempt to reduce the query complexity of [BHMT00] in terms of constant factors. This was achieved by [GGZW19], who present a modified version of the algorithm developed in [AR19]. They also give a comprehensive constant-factor performance analysis, demonstrating that their new algorithm has the best performance of all currently known algorithms for approximate counting and amplitude estimation. At time of writing, [GGZW19] continues to be the state of the art algorithm for the task.

Chapter 3: An Introduction to Block Encodings

In the previous section we mentioned block encodings, a method for manipulating non-unitary matrices on a quantum computer. This capability is enormously useful, since in the above we encountered several matrices that are not unitary: observables O, the Hamiltonian H, as well a projector onto the ground state Π. Being able to reason about non-unitary matrices directly greatly simplifies the construction of quantum algorithms. Chapters 4 and 5 leverage these capabilities, so this chapter provides a short review of these techniques.

This chapter is largely based on the results of [GSLW18], which is extremely comprehensive but also extremely technical and challenging to read. A recent review article [MRTC21] is less technical, and argues that these techniques enable a 'grand unification of quantum algorithms'. They give many examples of their applications without detailing their implementation. Our contribution aims to strike a middle ground: we focus on the techniques themselves rather than their applications, and explain how they work with a balance between rigor and readability. Unlike the other chapters, this chapter is not based on an existing publication.

In an earlier section we discussed the capabilities of quantum computers in terms of quantum circuits C. A quantum circuit C is a symbolic expression composed of unitary matrices U, initial states $|0\rangle$, matrix multiplications $\cdot$, tensor products $\otimes$, and adjoints †. If a circuit requires s bits to write down, then a quantum computer can evaluate the expression in time poly(s). Block encodings let us define a type of circuit called a 'block encoding circuit' B with a slightly more general set of operations.

$$
\begin{aligned}
B :=\ & M && \text{(any } 2^n \times 2^m \text{ matrix)} && (1.37)\\
\text{or}\ & B \cdot B && \text{(matrix multiplication)} && (1.38)\\
\text{or}\ & B + B && \text{(matrix addition)} && (1.39)\\
\text{or}\ & B \otimes B && \text{(tensor product)} && (1.40)\\
\text{or}\ & B^\dagger && \text{(adjoint / conjugate transpose)} && (1.41)\\
\text{or}\ & p(B) && \text{(singular value transformation by a polynomial } p\text{).} && (1.42)
\end{aligned}
$$

The key idea is that for every block encoding circuit B there exists a unitary circuit C that is only polynomially larger and evaluates to the same thing. In this sense, we can think of block encoding circuits as a 'high level programming language' for quantum computers that 'compiles' to unitary quantum circuits.

There are three major differences between block encoding circuits B and unitary quantum circuits C. First, the unitary matrix U and initialization $|0\rangle$ primitives have been replaced with a single primitive: arbitrary $2^n \times 2^m$ matrices M. Second, we have gained the ability to add circuits together. This is a generalization of the 'linear combinations of unitaries' technique explored by [CW12, Berry&14, CKS15] for the purposes of Hamiltonian simulation. Finally, we have gained the ability to transform the singular values of a circuit by a polynomial p. This is the most complicated new primitive, but also by far the most useful. Singular value transformation underpins the most modern quantum methods for Hamiltonian simulation, solving linear systems of equations [CKS15], and even amplitude amplification [MRTC21].

Chapter 4: Estimation of Physical Quantities

Quantum computers are a powerful tool for studying physical systems. However, earlier we discussed how different they are from the systems we want to study: time evolution is discrete and unitary, and the quantum states are mostly pure. Translating problems from physics into the language of quantum computation has become a research area in its own right. Hamiltonian simulation alone is already an incredibly rich sub-field of quantum algorithms. Say a condensed matter physicist, a nuclear physicist, or a chemist want to benefit from the power of a quantum computer. Do they have to spend countless hours familiarizing themselves with the details of quantum algorithms?

In [Rall20], we build quantum algorithms for some physical quantities, such as correlation functions in the Heisenberg picture, the density of states, and linear response functions. These algorithms are all based on the block encoding circuits we discussed in the previous section. However, the central takeaway of the work is this: translating a physical quantity into a language that quantum computers can understand is really not so difficult.

It is clear from previous approaches [Pedernales&14, RC18] that the primary challenge is to translate non-unitary observables into unitary matrices. But block encodings handle all of that.

The technical tool of [Rall20] that bridges the gap between algebraic expressions for physical quantities and quantum circuits is a quantum algorithm for observable estimation. We already saw in the previous section that if O is an observable with a block encoding, and $|\psi\rangle$ is a quantum state, then there exists a quantum algorithm that estimates $|\langle\psi| O |\psi\rangle|$ to precision ε using $O(\varepsilon^{-1} \cdot |O|)$ queries. The observable estimation algorithm generalizes this to estimation of $\text{Tr}(\rho O)$: the quantum state can now be mixed, the expectation can be negative, and furthermore both the state ρ and the observable O only need to be represented approximately.

From there, constructing an algorithm that estimates a complicated observable is relatively easy: previous results for state preparation and Hamiltonian simulation can be plugged together without needing to understand how they work. For example, we prepare a thermal state of a Hamiltonian H using [CS16]. Then we use the algorithm presented in [GSLW18] to make a block encoding of the time evolution operator e^{-iHt}. Finally, we use linear combinations of Pauli matrices [CKS15] to make block encodings of some observables O_1, O_2, O_3. Now we just combine all of these together: we move each the observables into the Heisenberg picture with $O_i(t_i) := e^{iHt_i} O_i e^{-iHt_i}$, multiply them, and use the observable estimation algorithm to compute:

$$\langle O_1(t_1) O_2(t_2) ... \rangle = \text{Tr}(\rho O_1(t_1) O_2(t_2) ...) \tag{1.43}$$

Other interesting physical quantities are functions of the energy, and require a little bit more work. For example, say H is a Hamiltonian with dimension D and eigenvalues E_i. Then, the density of states is given by:

$$\rho(E) := \frac{1}{D} \sum_i \delta(E_i - E) \tag{1.44}$$

Since this is a sum of delta functions, particular values of $\rho(E)$ are either 0 or ∞. Clearly we are not actually interested in evaluating $\rho(E)$, but rather 'sketching' the function over the entire range of energies. One method for sketching $\rho(E)$ is to compute a histogram: we split the range of energies into several intervals $[a, b]$ and compute:

$$\text{histogram-bin}(a, b) = \int_a^b \rho(E) dE \tag{1.45}$$

Another standard technique for sketching $\rho(E)$ that is used outside of the quantum computing literature is the Kernel Polynomial Method (KPE) [WWAF05]. We normalize the Hamiltonian so that $|E| \leq 1$, and then split $\rho(E)$ into a linear combination of Chebyshev polynomials $T_n(E)$:

$$\rho(E) = \frac{1}{\pi\sqrt{1 - E^2}} \sum_{n=0}^{\infty} c_n T_n(E) \tag{1.46}$$

Now, another method of sketching $\rho(E)$ is to truncate the above series at some desired precision, and to compute the first couple 'Chebyshev moments' c_n, given by:

$$c_n := \int_{-1}^{1} \rho(E) T_n(E) dE \tag{1.47}$$

We find that we can evaluate both c_n and histogram-bin(a, b) by performing singular value transformation on H to construct an observable, and then estimating its expectation with the maximally mixed state. In the case of Chebyshev moments, we simply perform singular value transformation with $p(x) = T_n(x)$. If $H = \sum_i E_i |\psi_i\rangle \langle\psi_i|$, we find:

$$\mathrm{Tr}\left(\frac{I}{D}T_n(H)\right) = \frac{1}{D}\sum_i T_n(E_i) = \int_{-1}^{1} \frac{1}{D}\sum_i \delta(E_i - E)T_n(E)dE = \int_{-1}^{1} \rho(E)T_n(E)dE \tag{1.48}$$

So we can compute c_n by evaluating the expectation of the observable $T_n(H)$ on the maximally mixed state I/D. If instead we want to compute histogram-bin(a, b), we find a polynomial $w_{a,b}(x)$ satisfying:

$$w_{a,b}(x) \approx \begin{cases} 1 & \text{if } x \in [a, b] \\ 0 & \text{otherwise} \end{cases} \tag{1.49}$$

Then, following the same calculation as (1.48), histogram-bin(a, b) is approximated by the expectation of $w_{a,b}(H)$ with I/D. Such a polynomial approximation can be constructed using several ways. In [Rall20] we present a method based on Jackson's theorem [Rivlin69] and amplifying polynomials [Diak09]. However, in our later work [Rall21] which we discuss in our next chapter, we find that a construction based on the Jacobi-Anger expansion [LC17] yields better constant factors.

The above method for sketching the density of states $\rho(E)$ also works for other quantities like the local density of states, as well as correlation functions of the form $A(E) = \langle B\delta(H - E)C\rangle$ which are useful for studying electrical conductivity [DiVentra08]. Overall we find that, using only a couple of tricks like a quantum algorithm for observable estimation and the calculation in (1.48), block encoding techniques directly yield algorithms for estimating quantities of physical interest.

Chapter 5: Measuring Observables

In this chapter we revisit the problem of 'measuring an observable' on a quantum computer. Say O is a Hermitian matrix with eigendecomposition:

$$O = \sum_i \lambda_i \left|\lambda_i\right\rangle \left\langle\lambda_i\right| \tag{1.50}$$

Then, if an experimenter measures the observable O given a quantum system in the state $|\psi\rangle$, then with probability $|\langle\lambda_i|\psi\rangle|^2$ the experimenter sees λ_i while the system collapses into the state $|\lambda_i\rangle$.

But, as discussed earlier, this is not how measurement works on a quantum computer. Instead, we have the ability to measure individual qubits, collapsing their state to either $|0\rangle$ or $|1\rangle$. If the qubits are entangled with the rest of the system, their collapse induces a collapse of other quantum data. In order to implement observable measurement in the physics sense, we must implement a quantum algorithm that performs the transformation:

$$\sum_i \alpha_i \left|\lambda_i\right\rangle \left|0^n\right\rangle \qquad \to \qquad \sum_i \alpha_i \left|\lambda_i\right\rangle |\hat\lambda_i\rangle \tag{1.51}$$

where $|\hat\lambda_i\rangle$ is a computational basis state encoding an estimate of λ_i. We can then proceed to measure the $|\hat\lambda_i\rangle$ state.

Measuring an observable, especially the Hamiltonian, is an extremely common subroutine in quantum algorithms. It is used in algorithms for preparing ground states [LT20] and thermal states [Temme&09, YA11, Lemi&19], but also also for linear algebra [HHL08, CKS15], estimating partition functions [Mon15] and performing Bayesian inference [HW19]. The textbook method for performing the above transformation is phase estimation [NC00], an extremely complicated algorithm involving both Hamiltonian simulation and the quantum Fourier transform.

In [Rall21] we present a new quantum algorithm for observable measurement that is conceptually simpler, and offers an about 20x constant factor speedup over the traditional method. The same techniques also yield new algorithms for phase estimation and amplitude estimation.

The algorithm is based on block encoding techniques. Since $|\hat{\lambda}_i\rangle$ is a computational basis state, we can write it as a tensor product of the bits in the binary expansion:

$$|\hat{\lambda}_i\rangle = |\text{bit}_n(\lambda_i)\rangle \otimes |\text{bit}_2(\lambda_i)\rangle \otimes |\text{bit}_1(\lambda_i)\rangle \tag{1.52}$$

where $\text{bit}_i : \mathbb{R} \to \{0, 1\}$ indicates the i'th bit in the binary expansion. The main idea is to construct approximate block encodings of projectors Π_i that encode each of these bit values:

$$\Pi_i = \sum_i \text{bit}_i(\lambda_i) |\lambda_i\rangle \langle\lambda_i| \approx \sum_i p_i(\lambda_i) |\lambda_i\rangle \langle\lambda_i| = p_i(O) \tag{1.53}$$

where $p_i(x)$ is a polynomial that approximates the $\text{bit}_i(x)$ constructed from the Jacobi-Anger expansion [LC17].

Then, the algorithm leverages a novel 'block measurement theorem' that implements the unitary U_{Π_i} given a block encoding of Π_i:

$$U_{\Pi_i} = \Pi_i \otimes X + (I - \Pi_i) \otimes I \tag{1.54}$$

$$U_{\Pi_i}(|0\rangle \otimes |\lambda_i\rangle) = |\text{bit}_i(\lambda_i)\rangle \otimes |\lambda_i\rangle \tag{1.55}$$

While the algorithm is very simple on a high level, a significant amount of technical work is required for the analysis. A central challenge is dealing with the approximation error in the construction of the polynomials $p_i(x) \approx \text{bit}_i(x)$. Since polynomials are continuous everywhere, but $\text{bit}_i(x)$ has discontinuities, we find that achieving $|p_i(x) - \text{bit}_i(x)| \leq \varepsilon$ for all x is actually impossible. Worse yet, *any* quantum algorithm that is completely unitary has this limitation. The analysis requires carefully selecting and minimizing the size of regions where the approximation fails to hold, and showing that the resulting errors do not propagate to later bits.

We also perform a numerical constant factor analysis, where we show that the novel observable estimation algorithm is about 20x faster than the algorithm based on traditional phase estimation. The novel phase estimation algorithm based on the same techniques is about 14x faster. Since phase and observable estimation are extremely common subroutines in quantum algorithms, we expect these algorithms to be very useful for the design of future algorithms. For example, [WT21] uses our observable estimation algorithm to design a novel method for thermal state preparation.

Chapter 6: Estimating Near-Clifford Quantum Circuits

In the final chapter, we return to the design of classical Monte Carlo algorithms. Earlier, we discussed how a quantum state $|\psi\rangle$ could be decomposed into a linear combination of some family of quantum states $\{|\phi_i\rangle\}$:

$$|\phi\rangle\langle\phi| = \sum_i q_i |\phi_i\rangle\langle\phi_i| \tag{1.56}$$

Then, we could represent $|\psi\rangle$ via a probability distribution over the set $\{|\phi_i\rangle\}$:

$$X = |\vec{q}|_1 \cdot \frac{q_i}{|q_i|} \cdot |\phi_i\rangle\langle\phi_i| \text{ with probability } \frac{|q_i|}{|\vec{q}|_1}. \tag{1.57}$$

Then $\mathbb{E}[X] = |\psi\rangle\langle\psi|$. If the $|\phi_i\rangle$ have efficient classical representations, and we can sample from the distribution $|q_i|/|\vec{q}|_1$, then we obtain a classical algorithm for analyzing quantum mechanics.

This includes the evaluation of quantum circuits. [Bennink&17] performs exactly the above by selecting $\{|\phi_i\rangle\}$ to be the stabilizer states, which are a discrete family of n-qubit quantum states with $O(n^2)$-size classical representations [Gottesman98]. A famous early result in quantum computation was the Gottesmann-Knill theorem, which states that quantum circuits composed entirely of a family of unitaries called the 'Clifford gates' are efficiently classically simulable [AG04], since all they can produce are stabilizer states.

With quantum Monte Carlo approach above, [Bennink&17] presents a new algorithm we call 'stabilizer propagation' that extends the capabilities of the Gottesmann-Knill theorem. The new algorithm samples from a random variable whose expectation is the probability of seeing a particular outcome at the end of *any* quantum circuit. Furthermore, if the circuit has size s, then we can sample from the random variable in time poly(s). However, the number of samples required scales exponentially in the number of non-Clifford unitaries and non-stabilizer states.

In this chapter we present the two algorithms from [RLCK19], which have some similarities to stabilizer propagation. Rather than using a collection of efficiently representable quantum states $\{|\phi_i\rangle\}$ as a starting point for a Monte Carlo strategy, the algorithms leverage n-qubit tensor products of Pauli matrices. These Pauli matrices can be propagated through the circuit from start to finish - we call this Schrödinger propagation, or they can be propagated through the circuit in reverse - we call this Heisenberg propagation. We find that these algorithms have some extremely surprising properties.

First, Schrödinger propagation and Heisenberg propagation can simulate Clifford circuits in linear time: if the circuit has n qubits and m gates then the runtime is $O(n+m)$. This is surprising because the previously best known classical algorithms [AG04] require $O(nm + n^3)$ time (or $O(nm + n^2)$ depending on the situation). The catch is that the algorithms in [RLCK19] perform a much weaker notion of simulation: instead of calculating output probabilities exactly, they merely compute an ε-error estimate of the probability.

Second, Schrödinger propagation and Heisenberg propagation can simulate non-stabilizer states. In fact, Heisenberg propagation does not care about the initial state at all, provided it is separable into constant-size pieces. On the other hand, Schrödinger propagation can simulate a family of states called 'hyper-octahedral states' which contains states that are not probabilistic mixtures of stabilizer states. Even more surprisingly, Schrödinger propagation requires exponentially fewer samples if the input states are noisy.

Finally, the runtime of these algorithms is given by a quantity called the 'stabilizer-norm', which was originally constructed as a lower bound for stabilizer propagation [HC16]. Therefore, the family of unitaries and quantum states simulable by Schrödinger propagation and Heisenberg propagation is strictly larger than that of stabilizer propagation. Many of the odd properties of these new algorithms stem from the properties of the stabilizer norm.

The algorithms in [RLCK19] join a large family of Monte Carlo approaches for simulating quantum circuits dominated by Clifford gates. These algorithms are very useful for performing numerical simulations of quantum error correction. Prior to [Bennink&17], [BG16] gave a very fast algorithm for simulating circuits composed of noiseless Clifford gates and a single non-Clifford gate. This algorithm has since been improved to support arbitrary unitary quantum circuits [Bravyi&18], quantum circuits composed of qutrits [HL18], and even later quantum circuits with noise [Seddon&20]. Each of these algorithms yields a new measure of 'non-stabilizerness' in bounding its runtime, and has advantages and disadvantages compared to the others.

Chapter 2

Simplified Approximate Counting and Amplitude Estimation

This chapter is based on [AR19]. Scott Aaronson had the original idea for an algorithm that refines a relative-error estimate for an amplitude. Patrick Rall developed the algorithm that makes the initial rough guess that is required to 'jump-start' Dr. Aaronson's procedure, and wrote the error analysis of the protocols. Dr. Aaronson contributed the majority of the prose.

Approximate counting is one of the most fundamental problems in computer science. Given a list of N items, of which $K > 0$ are marked, the problem is to estimate K to within a multiplicative error of ε. One wants to do this with the minimum number of *queries*, where a query simply returns whether a given item $i \in [N]$ is marked.

Two decades ago, Brassard, Høyer, Mosca, and Tapp [BHMT00] gave a celebrated quantum algorithm for approximate counting, which uses $O\left(\varepsilon^{-1}\sqrt{N/K}\right)$ queries. This is tight, matching a lower bound of Nayak and Wu [NW98], and is a quadratic speedup over the best possible classical query complexity of $\Theta\left(\varepsilon^{-2}(N/K)\right)$. This is the same type of quantum speedup as provided by the famous *Grover's algorithm* [Grover96], for *finding* a marked item in a list of size N, and indeed the BHMT algorithm builds on Grover's algorithm.

Curiously, though, the BHMT algorithm was *not* just a simple extension of Grover's algorithm to a slightly more general problem (approximate counting rather than search). Instead, BHMT made essential use of the Quantum Fourier Transform (QFT): a component that appears nowhere in Grover's algorithm, and that's more commonly associated with the exponential speedup of Shor's factoring algorithm [Shor95]. Indeed, BHMT presented their approximate counting algorithm as a sort of hybrid of Grover's and Shor's algorithms.

This raises an obvious question: is the QFT in any sense *necessary* for the quadratic quantum speedup for approximate counting? Or can we obtain that "Grover-like speedup" by purely Grover-like means?

In this paper we settle that question, by giving the first rigorous quantum approximate counting algorithm that's based *entirely* on Grover iterations, with no QFTs or other quantum-mechanical ingredients. Matching [BHMT00], the query complexity of our algorithm is the optimal $O\left(\varepsilon^{-1}\sqrt{N/K}\right)$, while the computational complexity exceeds the query complexity by only an $O(\log N)$ multiplicative factor. Because of its extreme simplicity, our algorithm might be more amenable than [BHMT00] or other alternatives to implementing on near-term quantum computers. The analysis of our algorithm is also simple, employing standard classical techniques akin to estimating the bias of a coin via many coin tosses.[1]

An approach broadly similar to ours was outlined by Grover [Grover97] in 1997, with fuller discussion by Abrams and Williams [AW99] in 1999. The latter authors sketched how to estimate the integral of a function over some domain, to additive error ε, using $O(\varepsilon^{-1})$ quantum queries. Crucially, however, neither Grover nor Abrams and Williams prove the

[1]Indeed, [BHMT00] also requires such classical techniques, since the QFT alone fails to reliably extract the desired information from the Grover operator. By removing the QFT, we show that Grover and estimation techniques alone can do the whole job. This gives a clear way to understand in what sense our algorithm is "simpler."

correctness of their approach—among other issues, they assume that a probability can be estimated to a desired precision without any chance of failure. Also, it is not clear how to adapt their approaches to the broader problem of amplitude estimation.

As we were writing this paper, two other quantum algorithms for approximate counting were announced that avoid the use of QFTs. Surprisingly, both algorithms differ significantly from ours.

In April, Suzuki et al. [Suzuki&19] gave an $O\left(\varepsilon^{-1}\sqrt{N/K}\right)$-query quantum algorithm that first collects samples from various Grover iterations, and then extracts an approximate value of K via maximum likelihood estimation. Finding the maximum of the likelihood function, according to Suzuki et al., incurs a $\log \frac{1}{\varepsilon}$ computational overhead. More importantly, even if we focus only on query complexity, Suzuki et al. do not prove their algorithm correct. Their analysis gives only a *lower* bound on the error, rather than an upper bound, so is supplemented by numerical experiments. By contrast, our analysis is fully rigorous. On the other hand, the Suzuki et al. algorithm has the interesting feature that its invocations of Grover's algorithm are nonadaptive (i.e., can be performed simultaneously), whereas our algorithm requires adaptivity.

In July, Wie [Wie19] sketched another $O\left(\varepsilon^{-1}\sqrt{N/K}\right)$-query, QFT-free quantum approximate counting algorithm. Wie's algorithm is based on Hadamard tests, which require the more expensive "controlled-Grover" operation rather than just bare Grover iterations. Replacing the QFT with Hadamard tests is called "iterative phase estimation," and was suggested by Kitaev [Kit95]. Wie modifies iterative phase estimation in order to apply it to the BHMT algorithm. Unfortunately, and like the previously mentioned authors, Wie gives no proof of correctness. Indeed, given a subroutine that accepts with probability

p, Wie (much like Abrams and Williams [AW99]) simply assumes that p can be extracted to the requisite precision. There is no analysis of the overhead incurred in dealing with inevitable errors. Again, in place of analysis there are numerical experiments.

One reason why approximate counting is of interest in quantum computation is that it generalizes to a technique called *amplitude estimation*. Amplitude estimation is a pervasive subroutine in quantum algorithms, yielding for example faster quantum algorithms for mean estimation, estimation of the trace of high-dimensional matrices, and estimation of the partition function in physics problems [Mon15]. In general, amplitude estimation can upgrade almost any classical Monte-Carlo-type estimation algorithm to a quantum algorithm with a quadratic improvement in the accuracy-runtime tradeoff. Once we have our quantum approximate counting algorithm, it will be nearly trivial to do amplitude estimation as well.

In Section 2.2 we present our main result—the QFT-free approximate counting algorithm and its analysis—and then in Section 2.3 we generalize it to amplitude estimation. We conclude in Section 2.4 with some open problems.

2.1 Main Ideas

Our algorithm for approximate counting mirrors a standard classical approach for the following problem: given a biased coin that is heads with probability p, estimate p. First, for $k = 0, 1, 2, \ldots$ the coin is tossed 2^k times, and stop at the k where heads was observed least once. This gives a rough guess for p, up to some multiplicative constant. Second, this rough guess is improved to the desired $1+\varepsilon$ approximation via more coin tosses.

Of course, we'd like to use Grover's algorithm [Grover96] to speed up this classical approach quadratically. Grover's algorithm can be seen as a special 'quantum coin,' which works as follows. If K out of N items are marked, then define $\theta := \arcsin \sqrt{K/N}$ to be the 'Grover angle.' For any odd integer r, Grover's algorithm lets us prepare a coin that lands heads with probability $p = \sin^2(r\theta)$, by making $O(r)$ queries to the oracle.

The key idea of our algorithm is to use this 'Grover coin' repeatedly, in a manner akin to binary search—adaptively varying the value of r in order to zero in on the correct value of θ and hence K. In more detail, suppose that θ has already been narrowed down to the range $[\theta_{\min}, \theta_{\max}]$. Then in a given iteration of the algorithm, the goal is to shrink this range by a constant factor, either by increasing $\theta_{\min}$ or by decreasing $\theta_{\max}$. To do so, we need to rule out one of the two possibilities $\theta \approx \theta_{\min}$ or $\theta \approx \theta_{\max}$. This, in turn, is done by finding some value of r that distinguishes the two possibilities, by making $\theta \approx \theta_{\min}$ and $\theta \approx \theta_{\max}$ lead to two nearly-orthogonal quantum states that are easy to distinguish by a measurement.

But why should such a value of r even exist—and if it does, why should it be small enough to yield the desired query complexity? Here we need a technical claim, which we call the "Rotation Lemma" (Lemma 2.2). Consider two runners, who race around and around a circular track at differing constant speeds (corresponding to $\theta_{\min}$ and $\theta_{\max}$). Then informally, the Rotation Lemma upper-bounds how long we need to wait until we find one runner reasonably close to the start or the midpoint of the track, while the other runner is reasonably close to the one-quarters or three-quarters points. Here we assume that the ratio of the runners' speeds is reasonably close to 1. We ensure this property with an initial preprocessing step, to find bounds $\theta_{\min} \leq \theta \leq \theta_{\max}$ such that $\theta_{\max}/\theta_{\min} \leq 1.65$.

Armed with the Rotation Lemma, we can zero in exponentially on the correct value of θ, gaining $\Omega(1)$ bits of precision per iteration. The central remaining difficulty is to deal with the fact that our ‘Grover coin’ is, after all, a *coin*—which means that each iteration of our algorithm, no matter how often it flips that coin, will have some nonzero probability of guessing wrong and causing a fatal error. Of course, we can reduce the error probability by using amplification and Chernoff bounds. However, amplifying naïvely produces additional factors of $\log(\frac{1}{\varepsilon})$ or $\log\log(\frac{1}{\varepsilon})$ in the query complexity. To eliminate those factors and obtain a tight result, we solve an optimization problem to find a carefully-tuned amplification schedule, which then leads to a geometric series for the overall query complexity.

2.2 Approximate Counting

We are now ready to state and analyze our main algorithm.

Theorem 2.1. *Let $S \subseteq [N]$ be a nonempty set of marked items, and let $K = |S|$. Given access to a membership oracle to S and $\varepsilon, \delta > 0$, there exists a quantum algorithm that outputs an estimate $\hat{K}$. The output $\hat{K}$ satisfies*

$$K(1-\varepsilon) < \hat{K} < K(1+\varepsilon)$$

with probability at least $1-\delta$. There exists a function $Q(N, K, \varepsilon, \delta) \in O\left(\sqrt{N/K}\varepsilon^{-1}\log(\delta^{-1})\right)$ such that the algorithm makes fewer than $Q(N, K, \varepsilon, \delta)$ queries whenever the estimate is accurate. The algorithm needs $O(\log N)$ qubits of space.

Proof. The algorithm is as follows.

Algorithm: Approximate Counting

Inputs: $\varepsilon, \delta > 0$ and an oracle for membership in a nonempty set $S \subseteq [N]$.

Output: An estimate of $K = |S|$.

We can assume without loss of generality that $K \ll N$, for example by padding out the list with $10^6 N$ unmarked items. Let $\theta := \arcsin \sqrt{K/N}$; then since $K/N \leq (10^6+1)^{-1}$, we have $\theta \leq 0.001$.

Let U be the membership oracle, which satisfies $U|x\rangle = (-1)^{x \in S}|x\rangle$. Also, let $|\psi\rangle$ be the uniform superposition over all N items, and let $G := (I - 2|\psi\rangle\langle\psi|)U$ be the Grover diffusion operator.

1. For $k := 0, 1, 2, \ldots$:
   (a) Let r_k be the largest odd integer less than or equal to 1.05^k. Prepare the state $G^{(r_k-1)/2}|\psi\rangle$ and measure. Do this at least $5000 \cdot \ln(5/\delta)$ times.
   (b) If a marked item was measured $\geq 95\%$ of the time, exit the loop on $k =: k_{\text{end}}$.
2. Initialize $\theta_{\min} := 0.9 \cdot 1.05^{-k_{\text{end}}}$ and $\theta_{\max} := 1.65 \cdot \theta_{\min}$. Then, for $t := 0, 1, 2, \ldots$:
   (a) Use Lemma 2.2 to choose r_t.
   (b) Prepare the state $G^{(r_t-1)/2}|\psi\rangle$ and measure. Do this at least $250 \cdot \ln\left(\delta_t^{-1}\right)$ times, where $\delta_t := (\delta\varepsilon/65) \cdot (0.9)^{-t}$.
   (c) Let $\gamma := \theta_{\max}/\theta_{\min} - 1$. If a marked item was measured at least 12% of the time, set $\theta_{\min} := \theta_{\max}/(1 + 0.9\gamma)$. Otherwise, set $\theta_{\max} := (1 + 0.9\gamma)\theta_{\min}$.
   (d) If $\theta_{\max} \leq (1 + \varepsilon/5)\theta_{\min}$ then exit the loop.
3. Return $\hat{K} := N \cdot \sin^2(\theta_{\max})$ as an estimate for K.

The algorithm naturally divides into two pieces. First, step 1 computes an "initial rough guess" for the angle θ (and hence, indirectly, the number of marked items K), accurate up to *some* multiplicative constant, but not necessarily $1 + \varepsilon/5$. More precisely, step 1 outputs bounds $\theta_{\min}$ and $\theta_{\max}$, which are supposed to satisfy $\theta_{\min} \leq \theta \leq \theta_{\max}$ and satisfy $\theta_{\max}/\theta_{\min} \leq 1.65$. Next, step 2 improves this constant-factor estimate to a $(1 + \varepsilon/5)$-factor estimate of θ, yielding a $(1 + \varepsilon)$-factor estimate of K.

Both of these steps repeatedly prepare and measure the following quantum state [Grover96]:

$$G^{(r-1)/2}|\psi\rangle = \frac{\sin(r\theta)}{\sqrt{K}} \sum_{x \in S} |x\rangle + \frac{\cos(r\theta)}{\sqrt{N-K}} \sum_{x \notin S} |x\rangle. \tag{2.1}$$

Note that, if this state is measured in the computational basis, then the probability of observing a marked item is $\sin^2(r\theta)$. This circuit requires $O(r)$ queries and needs $O(\log N)$ qubits to store $|\psi\rangle$.

In what follows, we'll first prove that step 1 indeed returns a constant-factor 1.65 approximation to θ with probability $\geq 1 - \delta/2$. When it succeeds at this, we show that its query complexity is also $O\left(\sqrt{N/K} \log\left(\delta^{-1}\right)\right)$. Next, we show that step 2 improves the estimate to a $(1+\varepsilon/5)$-factor approximation with probability $\geq 1-\delta/2$, and deterministically requires an additional $O\left(\sqrt{N/K}\varepsilon^{-1} \log\left(\delta^{-1}\right)\right)$. The total failure probability is $\leq \delta/2 + \delta/2 = \delta$.

Correctness of step 1. First, we describe the ideal behavior: we want to terminate at a k_{end} such that the resulting bounds $\theta_{\min}, \theta_{\max}$ are accurate. Let k_0 be given by:

$$k_0 := \text{the largest integer such that } \theta \cdot 1.05^{k_0} \leq 0.9 \tag{2.2}$$

Ideally, for values of k satisfying $0 \leq k \leq k_0$, we do not terminate. Then we might terminate at k_0+1, k_0+2, etc., but if we reach k_0+10 then we definitely terminate there. Given that we have $k_0+1 \leq k_{\text{end}} \leq k_0+10$, we derive:

$$k_{\text{end}} - 10 \leq k_0 \leq k_{\text{end}} - 1 \tag{2.3}$$

$$\theta \cdot 1.05^{k_{\text{end}}-10} \leq \theta \cdot 1.05^{k_0} \leq 0.9 \leq \theta \cdot 1.05^{k_0+1} \leq \theta \cdot 1.05^{k_{\text{end}}} \tag{2.4}$$

$$0.9 \cdot 1.05^{-k_{\text{end}}} \leq \theta \leq 0.9 \cdot 1.05^{-k_{\text{end}}} \cdot 1.05^{10} \tag{2.5}$$

$$\leq 0.9 \cdot 1.05^{-k_{\text{end}}} \cdot 1.65 \tag{2.6}$$

We have proved that in the ideal case we have $\theta \in [\theta_{\min}, \theta_{\max}]$. Also see how $\theta_{\max}/\theta_{\min} = 1.65$. Next we move on to demonstrating that this ideal case occurs with probability $\geq 1-\delta/2$. The proof splits into two cases: terminating too late and terminating too early.

We show that we do not terminate too late: if we reach $k = k_0+10$, then we terminate there with probability $\geq 1-\delta/5$. Since r_k is obtained by rounding downward to the nearest odd number, we can bound the rounding error:

$$r_{k_0+10} \geq 1.05^{k_0+10} - 2 \tag{2.7}$$

$$1.58 \geq 0.9 \cdot 1.05^{10} \geq 1.05^{k_0+10}\theta \geq 0.9 \cdot 1.05^{10} \geq 1.396 \tag{2.8}$$

$$\sin^2(r_{k_0+10}\theta) \geq \sin^2((1.05^{k_0+10} - 2)\theta) \geq 0.99 \cdot \sin^2(1.05^{k_0+10}\theta). \tag{2.9}$$

In the final equation we leverage that, on the interval $[1.396, 1.58]$, we have that $\sin^2(x)$ is increasing and that $\sin^2(x - 0.002) > 0.99 \cdot \sin^2(x)$. We also use that $\theta < 0.001$, and that the error in $\sin^2(r\theta)$ due to rounding r only gets better with decreasing θ.

From this we conclude $\sin^2(r_{k_0+10}\theta) \geq 0.99 \cdot \sin^2(1.396) \geq 0.96$. We see that:

$$\begin{aligned}
\Pr[\text{fail to terminate at } k = k_0 + 10] &= \Pr[\text{fraction of marked items} \leq 95\%] && (2.10)\\
&\leq \exp\Big(-2 \cdot 5000 \cdot (0.96 - 0.95)^2 \ln(5/\delta)\Big) && (2.11)\\
&= \exp(-\ln(5/\delta)) = \delta/5 && (2.12)
\end{aligned}$$

So all that remains to show is that we do not terminate too early: for all $k \leq k_0$ we do not see enough heads. Abbreviate $p_k = \sin^2(r_k\theta)$. For this calculation we demand the tighter version of the Chernoff-Hoeffding theorem: assuming $p_k < 95\%$, if we toss coin with bias p_k a total of m times, then the probability we see more than 95% heads is bounded by:

$$\Pr[\text{fraction observed heads} \geq 95\%] \leq \exp(-mD(95\%||p_k)) \tag{2.13}$$

where $D(95\%||p_k)$ is the Kullback-Leibler divergence of two Bernoulli trials, given by:

$$D(95\%||p_k) = (95\%) \ln \frac{95\%}{p_k} + (1 - 95\%) \ln \frac{1 - 95\%}{1 - p_k} \tag{2.14}$$

With $m = 5000 \ln(5/\delta)$, we can now bound:

$$\begin{aligned}
\Pr[\text{terminate at } k] &\leq \exp(-mD(95\%||p_k)) && (2.15)\\
&= \left[\left(\frac{95\%}{p_k}\right)^{-95\%} \left(\frac{5\%}{1 - p_k}\right)^{-5\%}\right]^m && (2.16)\\
&\leq \Big[(95\%)^{-95\%}(5\%)^{-5\%}\Big]^m \cdot p_k^{m\cdot 95\%} && (2.17)\\
&\leq 1.22^m \cdot p_k^{m\cdot 95\%} && (2.18)
\end{aligned}$$

Next, we plug in $p_k = \sin^2(r_k\theta) \leq (r_k\theta)^2 \leq (\theta \cdot 1.05^k)^2$, and use the union bound to bound the probability that we terminate at or before k_0:

$$\begin{aligned}
\Pr[\text{terminate} \leq k_0] &= \sum_{k=0}^{k_0} \Pr[\text{terminate at } k] && (2.19)\\
&\leq 1.22^m \cdot \sum_{k=0}^{k_0} (\theta \cdot 1.05^k)^{1.9m} && (2.20)\\
\sum_{k=0}^{k_0} (\theta \cdot 1.05^k)^{1.9m} &= \theta^{1.9m} \cdot \sum_{k=0}^{k_0} (1.05^{1.9m})^k && (2.21)\\
&= \theta^{1.9m} \cdot \frac{(1.05^{1.9m})^{k_0+1} - 1}{1.05^{1.9m} - 1} && (2.22)\\
&\leq \left(\theta \cdot 1.05^{k_0}\right)^{1.9m} \cdot \frac{1.05^{1.9m}}{1.05^{1.9m} - 1} && (2.23)\\
&\leq 0.9^{1.9m} \cdot 1.01 && (2.24)\\
\Pr[\text{terminate} \leq k_0] &\leq \left[1.22 \cdot 0.9^{1.9}\right]^m \cdot 1.01 && (2.25)\\
&\leq [0.9984]^{5000 \ln(5/\delta)} \cdot 1.01 && (2.26)\\
&\leq \left[e^{-1}\right]^{\ln(5/\delta)} \cdot 1.01 && (2.27)\\
&= e^{-\ln(5/\delta)} = (\delta/5) \cdot 1.01 \leq \delta/4 && (2.28)
\end{aligned}$$

The probability we terminate too early is $\leq \delta/4$, and the probability we terminate too late is $\leq \delta/5$, so the probability that the first step has the ideal behavior is $\geq 1 - \delta/2$.

Query complexity of step 1. When the first step behaves ideally, then it terminates at $k_{\text{end}} = k_0 + 10$ at the latest. Since another way of writing k_0 is $k_0 = \lfloor \log_{1.05}(0.9/\theta) \rfloor$, we have

$$\sum_{k=0}^{k_{\text{end}}} r_k \leq \sum_{k=0}^{k_0+10} 1.05^k \leq \frac{1.05^{k_0+11} - 1}{1.05 - 1} \in O(1.05^{k_0}) \leq O(\theta^{-1}) \tag{2.29}$$

Since at each iteration we run $O(\log(\delta^{-1}))$ circuits, the total complexity in the ideal case is $O(\theta^{-1} \log(\delta^{-1})) = O(\sqrt{N/K} \log(\delta^{-1}))$.

Correctness of step 2. Let $\gamma := \theta_{\max}/\theta_{\min} - 1$. By definition, the $\theta_{\min}, \theta_{\max}$ initialized at the beginning of step 2 satisfy $\gamma = 1.65 - 1 = 0.65 \leq 0.7$. We also see that $\theta_{\max} \leq 1.65\theta \leq 0.002$, so we satisfy the conditions of Lemma 2.2. The coin described in the lemma is implemented by measuring the state $G^{(r-1)/2}|\psi\rangle$.

Each iteration of step 2 will modify $\theta_{\min}$ or $\theta_{\max}$ in order to reduce γ by exactly a factor of 0.9. Each iteration preserves $\theta_{\min} \leq \theta \leq \theta_{\max}$ with high probability. When the algorithm terminates we have $\theta_{\max}/\theta_{\min} \leq 1 + \varepsilon/5$ which implies that any value $\hat{\theta}$ between $\theta_{\min}$ and $\theta_{\max}$ satisfies $(1 - \varepsilon/5)\theta \leq \hat{\theta} \leq (1 + \varepsilon/5)\theta$ as desired. A simple calculation[2] shows that these multiplicative error bounds on the estimate for θ guarantee the desired $(1 - \varepsilon)K \leq \hat{K} \leq (1 + \varepsilon)K$.

We start with $t = 0$, so at the beginning of step t we have $\gamma_t = 0.65 \cdot 0.9^t$. Let T, the iteration after which we terminate in step 2, be given by:

$$T := \text{the largest integer satisfying } \gamma_T = 0.65 \cdot (0.9)^T \geq \frac{\varepsilon}{5} \tag{2.30}$$

This way, after the T'th step we have $\gamma_{T+1} \leq \varepsilon/5$, so we stop. (Note that there are $T + 1$ total iterations.) For convenience we define $b := 1/0.9 \approx 1.11$. Note that $b > 1$, which makes $\log_b(x)$ behave intuitively. Then:

$$T \leq \log_b \left(\frac{3.25}{\varepsilon} \right) \tag{2.31}$$

We used Lemma 2.2 to guarantee that the failure probability at the t'th iteration is at most $\delta_t := (\delta\varepsilon/65) \cdot (0.9)^{-t} = (\delta\varepsilon/65) \cdot b^t$. By the union bound, the overall failure

[2]Show $1 - \varepsilon \leq \sin^2(\theta(1 + \varepsilon/5))/\sin^2(\theta) \leq 1 + \varepsilon$ by Taylor expanding $\sin^2(\theta(1 + \varepsilon/5))/\sin^2(\theta)$ around $\varepsilon = 0$ and truncating the series to obtain bounds.

probability is then at most:

$$\sum_{t=0}^{T} \delta_t = \frac{\delta\varepsilon}{65} \sum_{t=0}^{T} b^t \le \frac{\delta\varepsilon}{65} \frac{b^{T+1}-1}{b-1} \le \frac{\delta\varepsilon}{65} \frac{b}{b-1} \frac{3.25}{\varepsilon} = \frac{\delta}{2} \tag{2.32}$$

Query complexity of step 2. From Lemma 2.2, we know that $r_t \in O(\theta^{-1}\gamma_t^{-1})$. Recall that $\gamma_t = 0.65 \cdot b^{-t}$. Each step makes $O(\ln(\delta_t^{-1}))$ queries. So the total query complexity is given by:

$$\begin{aligned}
O\left(\sum_{t=0}^{T} \frac{1}{\theta}\frac{1}{\gamma_t} \ln\left(\frac{1}{\delta_t}\right)\right) &\le O\left(\frac{1}{\theta}\sum_{t=0}^{T} b^t \left[\ln\left(\frac{1}{\delta}\right) + \ln\left(\frac{1}{\varepsilon}\right) - t\ln(b)\right]\right) && (2.33)\\
&\le O\left(\frac{1}{\theta} b^T \ln\left(\frac{1}{\delta}\right) + \frac{1}{\theta}\sum_{t=0}^{T} b^t \left[\ln\left(\frac{1}{\varepsilon}\right) - t\ln(b)\right]\right) && (2.34)\\
&\le O\left(\sqrt{\frac{N}{K}}\frac{1}{\varepsilon} \ln\left(\frac{1}{\delta}\right)\right) + O\left(\frac{1}{\theta}\sum_{t=0}^{T} b^t \left[\ln\left(\frac{1}{\varepsilon}\right) - t\ln(b)\right]\right) && (2.35)
\end{aligned}$$

The first term is the desired complexity, so all that remains to show is that the second term is dominated by the first term. To do so, we compute some bounds on T:

$$\begin{aligned}
T+1 &\ge \log_b\left(\frac{3.25}{\varepsilon}\right) && (2.36)\\
\ln(b)(T+1) &\ge \ln(b)\frac{\ln\left(\frac{3.25}{\varepsilon}\right)}{\ln(b)} = \ln\left(\frac{3.25}{\varepsilon}\right) \ge \ln\left(\frac{1}{\varepsilon}\right) && (2.37)
\end{aligned}$$

Now we bound the second term, dropping the $1/\theta$. To aid intuition, recall that $b > 1$.

$$\sum_{t=0}^{T} b^t \left[\ln\left(\frac{1}{\varepsilon}\right) - t\ln(b)\right] = \ln\left(\frac{1}{\varepsilon}\right)\frac{b^{T+1}-1}{b-1} - \ln(b)\,\frac{b}{(b-1)^2}\left[Tb^{T+1} - (T+1)b^T + 1\right] \tag{2.38}$$

$$\leq \ln\left(\frac{1}{\varepsilon}\right)\frac{b^{T+1}}{b-1} + \ln(b)\,\frac{b}{(b-1)^2}\left[-Tb^{T+1} + (T+1)b^T\right] \tag{2.39}$$

$$= \ln\left(\frac{1}{\varepsilon}\right)\frac{b^{T+1}}{b-1} + \ln(b)\,\frac{b}{(b-1)^2}\left[b^{T+1} - (T+1)b^{T+1} + (T+1)b^T\right] \tag{2.40}$$

$$= \ln\left(\frac{1}{\varepsilon}\right)\frac{b^{T+1}}{b-1} + \ln(b)\,\frac{b}{(b-1)^2}\left[b^{T+1} - (T+1)b^T(b-1)\right] \tag{2.41}$$

$$= \ln\left(\frac{1}{\varepsilon}\right)\frac{b^{T+1}}{b-1} - \ln(b)\,(T+1)\frac{b^{T+1}}{b-1} + \ln(b)\,\frac{b^{T+2}}{(b-1)^2} \tag{2.42}$$

$$\leq \ln\left(\frac{1}{\varepsilon}\right)\frac{b^{T+1}}{b-1} - \ln\left(\frac{1}{\varepsilon}\right)\frac{b^{T+1}}{b-1} + \ln(b)\,\frac{b^{T+2}}{(b-1)^2} \tag{2.43}$$

$$= \ln(b)\,\frac{b^{T+2}}{(b-1)^2} \in O\left(\frac{1}{\varepsilon}\right) \tag{2.44}$$

□

We note that this algorithm can also be used to determine if there are no marked items. If there is at least one marked item, then $\theta \geq \arcsin\sqrt{1/N}$. That means, that there must be some point in step 1(b) where we are likely to see enough marked items. If we fail to see enough marked items then, then we must have $K = 0$ with high probability.

Next we prove Lemma 2.2, which constructs a number of rotations r such that when $\theta \approx \theta_{\max}$ it is very likely to see a marked item, and when $\theta \approx \theta_{\min}$ it is very unlikely to see a marked item.

Lemma 2.2. ***Rotation lemma.*** *Say we are given $\theta_{min}, \theta_{max}$ such that $0 < \theta_{min} \leq \theta \leq \theta_{max} \leq 0.002$ and $\theta_{max} = (1+\gamma)\cdot\theta_{min}$ for some $\gamma \leq 0.7$. Then we can calculate an odd* *integer $r \in O(\theta^{-1}\gamma^{-1})$ such that the following is true: Consider tossing a coin that is heads with probability $\sin^2(r\theta)$ at least $250 \cdot \ln(\delta^{-1})$ times, and subsequently*

1. *if more than 12% heads are observed set θ_{min} to $\theta_{max}/(1+0.9\gamma)$,*
2. *and otherwise set θ_{max} to $(1+0.9\gamma)\theta_{min}$.*

This process fails to maintain $\theta_{min} \leq \theta \leq \theta_{max}$ with probability less than δ.

Proof. We compute r as follows (when rounding, ties between integers can be broken arbitrarily):

$$\Delta\theta := \theta_{\max} - \theta_{\min} \tag{2.45}$$

$$k := \text{ the closest integer to } \frac{\theta_{\min}}{2\Delta\theta} \tag{2.46}$$

$$r := \text{ the closest odd integer to } \frac{\pi k}{\theta_{\min}} \tag{2.47}$$

First we show that $r\theta_{\min} \approx \pi k$ and $r\theta_{\max} \approx \pi k + \frac{\pi}{2}$. We notice some basic facts about $\Delta\theta$ following from $\theta_{\max} = (1+\gamma)\cdot\theta_{\min}$:

$$\frac{\Delta\theta}{\theta_{\min}} = \gamma \tag{2.48}$$

$$\frac{\Delta\theta}{\theta_{\max}} = 1 - \frac{1}{1+\gamma} \tag{2.49}$$

Now we bound the rounding errors on k and r:

$$\left| k - \frac{\theta_{\min}}{2\Delta\theta} \right| \leq \frac{1}{2} \tag{2.50}$$

$$\left| r - \frac{\pi k}{\theta_{\min}} \right| \leq 1 \tag{2.51}$$

We arrive with bounds on $r\theta_{\min}$ and $r\theta_{\max}$:

$$\begin{aligned}
|r\theta_{\min} - \pi k| &\leq \theta_{\min} \leq 0.002 && (2.52)\\
\left|\pi k \frac{\Delta\theta}{\theta_{\min}} - \frac{\pi}{2}\right| &\leq \frac{\pi}{2} \cdot \frac{\Delta\theta}{\theta_{\min}} = \frac{\gamma\pi}{2} && (2.53)\\
\left|r\theta_{\max} - \left(\pi k + \frac{\pi}{2}\right)\right| &\leq \left|\pi k \frac{\theta_{\max}}{\theta_{\min}} - \left(\pi k + \frac{\pi}{2}\right)\right| + \theta_{\max} && (2.54)\\
&= \left|\left(\pi k + \pi k \frac{\Delta\theta}{\theta_{\min}}\right) - \left(\pi k + \frac{\pi}{2}\right)\right| + \theta_{\max} && (2.55)\\
&\leq \frac{\gamma\pi}{2} + \theta_{\max} && (2.56)\\
&\leq 0.7 \cdot \frac{\pi}{2} + 0.002 \leq 1.102 && (2.57)
\end{aligned}$$

Given these bounds, we can examine the two cases when we fail to preserve $\theta_{\min} \leq \theta \leq \theta_{\max}$ and demonstrate that they are unlikely: we could see many heads even though θ is small, thus fail to preserve $\theta_{\min} \leq \theta$, or we could see few heads even though θ is large, and thus fail to preserve $\theta \leq \theta_{\max}$.

First, suppose that $\theta_{\min} \leq \theta \leq \theta_{\max}/(1 + 0.9\gamma)$, so θ is near the bottom of the interval. We are to show that it is unlikely that we see too many heads.

$$\begin{aligned}
r\theta &\geq r\theta_{\min} \geq \pi k - 0.002 && (2.58)\\
r\theta &\leq \frac{r\theta_{\max}}{1 + 0.9\gamma} && (2.59)\\
&\leq r\theta_{\max} - \left(r\theta_{\max} - \frac{r\theta_{\max}}{1 + 0.9\gamma}\right) && (2.60)\\
&\leq r\theta_{\max} - r\Delta\theta \frac{\theta_{\max}}{\Delta\theta}\left(1 - \frac{1}{1 + 0.9\gamma}\right) && (2.61)\\
&\leq r\theta_{\max} - r\Delta\theta \frac{1 - \frac{1}{1+0.9\gamma}}{1 - \frac{1}{1+\gamma}} && (2.62)
\end{aligned}$$

We find $\frac{1-\frac{1}{1+0.9\gamma}}{1-\frac{1}{1+\gamma}} \geq 0.9$: viewing the left hand side as a function of γ, this function increases with γ, so the smallest value occurs as $\gamma \to 0$. Proceeding with the upper bound on $r\theta$:

$$
\begin{aligned}
r\theta &\leq r\theta_{\max} - 0.9 \cdot r\Delta\theta && (2.63)\\
&\leq 0.1 \cdot r\theta_{\max} + 0.9 \cdot r\theta_{\min} && (2.64)\\
&\leq \pi k + 0.1 \cdot \left(\frac{\pi}{2} + 1.102\right) + 0.9 \cdot 0.002 && (2.65)\\
&\leq \pi k + 0.27. && (2.66)
\end{aligned}
$$

Since $-0.002 \leq r\theta - \pi k \leq 0.27$, we have $\sin^2(r\theta) \leq 7.5\%$. So, the probability we fail is bounded by:

$$\Pr[\text{see more than 12\% heads}] \leq \exp(-2 \cdot (12\% - 7.5\%)^2 \cdot 250 \ln(\delta^{-1})) \leq \delta \qquad (2.67)$$

Now, we consider the case $\theta_{\min}(1 + 0.9\gamma) \leq \theta \leq \theta_{\max}$ where θ is near the top of the interval. We must show that it is unlikely that we see too few heads.

$$
\begin{aligned}
r\theta &\leq r\theta_{\max} \leq \pi k + \frac{\pi}{2} + 1.102 \leq \pi k + 2.68 && (2.68)\\
r\theta &\geq r\theta_{\min}(1 + 0.9\gamma) && (2.69)\\
&= r\theta_{min} + 0.9 \cdot r\Delta\theta && (2.70)\\
&= 0.9 \cdot r\theta_{\max} + 0.1 \cdot r\theta_{\min} && (2.71)\\
&\geq 0.9 \cdot \left(\pi k + \frac{\pi}{2} - 1.102\right) + 0.1 \cdot (\pi k - 0.002) && (2.72)\\
&\geq \pi k + 0.9 \cdot \frac{\pi}{2} - 0.9 \cdot 1.102 - 0.1 \cdot 0.002 && (2.73)\\
&\geq \pi k + 0.42. && (2.74)
\end{aligned}
$$

Since $0.42 \leq r\theta - \pi k \leq 2.68$ we have $\sin^2(r\theta) \geq 16.5\%$. So, the probability we see too few heads is bounded by:

$$\Pr[\text{see fewer than 12\% heads}] \leq \exp(-2 \cdot (16.5\% - 12\%)^2 \cdot 250 \ln(\delta^{-1})) \leq \delta \qquad (2.75)$$

We have shown correctness. Finally, we see that:

$$r \in O\left(\frac{k}{\theta_{\min}}\right) \leq O\left(\frac{1}{\Delta\theta}\right) \leq O\left(\frac{1}{\gamma\theta}\right). \tag{2.76}$$

□

2.3 Amplitude Estimation

We now show how to generalize our algorithm for approximate counting to amplitude estimation: given two quantum states $|\psi\rangle$ and $|\phi\rangle$, estimate the magnitude of their inner product $a = |\langle\psi|\phi\rangle|$. We are given access to these states via a unitary U that prepares $|\psi\rangle$ from a starting state $|0^n\rangle$, and also marks the component of $|\psi\rangle$ orthogonal to $|\phi\rangle$ by flipping a qubit. Recall that our analysis of approximate counting was in terms of the 'Grover angle' $\theta := \arcsin\sqrt{K/N}$. By redefining $\theta := \arcsin a$, the entire argument can be reused.

Theorem 2.3. *Given $\varepsilon, \delta > 0$ as well as access to an $(n+1)$-qubit unitary U satisfying*

$$U|0^n\rangle|0\rangle = a|\phi\rangle|0\rangle + \sqrt{1-a^2}|\tilde{\phi}\rangle|1\rangle,$$

where $|\phi\rangle$ and $|\tilde{\phi}\rangle$ are arbitrary n-qubit states and $0 < a < 1$,[3] *there exists an algorithm that outputs an estimate $\hat{a}$ that satisfies*

$$a(1-\varepsilon) < \hat{a} < a(1+\varepsilon)$$

and uses $O\left(a^{-1}\varepsilon^{-1}\log(\delta^{-1})\right)$ applications of U and $U^\dagger$ with probability at least $1-\delta$.

Proof. The algorithm is as follows.

[3]Note that we can always make a real by absorbing phases into $|\phi\rangle, |\tilde{\phi}\rangle$.

Algorithm: Amplitude Estimation

Inputs: $\varepsilon, \delta > 0$ and a unitary U satisfying $U|0^n\rangle|0\rangle = a|\phi\rangle|0\rangle + \sqrt{1-a^2}|\tilde{\phi}\rangle|1\rangle$.

Output: An estimate of a.

Let R satisfy $R|0\rangle = \frac{1}{1001}|0\rangle + \sqrt{1 - \left(\frac{1}{1001}\right)^2}|1\rangle$. Then:

$$U|0^n\rangle|0\rangle \otimes R|0\rangle = \frac{a}{1001}|\phi\rangle|00\rangle + \text{ terms orthogonal to } |\phi\rangle|00\rangle \tag{2.77}$$

Define $\theta := \arcsin \frac{a}{1001}$ and we have $0 \leq \theta \leq 0.001$. Let the Grover diffusion operator G be:

$$G := -(U \otimes R)(I - 2|0^{n+2}\rangle\langle 0^{n+2}|)(U \otimes R)^\dagger(I_{n+2} - 2(I_n \otimes |00\rangle\langle 00|)) \tag{2.78}$$

1. Follow steps 1 and 2 in the algorithm for approximate counting. An item is 'marked' if the final two qubits are measured as $|00\rangle$.
2. Return $\hat{a} := 1001 \cdot \sin(\theta_{\max})$ as an estimate for a.

If we write

$$(U \otimes R)|0^{n+2}\rangle = \sin\theta|\phi 00\rangle + \cos\theta|\phi 00^\perp\rangle \tag{2.79}$$

where $|\phi 00^\perp\rangle$ is the part of the state orthogonal to $|\phi 00\rangle$, then the Grover operator G rotates by an angle 2θ in the two-dimensional subspace spanned by $\{|\psi 00\rangle, |\psi 00^\perp\rangle\}$. Therefore:

$$G^{(r-1)/2}(U \otimes R)|0^{n+2}\rangle = \sin(r\theta)|\phi 00\rangle + \cos(r\theta)|\phi 00^\perp\rangle \tag{2.80}$$

making the probability of observing $|00\rangle$ on the last two qubits equal to $\sin^2(r\theta)$. This is the quantity bounded by Lemma 2.2.

The remainder of the proof is identical to the one for approximate counting, which guarantees that $\theta_{\max}$ is an estimate of θ up to a $1 + \varepsilon/5$ multiplicative factor. A simple calculation shows that $1001 \cdot \sin(\theta_{\max})$ is then an estimate of a up to a $1 + \varepsilon$ multiplicative factor. $\square$

2.4 Open Problems

What if we limit the "quantum depth" of our algorithm? That is, suppose we assume that after every T queries, the algorithm's quantum state is measured and destroyed. Can we derive tight bounds on the quantum query complexity of approximate counting in that scenario, as a function of T? This question is of potential practical interest, since near-term quantum computers *will* be severely limited in quantum depth. It's also of theoretical interest, since new ideas seem needed to adapt the polynomial method [Beals&98] or the adversary method [Ambainis02] to the depth-limited setting (for some relevant work see [JMdW13]).

Can we do approximate counting with the optimal $O\left(\frac{1}{\varepsilon}\sqrt{\frac{N}{K}}\right)$ quantum query complexity (and ideally, similar runtime), but with Grover invocations that are parallel rather than sequential, as in the algorithm of Suzuki et al. [Suzuki&19]?

Chapter 3

An Introduction to Block Encodings

This chapter is a review of [GSLW18], which is itself a culmination of a long line of research. We also present some material from the appendices of [LC17, Rall20]. No text in this chapter is copied verbatim from somewhere else.

Block encodings have completely revolutionized the way we think of quantum algorithms. Before 2010 or so, the primary tools of the trade in quantum algorithms were phase estimation [Kit95], the Trotter approximation [KKR04], and algorithms based on quantum walks [Sze04]. All of these tools succeed in making quantum computers do something useful while crucially making all the operations unitary. Block encodings lift the restriction of unitarity, letting us consider arbitrary matrices. In later chapters, we find that this makes the design of quantum algorithms incredibly natural and intuitive. Many of the most modern quantum algorithms, e.g. those for ground state finding [LT20] and thermal state preparation [CS16] make use of these techniques.

But it took a long time to get there. Block encoding techniques are a confluence of many research results that emerged in the 2010s, most of them focusing on improving the performance of Hamiltonian simulation. Two ideas form the core of the techniques: Linear Combinations of Unitaries (LCU) and Singular Value Transformation (SVT). We briefly outline their history here, although the references are in no way complete.

LCUs appeared in [CW12] and was later refined by [Berry&14, BCK15] and [CKS15]: the idea was to expand the Hamiltonian simulation operator e^{iHt} into a sum of many unitaries. Such a sum can be implemented by preparing a control register with the weights of the combination, applying the unitaries based on the control register, and then postselecting the control register.

SVT originated from a generalization of Grover's algorithm. In [YLC&14], explored using arbitrary rotations rather than reflections, and found an improved algorithm for fixed point search. Building on this idea, [LYC16, LC1606, LC1610, LC17] found that these sequences of rotations could build polynomials, and that these polynomials could approximate the function e^{ix}. The task of finding angle sequences for approximation of functions is called Quantum Signal Processing (QSP). QSP remains an active area of research, due to numerical instabilities that appear in the process [Haah18, Chao&20]. The 'qubitization' trick [LC1610] observes that the polynomials can be applied to each of the eigenvalues of a Hamiltonian by using the Jordan lemma (see e.g., [Sze04]) to split the Hilbert space into a direct sum of many qubits,

All of these techniques as well as many of their applications were collected in [GSLW18] and significantly refined. [GSLW18] is a seminal result of enormous importance, in large part because it is extremely comprehensive. It will surely remain a standard reference for quantum algorithms going forward. However, its thorough and all-embracing analysis comes at an enormous cost of readability. Personally, it took me countless attempts at reading the paper to finally understand it. Since block encodings are such an important technique in quantum algorithms, we require a more accessible introduction to the subject.

The review article [MRTC21] already tackles the challenge of making block encoding techniques more accessible. Their approach is to focus on the applications of block encoding techniques to eigenvalue thresholding, phase estimation, Hamiltonian simulation and matrix inversion. Their technical discussion focuses on QSP - the synthesis of functions through alternating phase rotations. This is very sensible since most applications of block encodings center around transforming the input with a function.

We find that a greater emphasis could be placed on how block encodings create a modular framework for the design of quantum algorithms. Rather than focusing on a single technique, we show how all the techniques fit together into a neat framework. To do so, we leverage some language from the programming languages community: we think of circuits as symbolic expressions that 'denote' certain matrices. We find there exist 'block encoding circuits' that can compile to regular quantum circuits. Second, rather than focusing on QSP, we offer a simplified derivation of qubitization: showing how a sequence of angles that build a function actually is applied to the singular values of a matrix. We find this also lets us understand some of the more subtle aspects of SVT. Thus, this review article complements the efforts of [MRTC21] to make [GSLW18] more accessible.

The key ingredient to escaping the restrictions of unitarity is postselection. While quantum operations themselves are still unitary, postselecting a register to be $|0\rangle$ can force a quantum computer to apply something non-unitary. To make use of this tool, we define a notion of a quantum circuit with postselection built into it.

Definition 3.1. *A* ***postselective unitary quantum circuit*** *is an expression of the fol-*

lowing form:

$$\begin{aligned} C := {} & U && \textit{(a unitary matrix on } \mathbb{C}^{2^n}\textit{)} && (3.1)\\ \mid {} & |0\rangle && \textit{(an initial state on } \mathbb{C}^{2^n}\textit{)} && (3.2)\\ \mid {} & C \cdot C && \textit{(matrix multiplication)} && (3.3)\\ \mid {} & C \otimes C && \textit{(tensor product)} && (3.4)\\ \mid {} & C^\dagger && \textit{(adjoint)} && (3.5)\\ \mid {} & I \oplus C && \textit{(control).} && (3.6) \end{aligned}$$

The expressions $|0\rangle$ and U are called ***primitives****.*

Consider evaluating a circuit C and obtaining a matrix M as a result: then we say C ***denotes*** *M. Say M has shape $2^n \times 1$ for some n: then M is really a state $|\psi\rangle$ and we say C* ***denotes the state*** *$|\psi\rangle$. When we write $C = M$ or $C = |\psi\rangle$, we mean that C denotes M or $|\psi\rangle$.*

Say C denotes M of shape $2^n \times 2^m$ and C' denotes M' of shape $2^{n'} \times 2^{m'}$. If a circuit contains an expression $C \cdot C'$, but $2^m \neq 2^{n'}$, making matrix multiplication $M \cdot M'$ undefined, then we say the circuit is ***invalid****. A circuit can also be invalid if it contains an expression $I \oplus C$ where C does not denote something square. All circuits in this manuscript are assumed to not be invalid.*

The ***size*** *or* ***complexity of a circuit*** *C is the number of bits required to write down C.*

A postselective circuit could be $(|0\rangle)^\dagger \cdot H \cdot |0\rangle = \langle 0| H |0\rangle = \frac{1}{\sqrt{2}}$, which clearly denotes something non-unitary. Of course, a quantum computer cannot actually tell us the matrix elements of a matrix denoted by a circuit C. Instead, we have the sampling access given to us by quantum mechanics:

Definition 3.2. ***Born rule.*** *Say that a quantum circuit C denotes a scalar $\alpha \in \mathbb{C}$. Observe that $|\alpha|^2 \leq 1$. Then, a quantum computer can output 'heads' with probability $|\alpha|^2$ and 'tails' with probability $1 - |\alpha|^2$.*

Of course, the Born rule is a little bit stronger than this: if C denotes a normalized quantum state $\sum_i \alpha_i |i\rangle$, then we can sample i with probability $|\alpha_i|^2$. But the fact that the state must be normalized disallows postselection, or at least complicates it. We find that the above capability is enough to perform the amplitude estimation algorithm from the previous chapter, which is in turn the only capability required by the next chapter on estimating physical quantities.

Now we are ready to state the definition of block encoding circuits, which let us perform all the operations that are natural for complex matrices.

Definition 3.3. *A **block encoding circuit** is an expression of the following form:*

$$B := M \quad \textit{(any } 2^n \times 2^m \textit{ matrix)} \tag{3.7}$$

$$\textit{or } B \cdot B \quad \textit{(matrix multiplication)} \tag{3.8}$$

$$\textit{or } B + B \quad \textit{(matrix addition)} \tag{3.9}$$

$$\textit{or } B \otimes B \quad \textit{(tensor product)} \tag{3.10}$$

$$\textit{or } B^\dagger \quad \textit{(adjoint / conjugate transpose)} \tag{3.11}$$

$$\textit{or } p(B) \quad \textit{(singular value transformation by a polynomial } p\textit{).} \tag{3.12}$$

The expression M is the only ***primitive****. Notions of denotation, validity and size are just as in Definition 3.1, with the exception the definition of the $p(B)$ operation which is given later in Theorem 3.17.*

Just like a postselective quantum circuit, a quantum computer is able to sample probabilities denoted by circuits denoting scalars. However, with postselective circuits we had the guarantee that the circuit always evaluates to an amplitude: $a \in \mathbb{C}$ satisfying $|a| \leq 1$. But with block encoding circuits we have no such guarantee. Thus we need to establish a notion of 'scale' that scales down the circuit to something with bounded spectral norm.

Theorem 3.4. ***Compiling Block Encoding Circuits.*** *Say B is a block encoding circuit of size s denoting M. Then there exists a constant $\alpha \in \mathbb{R}^+$ and a quantum circuit C of size poly(s) such that C denotes M/α.*

The rest of this chapter is primarily dedicated to fleshing out the above theorem.

3.1 Defining Block Encodings

Postselective quantum circuits can already denote non-unitary matrices, and thereby already have many of the capabilities required by block encodings built in. If the number of initializations $|0\rangle$ and adjoint initializations $\langle 0|$ is not the same, then then can even denote non-square matrices. But there are several other capabilities that they do not keep track of. In the previous section we already mentioned scale: if C denotes M then we always have $|M| \leq 1$. But block encoding circuits can denote matrices with larger spectral norm.

There are three other features we would like to keep track of: accuracy, complexity and ancilla count. Tracking accuracy is particularly important for singular value transformation, since we often would like to transform our matrices by a function approximated by a polynomial. Tracking complexity is key for analyzing the performance of quantum algorithms, and keeping track of ancilla count can sometimes let us avoid uncomputation.

But what does the 'ancilla count' of a circuit C even mean? One way of answering this question is to take all the $|0\rangle$'s pull them out to the very beginning of the circuit, and to take all the $\langle 0|$'s and pull them out to the very end of the circuit. We can always do this due to the principle of deferred measurement. We end up with a circuit $U(C)$ we call the 'unitary sub-circuit' that makes no use of the '$|0\rangle$' primitive at all. Then, by counting the number of $|0\rangle$'s and $\langle 0|$'s we can reasonably define ancilla count.

Lemma 3.5. ***Principle of deferred measurement.*** *Say C is a quantum circuit that denotes a matrix M of shape $2^n \times 2^m$. Then there is a* ***unitary sub-circuit*** *$U(C)$ that denotes a unitary matrix U of shape $2^{\max(n,m)+k(C)} \times 2^{\max(n,m)+k(C)}$ for some* ***ancilla count*** *$k(C)$. This circuit $U(C)$ makes no use of the $|0\rangle$ circuit component and has the property that:*

$$(\langle 0|^{\otimes k(C)} \otimes \langle 0|^{\otimes \max(m-n,0)} \otimes I^{\otimes n}) \cdot U(C) \cdot (|0\rangle^{\otimes k(C)} \otimes |0\rangle^{\otimes \max(n-m,0)} \otimes I^{\otimes m}) \text{ denotes } M \tag{3.13}$$

Proof. The principle of deferred measurement is established in [NC00]. □

It turns out that the notion of $U(C)$ and $k(C)$ will be very useful for transforming old block encoding circuits into new ones. This view also helps us understand the term 'block encoding'. Say C denotes a square matrix A of shape $2^n \times 2^n$, and $U(C)$ denotes U_A of shape $2^{n+k(C)} \times 2^{n+k(C)}$. We see that U_A can be split into $2^{k(C)} \times 2^{k(C)}$ blocks, each containing a matrix of size $2^n \times 2^n$. In this view, U_A contains A in the top-left block:

$$U_A = \begin{bmatrix} A & \cdot & \dots & \cdot \\ \cdot & \cdot & \dots & \cdot \\ \vdots & \vdots & \ddots & \vdots \\ \cdot & \cdot & \dots & \cdot \end{bmatrix} \tag{3.14}$$

Having defined the ancilla count of a circuit, we are ready to give a definition of a block encoding with all of the features.

Definition 3.6. ***Full-featured block encoding.*** *Say A is a rectangular complex matrix. Say C_A is a postselective quantum circuit denoting a matrix $\tilde{A}$ with the same shape. Say α, ε are real numbers with $\alpha > 0$ and $\varepsilon \geq 0$. We say C_A* ***is an α-scaled ε-accurate block encoding of A with complexity Q and k ancillae*** *if:*

$$|\alpha\tilde{A} - A| \leq \varepsilon, \tag{3.15}$$

that is, $\alpha\tilde{A}$ is ε-close in spectral norm to A, and the size of C_A is Q, and the ancilla count of C_A is k. Additionally:

- *If 'α-scaled' is omitted we assume $\alpha = 1$. We can also say 'unscaled' rather than '1-scaled' to be explicit.*
- *If 'ε-accurate' is omitted we assume $\varepsilon = 0$. We can also say 'exact' rather than '0-accurate' to be explicit.*
- *If 'with complexity Q' is omitted then the size of C_A could be anything. We can also say 'with complexity at most Q' or 'with complexity $O(Q)$' to make weaker claims about the size of the circuit C_A.*
- *Similarly, if 'with l ancillae' is omitted then the ancilla count could be anything.*

Working towards Theorem 3.4, we can already show how to construct exact block encodings of arbitrary matrices M.

Theorem 3.7. ***Constructing a block encoding.*** *Say M is $2^n \times 2^m$ complex matrix. Then there exists a* $\max(1, |M|)$*-scaled block encoding of M with 1 ancilla.*

Proof. Let $M = U\Sigma V$ be a singular value decomposition of M, where U, V are unitary and Σ is a diagonal matrix with positive real elements. Let $\alpha := \max(1, |M|)$ and $\bar{\Sigma} := \Sigma/\alpha$, and observe that $|\Sigma| \leq 1$. We construct a circuit C which is a block encoding of $\bar{\Sigma}$. Then UCV is an α-scaled block encoding of M.

Since $\bar{\Sigma}$ can be rectangular, we can select a computational basis with a varying number of bits that still lets us sum over the diagonal. Let $k := \min(n, m)$, and let $x \in \{0,1\}^k$ be an index. Then, let:

$$|x^{(m)}\rangle := |0\rangle^{\otimes(m-k)} \otimes |x\rangle \tag{3.16}$$

$$|x^{(n)}\rangle := |0\rangle^{\otimes(n-k)} \otimes |x\rangle \tag{3.17}$$

Which lets us write:

$$\bar{\Sigma} := \sum_x \bar{\sigma}_x \, |x^{(n)}\rangle \langle x^{(m)}| \tag{3.18}$$

With this notation, we can construct C. We begin by making block encodings for the $\bar{\sigma}_x$:

$$U(\sigma) := \begin{bmatrix} \sigma & \sqrt{1-\sigma^2} \\ \sqrt{1-\sigma^2} & -\sigma \end{bmatrix} \tag{3.19}$$

and observe that $\langle 0| U(\sigma) |0\rangle = \sigma$. Then, let:

$$U_{\bar{\Sigma}} := \sum_x |x\rangle \langle x| \otimes U(\bar{\sigma}_x) \tag{3.20}$$

$$C := (|0\rangle^{\otimes(n-k)} \otimes I^{\otimes k} \otimes \langle 0|) U_{\bar{\Sigma}} (\langle 0|^{\otimes(m-k)} \otimes I^{\otimes k} \otimes |0\rangle) \tag{3.21}$$

We can build a quantum circuit for $U_{\bar{\Sigma}}$ directly via the 'U' primitive, since we know all of its matrix elements. We see that:

$$C = (|0\rangle^{\otimes(n-k)} \otimes I^{\otimes k}) \left(\sum_x |x\rangle \langle x| \otimes \langle 0| U(\bar{\sigma}_x) |0\rangle \right) (\langle 0|^{\otimes(m-k)} \otimes I^{\otimes k}) \tag{3.22}$$

$$= \sum_x \bar{\sigma}_x (|0\rangle^{\otimes(n-k)} \otimes |x\rangle)(\langle 0|^{\otimes(m-k)} \otimes \langle x|) = \sum_x \bar{\sigma}_x |x^{(n)}\rangle \langle x^{(m)}| = \bar{\Sigma} \tag{3.23}$$

□

Notice that we can also make block encodings of scalars this way, by pretending it is a 1×1 matrix. A complex scalar $\alpha = re^{i\theta}$ splits into a magnitude and a phase. The phase can be handled by one of the U or V unitaries, whereas r is positive and real and is handled by $U_{\bar{\Sigma}}$. We can then multiply a matrix by the scalar via a tensor product.

The method for constructing block encodings presented above only works for matrices M that are small enough to write down in full. There are actually several methods for constructing block encodings of larger matrices. This includes density matrices, POVM operators, Gram matrices, sparse matrices with sparse-access oracles, and matrices recorded in quantum data structures. These are detailed in [GSLW18] Section 4.2.

3.2 Block Encoding Circuit Algebra

In the previous section we showed how to implement the 'M' primitive in Definition 3.3. Continuing our work towards Theorem 3.4, in this section we show how to implement $B \cdot B$, $B + B$ and $B \otimes B$ via postselective quantum circuits.

Theorem 3.8. ***Algebra of block-encodings.*** *For some set of indices $i \in [n]$, say the circuits C_i denote α_i-scaled ε_i-accurate block encodings of A_i with k_i ancillae and complexities Q_i. Then:*

- *(Addition) There exists a circuit denoting a $\sum_i \alpha_i$-scaled $\sum \varepsilon_i$-accurate block encoding of $\sum_i A_i$ with $(\max_i(k_i) + \lceil \log_2(n) \rceil)$ ancillae and complexity $\sum_i Q_i + O(\max_i(k_i)) + O(\log^2(n))$.*
- *(Multiplication) There exists a circuit denoting a $(\prod_i \alpha_i)$-scaled $(\prod_i \alpha_i)\left(\sum_i \frac{\varepsilon_i}{\alpha_i}\right)$-accurate block encoding of $\prod_i A_i$ with $\sum_i k_i$ ancillae and complexity at most $\sum_i Q_i$.*
- *(Tensor product) There exists a circuit denoting a $(\prod_i \alpha_i)$-scaled $(\prod_i \alpha_i)\left(\sum_i \frac{\varepsilon_i}{\alpha_i}\right)$-accurate block encoding of $\bigotimes_i A_i$ with $\sum_i k_i$ ancillae and complexity at most $\sum_i Q_i$.*

Proof. (Addition) Say C_i denotes $\tilde{A}_i$. Let $\alpha := \sum_i \alpha_i$, and $\bar{\alpha}_i := \alpha_i/\alpha$. The circuit C_{add} requires a subcircuit C_{prep}, which denotes:

$$C_{\text{prep}} = \sum_i \sqrt{\bar{\alpha}_i}\, |i\rangle \tag{3.24}$$

where we have encoded the index $|i\rangle$ into $\lceil \log_2(n) \rceil$ qubits. We can build C_{prep} via any unitary that features the state $\sum_i \sqrt{\bar{\alpha}_i}\, |i\rangle$ in the first column, requiring $O(\log^2(n))$ bits to write down.

Then, let $U(C_i)$ be the unitary sub-circuit of the block encoding C_i. Since the $\tilde{A}_i$ all have the same shape $2^n \times 2^m$, we see that:

$$(\langle 0|^{\otimes k_i} \otimes \langle 0|^{\otimes \max(m-n,0)} \otimes I^{\otimes n}) \cdot U(C_i) \cdot (|0\rangle^{\otimes k_i} \otimes |0\rangle^{\otimes \max(n-m,0)} \otimes I^{\otimes m}) = \tilde{A}_i \tag{3.25}$$

If we let $k_{\text{max}} := \max_j(k_j)$, then we can pad out each of these to $\bar{U}(C_i) = I^{\otimes(k_{\text{max}}-k_i)} \otimes U(C_i)$ so that all the $\bar{U}(C_i)$ act on $k_{\text{max}}+\max(n,m)$ many qubits. With use of classical computation via Toffoli gates as well as the controlled circuit $I \oplus \bar{U}(C_i)$, we can construct a circuit C_{select} denoting:

$$C_{\text{select}} = \sum_i |i\rangle \langle i| \otimes \bar{U}(C_i) \tag{3.26}$$

Now we add the postselection bits to obtain C_{sum}:

$$(C_{\text{prep}}^\dagger \otimes \langle 0|^{\otimes k_{\text{max}}} \otimes \langle 0|^{\otimes \max(m-n,0)} \otimes I^{\otimes n}) \cdot C_{\text{select}} \cdot (|0\rangle^{\otimes k_{\text{max}}} \otimes |0\rangle^{\otimes \max(n-m,0)} \otimes I^{\otimes m} \otimes C_{\text{prep}}) \tag{3.27}$$

which denotes $\sum_i \bar{\alpha}_i \tilde{A}_i$. We claim that C_{sum} is an α-scaled approximate block encoding of $\sum_i A_i$, given that $|A_i - \alpha_i \tilde{A}_i| \le \varepsilon_i$. This just follows from the triangle inequality:

$$\left|\sum_i A_i - \alpha \sum_i \bar{\alpha}_i \tilde{A}_i\right| = \left|\sum_i A_i - \sum_i \alpha_i \tilde{A}_i\right| \le \sum_i |A_i - \alpha_i \tilde{A}_i| \le \sum_i \varepsilon_i. \tag{3.28}$$

(Multiplication) Here the circuit construction is much simpler. If C_i denotes $\tilde{A}_i$, then $\prod_i C_i$ denotes $\prod_i \tilde{A}_i$. If we let $\alpha := \prod_i \alpha_i$ (not the sum, as in the addition case), then all that remains to show is that:

$$\left|\prod_{i=1}^n A_i - \alpha \prod_{i=1}^n \tilde{A}_i\right| \le \alpha \sum_i \frac{\varepsilon_i}{\alpha_i} \tag{3.29}$$

We prove this by induction on n. The $n = 1$ case is trivial. Now we consider two terms:

$$|A_1 A_2 - (\alpha_1 \tilde{A}_1)(\alpha_2 \tilde{A}_2)| = |A_1 A_2 - A_1(\alpha_2 \tilde{A}_2) + A_1(\alpha_2 \tilde{A}_2) - (\alpha_1 \tilde{A}_1)(\alpha_2 \tilde{A}_2)| \tag{3.30}$$

$$= |A_1(A_2 - \alpha_2 \tilde{A}_2) + (A_1 - \alpha_1 \tilde{A}_1)(\alpha_2 \tilde{A}_2)| \tag{3.31}$$

$$\le \alpha_1 \varepsilon_2 + \varepsilon_1 \alpha_2 \tag{3.32}$$

The inductive step follows by plugging into the above relation.

(Tensor product) This just follows from the proof for multiplication: we can just pad each of the circuits C_i with tensor products of identities and multiply them together. $\square$

The above theorem presents generalizations of results in Sections 4.3 and 4.4 in [GSLW18]. Rather than focusing on pairwise addition, multiplication and tensor products, it shows how the errors propagate when we combine n many sub-circuits this way. Note that [GSLW18] also gives machinery for analyzing linear combinations where the coefficients may be approximate. We assume that the scale factors of each of the sub-circuits are known exactly, since this is usually the case.

3.3 Singular Value Transformation

This section is dedicated to the most complicated but also the most powerful tool in Definition 3.3. Just writing down what it does is quite involved, so as an introduction we present a version of it without the baggage of scale, accuracy or complexity.

Singular value transformation lets us transform block encoded matrices by polynomials. The parity of the polynomial is central to how the transformation behaves. We establish some vocabulary:

Definition 3.9. *A polynomial in x is said to be* ***even*** *or* ***even parity*** *if all of the nonzero terms have an even power of x. Similarly, it is* ***odd*** *if all the powers of x are odd. If a polynomial is either odd or even we say it has* ***fixed parity****, otherwise it has* ***mixed parity****. If p is even and q is odd or vice versa, then we say the polynomials have* ***opposite parity****.*

Now we are *almost* ready to state a simplified version of the capabilities of singular value transformation.

Theorem 3.10. ***Simple singular value transformation.*** *Say A is a matrix with singular value decomposition:*

$$A = \sum_i \sigma_i \left|l_i\right\rangle \left\langle r_i\right| \tag{3.33}$$

Say $P(x)$ is a polynomial with fixed parity and degree d, and let:

$$P(A) := \sum_i P(\sigma_i) \left|l_i\right\rangle \left\langle r_i\right| \;\; \textit{if } P \textit{ is odd} \tag{3.34}$$

$$P(A) := \sum_i P(\sigma_i) \left|r_i\right\rangle \left\langle r_i\right| \;\; \textit{if } P \textit{ is even.} \tag{3.35}$$

Suppose A has a block-encoding C_A, and suppose $P(x)$ is either PQ-completable or FG-completable. Then $P(A)$ has a block encoding that makes d uses of C_A's unitary sub-circuit.

This should give an overview of the capabilities of the operations, but in order to understand its limitations we need to understand what 'PQ-completable' and 'FG-completable' mean. In Section 3.3.1 we will show how the circuit for singular value transformation lets us implement certain functions $f(\sigma)$ of the singular values. In Section 3.3.2 we outline how to 'complete' polynomials so that they fit into the special form of $f(\sigma)$. Finally, in Section 3.3.3, we give a full description of singular value transformation with notions of accuracy, scale, complexity, and ancilla count.

3.3.1 Circuit Analysis

An enormous barrier to understanding singular value transformation is the qubitization technique from [LC1610]. A complete derivation is to be found in Section 3.2 of [GSLW18], but it is enormously complicated. Singular value transformation can largely be used without really understanding how it works in depth. But we find that making a simple assumption on the block encoded matrix A makes the derivation significantly simpler for that special case. A deeper understanding lets us perform some ancilla saving tricks that we discuss in Remark 3.19.

Theorem 3.11. ***Circuits for singular value transformation.*** *Say d is a positive integer and say $f(\sigma) : [-1,1] \to \mathbb{C}$ is a function of the form:*

$$f(\sigma) = \langle 0| e^{iX \arccos(\sigma)} \prod_{j=1}^{d-1} e^{iZ\phi_j} e^{iX \arccos(\sigma)} |0\rangle \tag{3.36}$$

for some phases ϕ_j with $j \in [d-1]$. Say A is a matrix with singular value decomposition:

$$A = \sum_i \sigma_i |l_i\rangle \langle r_i| . \tag{3.37}$$

and say furthermore that A has an unscaled exact block encoding C_A. Then there exists a circuit $C_{f(A)}$ with d applications of $U(C_A)$ that is an unscaled exact block encoding of $f(A)$, where $f(A)$ is defined as:

$$f(A) := \sum_i f(\sigma_i) |l_i\rangle \langle r_i| \;\; \textit{if } d \textit{ is odd} \tag{3.38}$$

$$f(A) := \sum_i f(\sigma_i) |r_i\rangle \langle r_i| \;\; \textit{if } d \textit{ is even} \tag{3.39}$$

Furthermore, C_A and $C_{f(A)}$ have the same ancilla count.

Assumption. We will assume that A is square, full rank, and all the singular values σ_i satisfy $0 < \sigma_i < 1$. The theorem above holds regardless of this assumption, but the proof becomes significantly more cumbersome. We focus on this special case because we feel there is a need in the literature for a more pedagogical introduction to this technique.

Proof. Let $\bar{\phi}_k := \phi_k - \frac{3}{2}\pi \bmod 2\pi$. Consider the quantum circuit:

$$(\langle 0| \otimes I) \cdot U_A^{(d)} \prod_{j=1}^{d-1} (e^{iZ\bar{\phi}_j} \otimes I) U_A^{(j)} \cdot (|0\rangle \otimes I) \tag{3.40}$$

Where $U_A^{(j)}$ is U_A when j is even and $U_A^\dagger$ when j is odd. We will show that this circuit denotes a block-encoding of $f(A)$. Let:

$$|\bar{l}_i\rangle := |0\rangle^{\otimes k} |l_i\rangle \qquad \langle\bar{r}_i| := \langle 0|^{\otimes k} \langle r_i| \tag{3.41}$$

Then, let $|\bar{l}_i^\perp\rangle$ and $\langle\bar{r}_i^\perp|$ be the unique states that satisfy:

$$U_A |\bar{r}_i\rangle = \sigma_i |\bar{l}_i\rangle + \sqrt{1-\sigma_i^2}\, |\bar{l}_i^\perp\rangle \tag{3.42}$$

$$\langle\bar{l}_i| U_A = \sigma_i \langle\bar{r}_i| + \sqrt{1-\sigma_i^2}\, \langle\bar{r}_i^\perp| \tag{3.43}$$

Their existence is guaranteed since by the fact that U_A is unitary, the fact that $\langle\bar{l}_i| U_A |\bar{r}_i\rangle = \sigma_i$. The assumption that $0 < \sigma_i < 1$ guarantees uniqueness (it turns out that these states can still be defined without this assumption, but this is pretty cumbersome to show explicitly). Furthermore, we can show that the sets $\{|\bar{l}_i\rangle, |\bar{l}_i^\perp\rangle\}$ and $\{|\bar{r}_i\rangle, |\bar{r}_i^\perp\rangle\}$ of $2N$ vectors each are an orthonormal basis (A is $N \times N$).

Given the action of U_A on $|\bar{l}_i\rangle, |\bar{l}_i^\perp\rangle, |\bar{r}_i\rangle, |\bar{r}_i^\perp\rangle$, we write U_A in the following form:

$$U_A = \sum_i \left(\sigma_i |\bar{l}_i\rangle\langle\bar{r}_i| + \sqrt{1-\sigma_i^2}\, |\bar{l}_i^\perp\rangle\langle\bar{r}_i| + \sqrt{1-\sigma_i^2}\, |\bar{l}_i\rangle\langle\bar{r}_i^\perp| - \sigma_i |\bar{l}_i^\perp\rangle\langle\bar{r}_i^\perp| \right) \tag{3.44}$$

Since this is pretty hard to read, we use the following abuse of notation:

$$U_A = \sum_i \begin{bmatrix} \sigma_i |\bar{l}_i\rangle\langle\bar{r}_i| & \sqrt{1-\sigma_i^2}\, |\bar{l}_i\rangle\langle\bar{r}_i^\perp| \\ \sqrt{1-\sigma_i^2}\, |\bar{l}_i^\perp\rangle\langle\bar{r}_i| & -\sigma_i |\bar{l}_i^\perp\rangle\langle\bar{r}_i^\perp| \end{bmatrix} \tag{3.45}$$

Next, we observe that the other terms in the quantum circuit have decompositions in terms of $|\bar{l}_i\rangle, |\bar{l}_i^\perp\rangle, |\bar{r}_i\rangle, |\bar{r}_i^\perp\rangle$ within this abuse of notation. This comes simply from the fact that

$I = \sum_i |l_i\rangle \langle l_i| = \sum_i |r_i\rangle \langle r_i|$.

$$(e^{iZ\bar{\phi}_j} \otimes I) = \sum_i \begin{bmatrix} e^{i\bar{\phi}_j} |\bar{l}_i\rangle \langle \bar{l}_i| & 0 \\ 0 & e^{-i\bar{\phi}_j} |\bar{l}_i^\perp\rangle \langle \bar{l}_i^\perp| \end{bmatrix} = \sum_i \begin{bmatrix} e^{i\bar{\phi}_j} |\bar{r}_i\rangle \langle \bar{r}_i| & 0 \\ 0 & e^{-i\bar{\phi}_j} |\bar{r}_i^\perp\rangle \langle \bar{r}_i^\perp| \end{bmatrix} \tag{3.46}$$

$$(|0\rangle \otimes I) = \sum_i \begin{bmatrix} |\bar{l}_i\rangle \\ 0 \end{bmatrix} \langle l_i| = \sum_i \begin{bmatrix} |\bar{r}_i\rangle \\ 0 \end{bmatrix} \langle r_i| \tag{3.47}$$

Expanding the terms in the circuit in this manner yields a series of matrix multiplications, alternating in the $|\bar{l}_i\rangle, |\bar{l}_i^\perp\rangle$ and $|\bar{r}_i\rangle, |\bar{r}_i^\perp\rangle$ subspaces. While still somewhat verbose to expand explicitly, it should still be clear that inserting the above expressions into the circuit (3.40) and canceling the bras and kets yields:

$$(\langle 0| \otimes I) \cdot U_A^{(d)} \prod_{j=1}^{d-1} (e^{iZ\phi_j} \otimes I) U_A^{(j)} \cdot (|0\rangle \otimes I) \text{ denotes} \tag{3.48}$$

$$= \sum_i \begin{bmatrix} 1 \\ 0 \end{bmatrix}^\dagger \cdot \begin{bmatrix} \sigma_i & \sqrt{1-\sigma_i^2} \\ \sqrt{1-\sigma_i^2} & -\sigma_i \end{bmatrix} \prod_{j=1}^{d-1} \begin{bmatrix} e^{i\bar{\phi}_j} & 0 \\ 0 & e^{-i\bar{\phi}_j} \end{bmatrix} \begin{bmatrix} \sigma_i & \sqrt{1-\sigma_i^2} \\ \sqrt{1-\sigma_i^2} & -\sigma_i \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \end{bmatrix} \tag{3.49}$$

$$\cdot |l_i\rangle \langle r_i| \text{ if } d \text{ is odd or } \cdot |r_i\rangle \langle r_i| \text{ if } d \text{ is even} \tag{3.50}$$

Notice also that we defined $|\bar{l}_i\rangle, |\bar{l}_i^\perp\rangle, |\bar{r}_i\rangle, |\bar{r}_i^\perp\rangle$ in such a manner to make the matrix elements of U_A real, so while the conjugate transpose transforms $|\bar{l}_i\rangle \langle \bar{r}_i| \to |\bar{r}_i\rangle \langle \bar{l}_i|$ and so on, the actual matrix elements $\sigma_i, \sqrt{1-\sigma_i}$ remain the same, so we no longer need to care about the alternating conjugate transpose. Finally, we make the observation that:

$$\begin{bmatrix} \sigma_i & \sqrt{1-\sigma_i} \\ \sqrt{1-\sigma_i} & -\sigma_i \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & -i \end{bmatrix} \begin{bmatrix} \sigma_i & i\sqrt{1-\sigma_i^2} \\ i\sqrt{1-\sigma_i^2} & \sigma_i \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & -i \end{bmatrix} \tag{3.51}$$

$$= e^{i\frac{3}{4}\pi} e^{iZ\frac{3}{4}\pi} e^{iX \arccos(\sigma_i)} e^{iZ\frac{3}{4}\pi} \tag{3.52}$$

Substituting this into (3.49) yields:

$$=e^{\frac{3d}{2}\pi} \cdot \langle 0| e^{iZ\frac{3}{4}\pi} e^{iX\arccos(\sigma_i)} e^{iZ\frac{3}{4}\pi} \prod_{k=1}^{d-1} e^{iZ\bar{\phi}_k} e^{iZ\frac{3}{4}\pi} e^{iX\arccos(\sigma_i)} e^{iZ\frac{3}{4}\pi} |0\rangle \tag{3.53}$$

$$=e^{\frac{3(d+1)}{2}\pi} \cdot \langle 0| e^{iX\arccos(\sigma_i)} \prod_{k=1}^{d-1} e^{iZ\left(\bar{\phi}_k+\frac{3}{2}\pi\right)} e^{iX\arccos(\sigma_i)} |0\rangle = e^{\frac{3(d+1)}{2}\pi} \cdot f(\sigma_i) \tag{3.54}$$

So we see the circuit (3.40) implements $f(A)$ up to an easily correctable global phase that depends only on d. □

3.3.2 Completing Polynomials

We have shown that we can transform the singular values of matrices by functions of the form:

$$f(\sigma) = \langle 0| e^{iX\arccos(\sigma)} \prod_{j=1}^{d-1} e^{iZ\phi_j} e^{iX\arccos(\sigma)} |0\rangle \tag{3.55}$$

Selecting angles ϕ_j to make $f(\sigma)$ have desired behavior is called quantum signal processing. In this section we outline the results of this line of work, without going into much detail with the derivations. A much more detailed review of this subject is given in [MRTC21].

We find that ϕ_j can be selected to make $f(x)$ a variety of polynomials $P(x) \in \mathbb{C}[x]$. It may be surprising that a form like $f(x)$ might yield a polynomial, but this should be more clear when looking at the most natural example of singular value transformation:

Example 3.12. ***Chebyshev polynomials.*** *Let $T_d(\sigma)$ be the function we obtain when we plug $\phi_j = 0$ into f. If we let $\theta := \arccos(\sigma)$, we see that:*

$$T_d(\sigma) = \langle 0| \prod_{j=1}^{d} e^{iX\theta} |0\rangle = \langle 0| e^{iXd\theta} |0\rangle = \cos(n\theta) \tag{3.56}$$

$T_d(\sigma) = \cos(n\arccos(\sigma))$ is exactly the definition of the Chebyshev polynomials of the first kind.

The Chebyshev polynomials are a very special family of polynomials with a close relationship to rotations. We can make a more general family of polynomials $P(x) \in \mathbb{C}[x]$, but in some sense we need to preserve this relationship. This places several restrictions on the polynomials we can build.

First, these polynomials must have fixed parity. Second, these polynomials must be 'completable', in the sense that there exists a $Q(x)$ satisfying:

$$P(x) + (1 - x^2)Q(x) = 1 \tag{3.57}$$

This stems simply from the fact that the matrix in the definition of $f(x)$ must be unitary. Indeed, Theorem 1 of [MRTC21] and Theorem 3 of [GSLW18], following a characterization of [LYC16] find that:

$$e^{iX \arccos(\sigma)} \prod_{j=1}^{d-1} e^{iZ\phi_j} e^{iX \arccos(\sigma)} = \begin{bmatrix} P(\sigma) & iQ(\sigma)\sqrt{1-\sigma^2} \\ iQ^*(\sigma)\sqrt{1-\sigma^2} & P^*(\sigma) \end{bmatrix} \tag{3.58}$$

Unitarity of the above matrix clearly imposes that constraint on P and Q. 'Completion' entails finding a suitable Q given a polynomial P. PQ-completion is one such method:

Theorem 3.13. ***PQ-completion.*** *Say $P(x) \in \mathbb{C}[x]$ is a complex polynomial with degree d and fixed parity. Then the following conditions are equivalent:*

1. *There exist phases ϕ_k with $k \in [d-1]$ and some phase ϕ_0 such that $P(x) = e^{i\phi_0} f(x)$ with $f(x)$ obtained from the ϕ_k as in (3.55).*

2. *There exists a complex polynomial $Q(x)$ with degree at most d and opposite parity to $P(x)$ such that for all real x with $|x| \leq 1$:*

$$P(x) + (1 - x^2)Q(x) = 1 \tag{3.59}$$

3. The following three conditions on $P(x)$ hold:

- *For all real x with $|x| \leq 1$ we have $|P(x)| \leq 1$.*
- *For all real x with $|x| \geq 1$ we have $|P(x)| \geq 1$.*
- *If $P(x)$ has even parity, then for all real x we have $P(ix)P(-ix)^* \geq 1$,*

Proof. The equivalence 1-2 is shown in Theorem 3 in [GSLW18], and the equivalence 2-3 is Theorem 4. □

Condition 3 of PQ-completion is very restrictive. In particular the condition that even polynomials satisfy $P(ix)P(-ix)^* \geq 1$ is very difficult to ensure. FG-completion is a technique that lets us specify the real part of P only, and let a completion method specify the complex part. This lets us extend the capabilities of quantum signal processing to a larger family of real polynomials.

The name FG-completion stems from the methods explored by [Haah18, Chao&20] which involves a Laurent polynomial $F(\omega)$. If we let $\omega = e^{i\arccos(\sigma)}$, we find that:

$$\mathrm{Re}[F(\omega)] = \mathrm{Re}\left[\sum_{k=-d}^{d} c_d \omega^d\right] = \sum_{k=-d}^{d} c_d \cos(d \arccos(\sigma)) = \sum_{k=-d}^{d} c_d T_{|d|}(\sigma) \tag{3.60}$$

So the polynomial $F(\omega)$ has a relationship with the Chebyshev expansion of $P(x)$. Then we proceed to find a Laurent polynomial $G(\omega)$ so that $F(\omega)F(1/\omega) + G(\omega)G(1/\omega)$, explaining the name 'FG-completion'. We do not go into the technical details further, and simply state the result.

Theorem 3.14. ***FG-completion***. *Say $P(x) \in \mathbb{R}[x]$ is a real polynomial with degree d and fixed parity. Then the following conditions are equivalent:*

1. *There exists a complex polynomial $\tilde{P}(x)$ of degree d such that $P(x) = Re(\tilde{P}(x))$, and the conditions in Theorem 3.13 hold for $\tilde{P}(x)$ and $\tilde{Q}(\sigma) = 0$*

2. *For all real x with $|x| \leq 1$ we have $|P(x)| \leq 1$.*

Proof. This follows from Theorem 5 in [GSLW18], and the results of [Haah18, Chao&20]. □

Observe how the constraint on $P(\sigma)$ with FG-completion is significantly less stringent.

To let the definitions of completability play we well with scale, we introduce the notion of completability on an interval. We need this to state the full-featured version of singular value transformation.

Definition 3.15. *Say α is a real number with $\alpha \geq 1$. Say the conditions in Theorem 3.13 hold for a polynomial $P(x/\alpha)$. Then we say $P(x)$* ***is PQ-completable on*** *$[-\alpha, \alpha]$. When we omit 'on $[-\alpha, \alpha]$' we mean that $\alpha = 1$.*

Similarly we say $P(x)$ ***is FG-completable on*** *$[-\alpha, \alpha]$ when the conditions in Theorem 3.14 hold for $P(x/\alpha)$.*

Theorems 3.13 and 3.14 merely assert when completion is possible. Actually finding the angles is a completely separate task, which runs into numerical instability issues if not done carefully.

Theorem 3.16. ***Angle finding.*** *Say P is PQ-completable on $[-\alpha, \alpha]$. Then for any $\delta > 0$ there exists a classical algorithm that, given the coefficients of P, computes approximate versions of the ϕ_0, ϕ_k in Theorem 3.13, such that the resulting $f(x)$ satisfies $|e^{i\phi_0} f(x) - P(x)| \leq \delta$ for all real x with $|x| \leq \alpha$.*

Say P is FG-completable on $[-\alpha, \alpha]$. Then for any $\delta > 0$ there exists a classical algorithm that, given given the coefficients of P, finds a PQ-completable polynomial $\tilde{P}$ on $[-\alpha, \alpha]$ as in Theorem 3.14 such that $|Re(\tilde{P}(x)) - P(x)| \leq \delta$ for real x with $|x| \leq \alpha$.

Proof. This is the main subject of [Haah18, Chao&20]. □

3.3.3 Transforming Block Encodings by Polynomials

Armed with qubitization and quantum signal processing, we can re-state Theorem 3.10 with all the features of block encodings stated in Definition 3.6. The primary change is that we now keep track of accuracy and scale, using the robust singular value transformation technique discussed in [GSLW18].

Theorem 3.17. ***Full-featured singular value transformation.*** *Say A is a matrix and $P(x)$ is a fixed parity polynomial with degree d. Let $P(A)$ be defined just like Theorem 3.10.*

Suppose that for any $\varepsilon > 0$ there exists an α-scaled ε-accurate block encoding of A with k ancillae and complexity $Q(\varepsilon)$. Furthermore, suppose that $P(x)$ is either PQ-completable or FG-completable on $[-\alpha, \alpha]$.

Then for any δ exists an unscaled δ-accurate block-encoding of $P(A)$ with k' ancillae and complexity Q'. We have:

$$Q' \in O\left[d \cdot Q\left(\frac{\delta^2\alpha^2}{16d^2}\right)\right] \tag{3.61}$$

If $P(x)$ is PQ-completable on $[-\alpha, \alpha]$ then $k' = k$. If instead $P(x)$ is FG-completable on $[-\alpha, \alpha]$ then $k' = k + 1$.

Proof. In addition to Theorem 3.11, we also leverage the robustness of singular value transformation discussed in Section 3.3 of [GSLW18].

To construct a block encoding of $P(A)$ when $P(\sigma)$ is FG-completable, we leverage the trick from Corollary 18 in [GSLW18]. Recall that FG-completion lets us make a block encoding of $\tilde{P}(A)$ where $\mathrm{Re}(\tilde{P}(\sigma)) = P(\sigma)$. By flipping the sign of all the ϕ_j, we can also make a block encoding of $\tilde{P}^*(A)$. Then $P(A) = (\tilde{P}(A) + \tilde{P}^*(A))/2$. $\square$

The limitation that the polynomial is fixed parity stems from the fact that adding together $\sum_i p(\sigma_i) |l_i\rangle \langle r_i|$ and $\sum_i p(\sigma_i) |r_i\rangle \langle r_i|$ does not necessarily give the desired result. Adding them together might not even be well defined if they have different shapes. But for the special case of hermitian A, we have $|r_i\rangle = |l_i\rangle$, letting us drop this restriction:

Theorem 3.18. ***Singular value transformation of hermitian matrices by arbitrary polynomials.*** *Say A is a hermitian matrix, and say for every $\varepsilon > 0$ there is an α-scaled ε-accurate block encoding of A with k ancillae and complexity $Q(\varepsilon)$.*

Now suppose $P(x)$ is any polynomial with degree d. Let $P_{max} = \max_{x\in[-\alpha,\alpha]} P(x)$. Then for every $\delta > 0$ there exists a $2P_{max}$-scaled δ-accurate block encoding of $P(A)$ with $k+2$ ancillae and complexity Q' where Q' is bounded as in Theorem 3.17.

Proof. This is Theorem 56 of [GSLW18]. The trick is just to split $P(x)$ into even and odd parts, apply Theorem 3.17 to each of them, and then to add them together using Theorem 3.8. $\square$

Finally, in Chapter 5, we will perform singular value transformation with an FG-completable polynomial. This is possible with Theorem 3.17, but it requires adding an extra ancilla qubit to combine $\tilde{P}(A)$ and $\tilde{P}^*(A)$. We find that this extra ancilla can be avoided with an additional trick.

Remark 3.19. ***Avoiding an extra ancilla with a special block encoding.*** *Say $P(x)$ is FG-completable and a block encoding of A has 1 ancilla. Then using theorem 3.17 we can make a block encoding of $P(A)$ with 2 ancillae. But what if we really want to avoid adding the extra ancilla?*

This is possible if the unitary sub-circuit $U(C_A)$ denotes a unitary of the special form:

$$\sum_i V_i \otimes |l_i\rangle \langle r_i| \tag{3.62}$$

where the V_i are all single-qubit reflections. Then we can observe that the states appearing in theorem 3.17 take the form:

$$|\bar{l}_i\rangle := |0\rangle |l_i\rangle \qquad \langle \bar{r}_i| := \langle 0| \langle r_i| \tag{3.63}$$

$$|\bar{l}_i^{\perp}\rangle = |1\rangle |l_i\rangle \qquad \langle \bar{r}_i^{\perp}| = \langle 1| \langle r_i| \tag{3.64}$$

The fact that the first register of $|\bar{l}_i^{\perp}\rangle, |\bar{r}_i^{\perp}\rangle$ is $|1\rangle$ is guaranteed by the fact that the block encoding has only one ancilla. We could however still have $|\bar{l}_i^{\perp}\rangle = e^{i\theta} |1\rangle |r_i\rangle$ and $\langle \bar{r}_i^{\perp}| = e^{i\theta'} \langle 1| \langle r_i|$ for some unknown phases θ, θ'. Hermiticity of U_A guarantees that these phases vanish.

The circuit in Theorem 3.11 uses alternating applications of $e^{iZ\phi_k}$ and $e^{iX \arccos(\sigma)}$, finally projecting onto the $|0\rangle \otimes I$. In this situation we will instead project onto $|+\rangle \otimes I$:

$$(\langle +| \otimes I) \cdot U_A^{(d)} \prod_{j=1}^{d-1} (e^{iZ\bar{\phi}_j} \otimes I) U_A^{(j)} \cdot (|+\rangle \otimes I) \tag{3.65}$$

In the abuse of notation used in theorem X we can derive that:

$$(|+\rangle \otimes I) = \sum_i \frac{1}{\sqrt{2}} \begin{bmatrix} |\bar{l}_i\rangle \\ |\bar{l}_i^{\perp}\rangle \end{bmatrix} \langle l_i| = \sum_i \frac{1}{\sqrt{2}} \begin{bmatrix} |\bar{r}_i\rangle \\ |\bar{r}_i^{\perp}\rangle \end{bmatrix} \langle r_i| \tag{3.66}$$

Note that this would be untrue if the phases θ, θ' were nonzero. The new circuit now denotes $f(A)$ where $f(\sigma)$ is given by:

$$f(\sigma) := \langle +| \, e^{iX \arccos(\sigma)} \prod_{k=1}^{d-1} e^{iZ\phi_j} e^{iX \arccos(\sigma)} \, |+\rangle \, . \tag{3.67}$$

Now we make use of Theorem 13 in [MRTC21], which states that expressions of the above form can evaluate to FG-completable polynomials. One way of seeing this is to plug in equation (3.58), to obtain:

$$f(\sigma) = 2Re(\tilde{P}(\sigma)) + iRe(\tilde{Q}(x))\sqrt{1-\sigma^2} \tag{3.68}$$

Now we simply select $Re(\tilde{P}(\sigma)) = P(\sigma)$ and $Re(\tilde{Q}(\sigma)) = 0$.

3.4 Constructing Polynomials

Theorem 3.17 draws its power from the fact that many functions can be approximated by polynomials. Polynomial approximation is already a well established field. One of the most powerful tools for constructing these approximations is Jackson's theorem.

Theorem 3.20. ***Jackson's Theorem.*** *Say $f(x)$ is a continuous function on the interval $[-\alpha, \alpha]$. Let $\omega_f(\delta)$ be the modulus of continuity of $f(x)$, defined by:*

$$\omega_f(\delta) := sup\big\{ \; |f(x) - f(y)| \text{ where } x, y \in [-\alpha, \alpha] \text{ such that } |x-y| \leq \delta \big\}. \tag{3.69}$$

Then, for any positive integer d there exists a polynomial $J(x)$ of degree at most d so that for all $x \in [-\alpha, \alpha]$:

$$|J(x) - f(x)| \leq 6\omega_f(\alpha/d), \tag{3.70}$$

Proof. This is proved in Chapter 1 of [Rivlin69]. The main idea is to construct $J(x)$ via a Chebyshev expansion of $f(x)$, and then figuring out where to truncate. □

While Jackson's theorem is very powerful, it does not suffice for some applications. One such application is amplitude amplification, where we really want the degree to scale as $O(a^{-1}\log(\delta^{-1}))$ where a is a lower bound on the amplitude and δ is the success probability. Applying Jackson's theorem in this situation would yield $a^{-1}\delta^{-1}$. Furthermore, Jackson's theorem does not always yield the best constant factors when used directly.

Fortunately, [LC17] develops a polynomial approximation specifically for applications like this that has excellent constant factor performance, and has the correct asymptotic scaling.

Theorem 3.21. ***Approximation of the sign function.*** *Let $I_j(x)$ be the j'th modified Bessel function of the first kind. Now, for an odd integer n and positive real k, let:*

$$p^{(k,n)}_{sign}(x) := \frac{2ke^{-\frac{k^2}{2}}}{\sqrt{\pi}}\left(I_0\left(\frac{k^2}{2}\right)\cdot x + \sum_{j=1}^{(n-1)/2} I_j\left(\frac{k^2}{2}\right)(-1)^j\left(\frac{T_{2j+1}(x)}{2j+1} - \frac{T_{2j-1}(x)}{2j-1}\right)\right) \tag{3.71}$$

This polynomial is odd and of degree n. It approximates sign(x), defined by sign$(x<0) = -1$ and sign$(x>0) = 1$. For any $\kappa \in [0,1]$, let:

$$\varepsilon^{(k,n)}_\kappa := \max_{[-1,-\kappa/2]\cup[\kappa/2,1]} |sign(x) - p^{(k,n)}_{sign}(x)| \tag{3.72}$$

be the error of the approximation on $[-1,1]$ except a region around 0 of width κ. As n,k increase, $\varepsilon^{(k,n)}_\kappa$ strictly decreases. Moreover, for any δ, if we select:

$$k := \frac{\sqrt{2}}{\kappa}\sqrt{\ln\left(\frac{8}{\pi\delta^2}\right)} \tag{3.73}$$

$$n := \lceil \sqrt{2\lceil \max((ke)^2/2, \log(4/\delta))\rceil \log(8/\delta)}\rceil \tag{3.74}$$

then $\varepsilon^{(k,n)}_\kappa \leq \delta$. Observe that $n \in O\left(\kappa^{-1}\log(\delta^{-1})\right)$.

Proof. This is a result in Appendix A of [LC17]. □

We could also just increase k, n until the approximation is good enough to get a tighter bound on the error, but since $I_j(x)$ is challenging to evaluate numerically at high precision, this might be pretty cumbersome.

The polynomial itself is not PQ- or FG-completable since it overshoots the interval $[-1, 1]$. However, it can be scaled down by a factor of $1 + \delta$ to make it FG-completable.

Now we briefly give some results that we will need for later chapters. Chapter 4 will require a polynomial that approximates a window function that indicates if the input is in an interval $[a, b]$.

Corollary 3.22. ***Window Polynomial.*** *For any $\eta > 0$ and any a, b with $-1 < a < b < 1$, there exists a FG-completable polynomial $w(x)$ such that that for all functions $f(x)$:*

$$\left| \int_{-1}^{1} f(x)w(x)dx - \int_{a}^{b} f(x)dx \right| \leq \eta f_{max} \tag{3.75}$$

Where:

$$f_{max} \geq \max_{p<q} \frac{1}{q-p} \int_{p}^{q} f(x)dx \tag{3.76}$$

The polynomial has degree $O(\eta^{-1}\log(\eta^{-1}))$.

Proof. By shifting, scaling, and adding together two copies of $p_{\text{sign}}^{(k,n)}(x)$ from Theorem 3.21, we can construct a polynomial that is $\leq \eta$ outside of $[a, b]$ and $\geq 1 - \eta$ inside $[a, b]$, except for two region of width $\eta/2$ near a and b. Since we require $\delta \sim \kappa \sim \eta$, the degree is $O\left(\kappa^{-1}\log(\delta^{-1})\right) = O(\eta^{-1}\log(\eta^{-1}))$. □

Next, Chapter 5 will require a polynomial that performs ‘amplification’: if p is a small probability, then $A(p) \approx 0$. If p is a large probability, then $A(p) \approx 1$.

Corollary 3.23. ***Quantum Amplifying Polynomial.*** *For any $0 < \eta, \delta < 1/2$ there exists an FG-completable polynomial $A_{\eta\to\delta}(x)$ of degree $M_{\eta\to\delta} \in O\left(\eta^{-1}\log(\delta^{-1})\right)$, satisfying:*

$$\text{if } 0 \leq x \leq 1 \text{ then } 0 \leq A_{\eta\to\delta}(x) \leq 1 \tag{3.77}$$

$$\text{if } 0 \leq x \leq \frac{1}{2} - \eta \text{ then } A_{\eta\to\delta}(x) \geq 1 - \delta \tag{3.78}$$

$$\text{if } \frac{1}{2} + \eta \leq x \leq 1 \text{ then } A_{\eta\to\delta}(x) \leq \delta. \tag{3.79}$$

Proof. This can be achieved by shifting and scaling $p_{\text{sign}}^{(k,n)}(x)$. □

Chapter 4

Estimation of Physical Quantities

This chapter is based on [Rall20]. Some techniques presented in [Rall20] were already presented in Chapter 3, and are not restated here.

A central goal of quantum algorithms is to aid in the study of large quantum systems. It is well established, for example, that quantum computers can simulate the dynamics of most Hamiltonians of interest [Berry&19]. Hamiltonian simulation algorithms, sometimes combined with the quantum Fourier transform, have led to quantum algorithms for some physical quantities, including correlation functions [Pedernales&14] and dynamical linear response functions [RC18]. Both of these examples are crucial for the understanding of phenomena in condensed matter physics like electron and neutron scattering [West75, Sears84], conductivity and magnetization [DiVentra08].

Recent work in Hamiltonian simulation has yielded algorithms with exponential improvements in accuracy [CKS15] over Trotterization and guarantee linear scaling with the simulation time [Berry&19]. The strategies employed by these works can be neatly encompassed in terms of 'block encodings' - a tool that allows quantum computers to represent non-unitary matrices. These block encodings can be built using linear combinations of unitaries (LCUs) [BCK15, CKS15] and manipulated using quantum singular value transformation [GSLW18]. In addition to providing new and better algorithms, block encodings provide an intuitive and powerful framework for performing linear algebra on a quantum computer.

In this work we use block encodings along with amplitude amplification [BHMT00, AR19, GGZW19] to construct quantum algorithms for some physical quantities: n-time correlation functions, the local and non-local density of states, and dynamical linear response functions. These algorithms are more versatile than previous works [Pedernales&14, RC18] in that they can compute more general versions of the functions with greater accuracy.

The local and non-local density of states and linear response functions are all functions of the energy $f(E)$. We are usually interested in obtaining the general shape of $f(E)$ over a range of energies, i.e. in obtaining a 'sketch' of $f(E)$. We show how to perform two sketching strategies from modern classical numerical condensed matter physics [WWAF05, Holzner&11, Fan&18]. First, we show how to compute integrals of $f(E)$ over a range of energies: $\int_{E_A}^{E_B} f(E)dE$. Second, we show how to compute the moments of a Chebyshev expansion of $f(E)$: briefly assuming $|E| \leq 1$ for ease of explanation, if $T_n(E)$ is the n'th Chebyshev polynomial of the first kind, then we show how to compute constants c_n such that

$$f(E) \approx \frac{1}{\pi\sqrt{1-E^2}} \cdot \sum_{n=0}^{N} c_n T_n(E). \tag{4.1}$$

This procedure is known as the kernel polynomial method [WWAF05] and is intuitively similar to sketching a function by computing the first few coefficients in its Fourier series. Very recent work [Rogg20] shows how similar methods can also perform point-estimates of the density of states by approximating a delta function with a polynomial close to a narrow Gaussian.

Algorithms that compute physical quantities often face barriers from complexity theory, since computing expectations of observables on ground states of Hamiltonians is QMA-complete [KKR04]. This remains true even when severe restrictions are placed on Hamiltonians [Bookatz12]. For this reason we employ strategies that sidestep these barriers. For correlation functions, we do not provide algorithms for preparing ground states or other states of interest, since the best algorithms for their preparation must use properties of the particular Hamiltonian in question. Evaluating the density of states at particular energies is #P-complete [BFS10], but sketching the density of states via integrals and Chebyshev expansions is in BQP.

The structure of our chapter is as follows. In section 4.1 we present a lemma for estimating the expectations of observables. In section 4.2 we employ these techniques to study n-time correlation functions. If we have a set of observables O_i and times $\{t_i\}$ we compute expectations of the form

$$\langle O_1(t_1)O_2(t_2)...\rangle \tag{4.2}$$

employing the Heisenberg picture. In section 4.3 we outline quantum singular value transformation and tools for computing Chebyshev moments and integrals over energy intervals. In section 4.4 we employ these techniques to compute the density of states and the local density of states. If H has eigenvalues $\{E_i\}$ and dimension D then the density of states is:

$$\rho(E) = \frac{1}{D}\sum_i \delta(E_i - E). \tag{4.3}$$

Furthermore, say H is a Hamiltonian describing a particle with some set of positions $\{\vec{r}\}$ and position eigenstates $\{|\vec{r}\rangle\}$. If the eigenvectors of H are $\{|\psi_i\rangle\}$, then the local density of states is:

$$\rho_{\vec{r}}(E) = \sum_i \delta(E_i - E) |\langle \psi_i | \vec{r} \rangle|^2 \tag{4.4}$$

Finally in section 4.5 we show how to sketch linear response functions of the form

$$A(E) = \langle B\delta(E - H + E_0)C \rangle \tag{4.5}$$

where E_0 is the ground state energy of H and B, C are some observables.

4.1 Estimating Expectations of Observables

The algorithms in this work construct block encodings of a desired A and estimate $\mathrm{Tr}(A\rho)$ for some given ρ. To do so we assume that there is a unitary that prepares a purification of ρ, which is any pure state such that ρ can be obtained by tracing out some ancillary space $\mathbb{C}^l$.

Definition 4.1. *Let ρ be a density operator on $\mathcal{H}$ and let $|\boldsymbol{0}\rangle$ be some easy-to-prepare state in $\mathcal{H}$. A unitary U_ρ on $\mathcal{H} \otimes \mathbb{C}^l$ for some l is an R-preparation-unitary of ρ if we have*

$$\rho = Tr_{\mathbb{C}^l}(|\rho\rangle\langle\rho|), \tag{4.6}$$

where $|\rho\rangle = U_\rho(|\mathbf{0}\rangle|0\rangle_l)$ and U_ρ is implementable using R elementary gates.

Often we are interested in correlation functions and linear response with respect to ground states or thermal states of some Hamiltonian. Depending on the situation performing state preparation can be an extremely difficult computational task, and the identification of specific practical situations where state preparation is easy is an area of active research [BK16]. We consider the problem of state preparation itself out of scope for this work, but aim to present our algorithms in an abstract manner to maximize their versatility and permit the leveraging of future results. We do point out the existence of the following generic tool for constructing thermal states.

Lemma 4.2. *Let H be a Hamiltonian on a D-dimensional Hilbert space with an α-scaled block encoding with complexity Q-. Then for any $\beta \geq 0$ there exists an R-preparation unitary for a state ε-close in trace distance to the thermal state $e^{-\beta H}/Z$ where $Z = Tr(e^{-\beta H})$ and:*

$$R \in O\left(Q\alpha \cdot \sqrt{\frac{D\beta}{Z}} \log\left(\sqrt{\frac{D}{Z}}\frac{1}{\varepsilon}\right)\right) \tag{4.7}$$

Proof. This is the main result of [CS16], combined with the newer Hamiltonian simulation results of [LC1610, LC1606] with corrections from [GSLW18]. Briefly, the strategy is to construct a block encoding of $e^{-\beta H/2}$ from e^{iHt} using the Hubbard-Stratonovich transformation, and multiply it onto a purification of the maximally mixed state using a strategy called robust oblivious amplitude amplification. □

We now show how to use amplitude estimation to estimate the expectation of block encoded observables.

Lemma 4.3. *If A is Hermitian and has an α-scaled block encoding with complexity Q and ρ has an R-preparation-unitary, then for every $\varepsilon, \delta > 0$ there exists an algorithm that produces an estimate ξ of $Tr(\rho A)$ such that*

$$|\xi - Tr(\rho A)| \leq \varepsilon \tag{4.8}$$

with probability at least $(1-\delta)$. The algorithm has circuit complexity $O\left((R+Q)\cdot \frac{\alpha}{\varepsilon}\log\frac{1}{\delta}\right)$.

Proof. The algorithm is as follows:

Algorithm: Observable Estimation

Let $\bar{A} = (I + A/\alpha)/2$, and let $U_{\bar{A}}$ be its 1-scaled block encoding with complexity $O(Q)$ which exists by Theorem 3.8. Let $U_{\bar{A}}$ have k ancillae as in Definition 3.6, and let l and $|\mathbf{0}\rangle$ be as in Definition 4.1. Let:

$$|\rho\rangle := U_\rho |\mathbf{0}\rangle |0\rangle_l \tag{4.9}$$

$$|\Psi\rangle := (U_{\bar{A}} \otimes I) |0\rangle_k |\rho\rangle \tag{4.10}$$

$$\Pi := |0\rangle_k \langle 0|_k \otimes |\rho\rangle \langle\rho| \tag{4.11}$$

Perform amplitude estimation to obtain an estimate ξ_0 of $|\Pi |\Psi\rangle |$ to precision $\varepsilon/(2\alpha)$ with probability at least $(1-\delta)$. Return $\xi := (2\xi_0 + 1)\alpha$.

For details on how to perform amplitude estimation we refer to recent results [AR19, GGZW19] that avoid using the quantum Fourier transform, which was required by the traditional method [BHMT00] from 2002. These results establish that $|\Pi |\Psi\rangle |$ can be estimated to additive error ε and probability at least $(1-\delta)$ using $O\left(\frac{1}{\varepsilon}\log\frac{1}{\delta}\right)$ applications of a Grover operator:

$$G := -(I - 2\Pi)(I - 2|\Psi\rangle\langle\Psi|) \tag{4.12}$$

This operator requires four uses of U_ρ and two uses of $U_{\bar{A}}$, so it has circuit complexity $O(R+Q)$. This completes the runtime analysis.

Amplitude estimation estimates:

$$|\Pi |\Psi\rangle | = | \langle 0|_k \langle\rho| (U_{\bar{A}} \otimes I) |0\rangle_k |\rho\rangle | \tag{4.13}$$

$$= | \langle\rho| (\bar{A} \otimes I) |\rho\rangle | \tag{4.14}$$

$$= |\mathrm{Tr}(|\rho\rangle \langle\rho| (\bar{A} \otimes I))| \tag{4.15}$$

$$= |\mathrm{Tr}\left(\mathrm{Tr}_{\mathbb{C}^l}(|\rho\rangle \langle\rho|)\bar{A}\right) | = |\mathrm{Tr}(\rho\bar{A})| \tag{4.16}$$

Since $\bar{A}$ has $|\bar{A}| \leq 1$ its eigenvalues lie in the range $[-1, 1]$, so $\bar{A}$ is positive semi-definite. Therefore ξ_0 approximates $|\mathrm{Tr}(\rho\bar{A})| = \mathrm{Tr}(\rho\bar{A}) = (1 + \mathrm{Tr}(\rho A)/\alpha)/2$ to error $\varepsilon/(2\alpha)$, so ξ approximates $\mathrm{Tr}(\rho A)$ to error ε as desired. □

In addition to providing a simple framework for manipulating observables on a quantum computer, block encodings are often the starting point for modern Hamiltonian simulation algorithms [BCK15, Berry&19]. Once a block encoding of a Hamiltonian H is constructed, we can apply functions to its eigenvalues using quantum singular value transformation discussed in section 4.3.

4.2 Correlation Functions

In this section we show how to estimate n-time correlation functions, improving on an algorithm presented in [Pedernales&14]. This algorithm does not require any new technical tools. We include it primarily to illustrate how simple it is to construct algorithms for complex quantities via block encodings. We also show how to estimate non-Hermitian block-encoded observables. Consider a system evolving under a time-independent Hamiltonian H. If O_i is some Hermitian operator then in the Heisenberg picture:

$$O_i(t_i) := e^{iHt_i} O_i e^{-iHt_i} \tag{4.17}$$

To prepare block encodings of observables in the Heisenberg picture we leverage a modern result in Hamiltonian simulation for time-independent Hamiltonians. For simplicity we focus on time-independent Hamiltonians but there also exist block encodings for time evolution under time-dependent Hamiltonians [Berry&19, KSB18, LW18].

Lemma 4.4. *Let H be a Hamiltonian on a D-dimensional Hilbert space with an α-scaled block encoding with complexity Q. Then for any $t, \varepsilon > 0$ there exists an ε-accurate block encoding with complexity $T(t, \varepsilon)$ of e^{iHt} where:*

$$T(t,\varepsilon) \in O\left(Q\alpha|t| + \frac{Q\log(1/\varepsilon)}{\log(e+\log(1/\varepsilon)/(\alpha|t|))}\right) \tag{4.18}$$

Proof. This result originated in [LC1606, LC1610], but it is cleanly re-stated with minor corrections as Corollary 60 of [GSLW18]. □

Using this result we can state and analyze the estimation algorithm.

Theorem 4.5. *Let:*

- *H be a Hamiltonian with an α-scaled block encoding with complexity Q,*
- *$O_1, ..., O_n$ be some observables with β_i-scaled block encodings with respective complexities R_i,*
- *$t_1, ..., t_n$ be some times,*
- *and ρ be a state with an S-preparation unitary.*

Then for every $\varepsilon, \delta > 0$ there exists an algorithm that produces estimate an estimate $\xi \in \mathbb{C}$ of $Tr(\rho \prod_i O_i(t_i))$ to additive precision ε in the real and imaginary parts with probability at *least* $(1-\delta)$. *It has circuit complexity* $O\left((S+W)\cdot\frac{\gamma}{\varepsilon}\log\frac{1}{\delta}\right)$ *where* $\gamma=\prod_i\beta_i$ *and*

$$W \in O\left(\sum_{j=1}^{n} R_j + \sum_{j=0}^{n} T\left(\tau_j, \frac{\varepsilon}{2(n+1)^2}\right)\right) \tag{4.19}$$

$$\subset O\left(\sum_{j=1}^{n} R_j + Q\alpha\sum_{j=0}^{n}|\tau_j| + Qn^2\log\left(\frac{n}{\varepsilon}\right)\right) \tag{4.20}$$

where $T(t,\varepsilon)$ *is defined in Lemma 5.14 and* $\tau_j = t_{j+1}-t_j$, *padding the list of times with* $t_0 = t_{n+1} = 0$.

Proof. The algorithm is as follows:

Algorithm: n-time correlation functions

Making use of $e^{-iHt_j}e^{iHt_{j+1}} = e^{iH(t_{j+1}-t_j)} = e^{iH\tau_j}$, we rewrite the product of observables as follows:

$$\prod_{j=1}^{n} O_j(t_j) = e^{iHt_1}O_1e^{iH(t_2-t_1)}...O_ne^{-iHt_n} \tag{4.21}$$

$$= e^{iH\tau_0}\prod_{j=1}^{n} O_je^{iH\tau_j} \tag{4.22}$$

Invoking Lemma 5.14 we obtain $\frac{\varepsilon}{2(n+1)^2}$-accurate block encodings of $e^{iH\tau_j}$, and we multiply them together with the block encodings of O_i using Theorem 3.8. We obtain a block encoding U_Γ with complexity W of an operator Γ that approximates $\prod_i O_i(t_i)$.

Observe that $U_\Gamma^\dagger$ is a block encoding of $\Gamma^\dagger$. This allows us to use Theorem 3.8 to construct γ-scaled block encodings with complexities W of the Hermitian and anti-Hermitian parts of Γ, as below. Then we invoke Lemma 4.3 with target accuracy $\varepsilon/2$ for each of the below to obtain ε-accurate estimates of the real and imaginary parts of $\text{Tr}\left(\rho \prod_i O_i(t_i)\right)$.

$$\Re(\xi) := \text{estimate of Tr}\left(\rho \cdot \frac{\Gamma + \Gamma^\dagger}{2}\right) \tag{4.23}$$

$$\Im(\xi) := \text{estimate of Tr}\left(\rho \cdot \frac{\Gamma - \Gamma^\dagger}{2i}\right) \tag{4.24}$$

Since the block encodings of $e^{iH\delta t_j}$ are 1-scaled, the only contribution to γ are the scalings of the O_i, so $\gamma = \prod_i \beta_i$. The runtime is dominated by the complexity W of the block encoding for Γ, which by Theorem 3.8 is clearly given by (4.19). To obtain (4.20) we loosely bound $1/\log(e + \log(1/\varepsilon)/(\alpha|t|)) \leq 1$ in (4.18). This looseness overestimates the runtime in situations where n is very large but the τ_j are very small.

It remains to show that Γ is $\varepsilon/2$-close in spectral norm to $\prod_i O_i(t_i)$, given that the block encodings of $e^{iH\tau_j}$ are $\frac{\varepsilon}{2(n+1)^2}$-accurate. From there the $\varepsilon/2$-closeness of the Hermitian and anti-Hermitian parts, and the ε-accuracy of the final estimates follow. In general, Lemma 54 of [GSLW18] gives an argument that if $|A - U| \leq \varepsilon_0$ and $|B - V| \leq \varepsilon_1$ then

$$|AB - UV| \leq \varepsilon_0 + \varepsilon_1 + 2\sqrt{\varepsilon_0 \varepsilon_1}. \tag{4.25}$$

Iterating this bound for a product of $\prod_{i=0}^n U_i$ where $|U_i - A_i| \leq \varepsilon_0$ we obtain by solving a recurrence relation:

$$\left| \prod_{i=0}^n U_i - \prod_{i=0}^n A_i \right| \leq (n+1)^2 \varepsilon_0. \tag{4.26}$$

Plugging in $\varepsilon_0 := \frac{\varepsilon}{2(n+1)^2}$ gives the desired upper bound of $\varepsilon/2$. □

This algorithm improves over [Pedernales&14] in several ways. First, [Pedernales&14] restricts to Pauli observables since they are unitary. Here O_i do not have to be unitary. Secondly, since we are using amplitude estimation to obtain ξ we obtain a quadratic speedup in the accuracy dependence. Finally, [Pedernales&14] restricts to Hamiltonians where exact Hamiltonian simulation can be achieved using circuit identities. Of course, for situations where these restrictions apply and the accuracy speedup can be sacrificed, their construction yields significantly smaller circuits which may be more amenable to near-term quantum computers.

4.3 Integrals and Chebyshev Moments of Functions of the Energy

In this section we introduce some tools we will require for our quantum algorithms for computing the density of states and linear response functions.

Say a Hermitian matrix A has an eigenvalue-eigenvector decomposition $A = \sum_i \lambda_i |\phi_i\rangle \langle\phi_i|$. Given a block encoding of A, quantum singular value transformation allows us to construct block encodings of $p(A) = \sum_i p(\lambda_i) |\phi_i\rangle \langle\phi_i|$, for polynomials $p(x)$. This requires $p(x)$ to be appropriately bounded, and the complexity of the encoding scales linearly in the degree of the polynomial. This method can also be generalized to non-Hermitian A with some caveats. Singular value transformation is an extremely powerful result, and is a culmination of a long line of research in quantum algorithms, presented in its full generality in [GSLW18].

Lemma 4.6. *Let A have a block encoding with complexity Q, and let $p(x)$ be a degree-d polynomial satisfying $|p(x)| \leq 1$ for $x \in [-1, 1]$. Then for every $\delta > 0$ there exists a $\frac{1}{2}$-scaled δ-accurate block encoding with complexity $O(Qd)$ of $p(A)$. A description of the circuit can be computed in time poly $\left(d, \log \frac{1}{\delta}\right)$.*

Proof. This strategy originated in [LC1606, LC1610] and is developed in [GSLW18] where it is formalized as Theorem 56. Calculating the circuit demands careful consideration of numerical precision. Recent work [Chao&20] describes an elegant strategy for dealing with this issue. $\square$

The expressions for density of states (4.3,4.4) and linear response (4.55) are both functions of the energy $f(E)$ roughly of the form:

$$f(E) := \sum_i \delta(E - E_i) \langle\psi_i| A |\psi_i\rangle \tag{4.27}$$

where $\{E_i\}$ and $\{|\psi_i\rangle\}$ are the eigenvalues and eigenvectors of the Hamiltonian and A is some Hermitian matrix. Rather than computing point-estimates of $f(E)$ we will be interested in computing integrals of $f(E)$ over a range $[a, b]$ as well as the moments of a Chebyshev expansion of $f(E)$. To obtain the scaling requirements of Theorem 3.18 we observe that an α-scaled block encoding of a Hamiltonian H guarantees that $|H/\alpha| \leq 1$. Rescaling $\bar{a} = a/\alpha$ and $\bar{b} = b/\alpha$, we construct a polynomial $w(x)$ that allows us to approximate integrals over the range $[\bar{a}, \bar{b}]$:

Theorem 4.7. *(Corollary 3.22 restated.) For every $\eta > 0$ and any $\bar{a}, \bar{b}$ with $-1 < \bar{a} < \bar{b} < 1$ there there exists a polynomial $w(x)$ such that for all $f(\alpha x)$ bounded by f_{max} (defined below in (4.29)):*

$$\left| \int_{-1}^{1} f(\alpha x) w(x) dx - \int_{\bar{a}}^{\bar{b}} f(\alpha x) dx \right| \leq \eta \tag{4.28}$$

The polynomial has degree $d \in O(\frac{f_{max}}{\eta} \ln \frac{f_{max}}{\eta})$ and satisfies the requirements of Theorem 3.14.

Our accuracy analysis requires a bound on $f(\alpha x)$, which is a bit subtle to define since $f(\alpha x)$ is a sum of many delta functions. However, we only ever perform integrals of

$f(\alpha x)$. Therefore when we say '$f(\alpha x)$ is bounded by $f_{\max}$' we mean that for all $\bar{c} < \bar{d}$:

$$\int_{\bar{c}}^{\bar{d}} f(\alpha x) dx \leq f_{\max} \cdot (\bar{d} - \bar{c}) \tag{4.29}$$

The polynomial $w(x)$ immediately yields a strategy for computing integrals since the value can be expressed as a trace inner product.

$$\int_a^b f(E) dE = \int_{\bar{a}}^{\bar{b}} f(\alpha x) \cdot \alpha dx \tag{4.30}$$

$$\approx \alpha \int_{-1}^{1} f(\alpha x) w(x) dx \tag{4.31}$$

$$= \alpha \int_{-1}^{1} \sum_i \delta(\alpha x - E_i) \langle \psi_i | A | \psi_i \rangle w(x) dx \tag{4.32}$$

$$= \mathrm{Tr} \left(A \sum_i \int_{-1}^{1} \delta(x - E_i/\alpha) w(x) dx \, |\psi_i\rangle \langle \psi_i| \right) \tag{4.33}$$

$$= \mathrm{Tr} \left(A \sum_i w(E_i/\alpha) \, |\psi_i\rangle \langle \psi_i| \right) \tag{4.34}$$

$$= \mathrm{Tr} \left(A w(H/\alpha) \right) \tag{4.35}$$

In step (4.33) we used the identity $\delta(\alpha x) = \delta(x)/\alpha$. This final expression can then be estimated using Lemma 4.3.

Next we briefly outline our strategy for sketching $f(E)$ using the kernel polynomial method [WWAF05]. A sketch $f^{\mathrm{KPM}}(E)$ is a linear combination of Chebyshev polynomials of the first kind $T_n(x)$ weighted by coefficients $\mu_n^f g_n$. The μ_n^f are the Chebychev moments of $f(E)$ and the g_n are $f(E)$-independent smoothing coefficients (see for example the proof of Jackson's theorem in [Rivlin69]). Since Chebyshev expansions are performed on the domain $[-1, 1]$ we calculate moments of $f(\alpha x)$ for $x \in [-1, 1]$.

$$\mu_n^f := \int_{-1}^{1} T_n(x) f(\alpha x) dx \tag{4.36}$$

$$f^{\text{KPM}}(\alpha x) := \frac{1}{\pi\sqrt{1-x^2}} \left(g_0 \mu_0^f + 2 \sum_{n=0}^{N} \mu_n^f g_n T_n(x) \right) \tag{4.37}$$

For this work we concern ourselves only with estimation of μ_n^f and defer to [WWAF05, Fan&18] for details on how to construct $f^{\text{KPM}}(E)$. A similar derivation to (4.30-4.35) yields the identity:

$$\mu_n^f := \int_{-1}^{1} T_n(x) f(\alpha x) dx = \text{Tr}\left(A T_n(H/\alpha)\right) \tag{4.38}$$

Conveniently, quantum singular value transformation is particularly simple for Chebyshev polynomials.

Lemma 4.8. *Let A have a block encoding with complexity Q. Then for every n there exists an block encoding with complexity $O(nQ)$ of $T_n(A)$.*

Proof. This is Lemma 9 of [GSLW18]. See also Example 3.12. □

Now we have all the technical tools to state the main algorithms.

4.4 Density of States

In this section we show how to sketch the density of states (DOS):

$$\rho(E) = \frac{1}{D} \sum_i \delta(E_i - E). \tag{4.39}$$

This is easily rewritten in the form in (4.27) by choosing $A = I/D$. Following (4.30-4.35) and (4.38) we obtain:

$$\int_a^b \rho(E)dE \approx \mathrm{Tr}\left(\frac{I}{D}w(H/\alpha)\right) \tag{4.40}$$

$$\mu_n^\rho = \mathrm{Tr}\left(\frac{I}{D}T_n(H/\alpha)\right) \tag{4.41}$$

This argument makes use of of Theorem 4.7 which requires a bound on $\rho(E)$. Observe that in the sense of (4.29), $\rho(\alpha x)$ is bounded by any upper bound on the dimension of the largest eigenspace of H which we call $\rho_{\max}$.

These quantities can be estimated by leveraging the fact that I/D has an $O(\log(D))$-preparation unitary.

Theorem 4.9. *Let H have an α-scaled block encoding with complexity Q and take any $\varepsilon, \delta > 0$. Then:*

1. *For any a, b such that $-\alpha < a < b < \alpha$ there exists a quantum algorithm that produces an estimate ξ of $\int_a^b \rho(E)dE$ with circuit complexity*

$$O\left(\left(Q \cdot \frac{\rho_{max}}{\varepsilon}\log\frac{\rho_{max}}{\varepsilon} + \log D\right)\cdot\frac{1}{\varepsilon}\log\frac{1}{\delta}\right) \tag{4.42}$$

   and $O(poly(\rho_{max}/\varepsilon))$ classical pre-processing, where ρ_{max} is some upper bound on the dimension of the largest eigenspace of H.

2. *For any n there exists a quantum algorithm that produces an estimate ζ of μ_n^ρ with circuit complexity*

$$O\left((Q\cdot n + \log D)\cdot\frac{1}{\varepsilon}\log\frac{1}{\delta}\right). \tag{4.43}$$

The estimates ξ and ζ have error ε with probability at least $(1-\delta)$.

Proof. Observe that a preparation unitary for I/D simply prepares a Bell state on $\mathcal{H} \otimes \mathcal{H}$, call it $|\mathrm{Bell}(\mathcal{H})\rangle$. If $\mathcal{H}$ is encoded as some subspace of a n-qubit system where $n = \lceil \log_2(D) \rceil$ then $|\mathrm{Bell}(\mathcal{H})\rangle$ can be obtained from $|\mathrm{Bell}(\mathbb{C}^{2^n})\rangle$ via amplitude amplification. This procedure can be made exact via the following standard trick involving an ancilla qubit. Observe that

$$\beta := \langle \mathrm{Bell}(\mathbb{C}^{2^n}) | \mathrm{Bell}(\mathcal{H}) \rangle = \sqrt{D/2^n} \tag{4.44}$$

is known exactly. If U satisfies

$$U |0^{2n}\rangle = |\mathrm{Bell}(\mathbb{C}^{2^n})\rangle \tag{4.45}$$

$$= \beta |\mathrm{Bell}(\mathcal{H})\rangle + \sqrt{1-\beta^2} |\phi_\perp\rangle \tag{4.46}$$

for some $|\phi_\perp\rangle \perp |\mathrm{Bell}(\mathbb{C}^{2^n})\rangle$ then define U' such that:

$$U' |0^{2n+1}\rangle = \gamma U |0^{2n}\rangle |0\rangle + \sqrt{1-\gamma^2} |0^{2n}\rangle |1\rangle \tag{4.47}$$

$$= \gamma\beta |\mathrm{Bell}(\mathcal{H})\rangle |0\rangle + \sqrt{1-(\gamma\beta)^2} |\psi_\perp\rangle \tag{4.48}$$

for some $|\phi_\perp\rangle \perp |\mathrm{Bell}(\mathbb{C}^{2^n})\rangle |0\rangle$ where γ is the largest number ≤ 1 such that

$$\sin((2k+1)\arcsin(\gamma\beta)) = 1 \tag{4.49}$$

has a solution where k is a positive integer. Then, if $\theta = \arcsin(\gamma\beta)$ and $\Pi_{\mathcal{H}}$ is a projection onto the $\mathcal{H} \otimes \mathrm{span}(|0\rangle\langle 0|)$ subspace of $\mathbb{C}^{2n+1}$, then we can define a Grover operator G that exactly prepares $|\mathrm{Bell}(\mathcal{H})\rangle$.

$$G = U'(I - 2|0^{2n+1}\rangle\langle 0^{2n+1}|)(U')^\dagger (I - 2\Pi_{\mathcal{H}}) \tag{4.50}$$

$$G^k |\mathrm{Bell}(\mathbb{C}^{2^n})\rangle = \sin((2k+1)\theta)) |\mathrm{Bell}(\mathcal{H})\rangle |0\rangle$$
$$+ \cos((2k+1)\theta) |\psi_\perp\rangle \tag{4.51}$$

$$= |\mathrm{Bell}(\mathcal{H})\rangle |0\rangle \tag{4.52}$$

Since $2^n < 2D$ we have $\beta \in \Omega(1)$ so $k \in O(1)$, so the circuit complexity is dominated by U, which can be constructed using n Hadamard gates and n CNOT gates. Thus the state I/D on a Hilbert space $\mathcal{H}$ encoded in $\mathbb{C}^n$ has an $O(\log(D))$-preparation-unitary.

The algorithm for estimating integrals is as follows:

Algorithm: Integral of the Density of States

1. Use Theorem 4.7 to construct the polynomial $w(x)$ with $\eta := \frac{\varepsilon}{3}$.
2. Use Theorem 3.18 to construct an $\frac{\varepsilon}{3}$-accurate $\frac{1}{2}$-scaled block encoding of $w(H/\alpha)$. Say that this is an exact $\frac{1}{2}$-scaled block encoding of $\tilde{w}(H/\alpha)$.
3. Use Lemma 4.3 to produce an $\frac{\varepsilon}{3}$-accurate estimate ξ of $\text{Tr}\left(\frac{I}{D} \cdot \tilde{w}(H/\alpha)\right)$ with probability at least $(1-\delta)$.

By the triangle inequality the total error is at most ε. The polynomial $w(x)$ has degree:

$$d \in O\left(\frac{\rho_{\max}}{\varepsilon} \log \frac{\rho_{\max}}{\varepsilon}\right) \tag{4.53}$$

The approximate block encoding of $\tilde{w}(H/\alpha)$ has circuit complexity $O(dQ)$ and the preparation unitary for I/D has circuit complexity $\log D$. Combining these with the number of samples required by Lemma 4.3 gives the overall complexity (4.42).

The algorithm for Chebyshev Moments is significantly simpler:

Algorithm: Chebyshev Moments of Density of States

1. Use Lemma 4.8 to construct a block encoding of $T_n(H/\alpha)$.
2. Use Lemma 4.3 to produce an ε-accurate estimate ζ of $\mathrm{Tr}\left(\frac{I}{D} \cdot T_n(H/\alpha)\right)$ with probability at least $(1-\delta)$.

Since the block encoding and state preparation are exact, the error stems entirely from the estimation procedure in Lemma 4.3. The circuit complexity from Lemma 4.8 is $O(nQ)$, so the overall complexity (4.43) also follows from Lemma 4.3. $\square$

Estimation of integrals of $\rho(E)$ benefit from knowledge of an upper bound $\rho_{\max}$. Indeed even in pathological cases where $H \propto I$ we have $\rho_{\max} = 1$, so the circuit complexity can never suffer from high densities of state. We argue that in practical situations prior information on H can be used to bound $\rho_{\max}$, thereby improving the complexity. For example, the DOS of quantum many body systems with local interactions is often close to a Gaussian due to the central limit theorem. In particular, [HMH04] discusses the DOS of a nearest-neighbor Hamiltonian acting on a spin chain. From their work on the transverse-field Ising model with n sites we can derive:

$$\rho_{\max} = \frac{C}{D}\binom{n}{n/2} \approx C\pi\sqrt{\frac{2}{n}}$$

for some constant C (see the discussion surrounding equation 30 in [HMH04]). Here $\rho_{\max}$ decreases with the number of sites.

Furthermore, exact degeneracy in a Hamiltonian is connected to the Hamiltonian's symmetries [CF16]. If there exists a degenerate subspace of dimension $D\rho_{\max}$ then any unitary transformations on that subspace must preserve the Hamiltonian. Thus, prior knowledge of the symmetries could be used to obtain a bound on $\rho_{\max}$. However, if only a subset of the symmetries is known then this only leads to a lower bound on the dimension of the largest eigenspace, which is not useful here.

Of course, the efficiency of the algorithm relies on the $1/D$ factor in our definition of $\rho(E)$. If we were interested in the actual number of states within an interval, the circuit complexity would scale with D (for fixed ε). This is to be expected since the number of states in the ground space of a Hamiltonian is #P-hard to compute exactly and NP-hard to estimate to within relative error [BFS10].

Next we consider the local density of states. Say we are working with a Hamiltonian describing a single particle in real space or some space with a notion of locality so that for every position $\vec{r}$ there is a state $|\psi(\vec{r})\rangle$ denoting the state with the particle at $\vec{r}$. Then local density of states (LDOS) at $\vec{r}$ is given by [WWAF05, DiVentra08, YLMV13]:

$$\rho_{\vec{r}}(E) = \sum_i \delta(E_i - E) |\, \langle \psi_i | \vec{r} \rangle \,|^2 \tag{4.54}$$

The algorithms for sketching the LDOS are a simple modification of the algorithms for DOS: instead of preparing a maximally mixed state we simply prepare $|\psi(\vec{r})\rangle$. Indeed if $|\psi(\vec{r})\rangle$ has an $O(R)$-preparation unitary, the new circuit complexities are the same as those in Theorem 4.9 but with $\log D$ replaced with R.

If H is a lattice Hamiltonian, e.g. a Fermi-Hubbard model, then the states $|\psi(\vec{r})\rangle$ are trivial to prepare since the Jordan-Wigner transformation that maps H to qubits preserves locality. For Hamiltonians describing a particle in real-space, the cost of preparing $|\psi(\vec{r})\rangle$ depends on the particular choice of basis functions, e.g. Hartree-Fock, used to encode H on the quantum computer.

Similarly to the DOS, estimation of LDOS can benefit from bounds on $\rho_{\max}$ and it remains true that even for pathological Hamiltonians like $H \propto I$ we have $\rho_{\max} \leq 1$. However, it no longer makes sense to bound $\rho_{\max}$ via a central limit theorem since there is only one particle involved.

4.5 Linear Response

In this section we show how to sketch correlation functions of the form:

$$A(E - E_0) = \langle B\delta(E - H)C\rangle \tag{4.55}$$

We shift the function by the ground state energy E_0 since we consider estimation of the ground state energy out of scope. This work improves on an quantum algorithm by [RC18] and is useful to compare to a classical algorithm based on matrix product states [Holzner&11] that also uses the kernel polynomial method.

Following a similar argument to (4.30-4.35) and (4.38), we connect the desired quantities to expectations of observables that can be represented by block encodings:

$$\int_a^b A(E - E_0)dE \approx \langle Bw(H/\alpha)C\rangle \tag{4.56}$$

$$\mu_n^A = \langle BT_n(H/\alpha)C\rangle \tag{4.57}$$

This naturally yields quantum algorithms quite similar to those presented in Theorem 4.9, just with some constants changed.

Theorem 4.10. *Let:*

- *H have an α-scaled block encoding with complexity Q,*
- *ρ have an R-preparation-unitary,*
- *B have β-scaled block encoding with complexity S_B and C have γ-scaled block encoding with complexity S_C.*

Then for any $\varepsilon, \delta > 0$:

1. *For any a, b such that $-\alpha < a < b < \alpha$ there exists a quantum algorithm that produces an estimate ξ of $\int_a^b A(E)dE$ with circuit complexity*

$$O\left((Qd + S_B + S_C + R) \cdot \frac{\beta\gamma}{\varepsilon} \log \frac{1}{\delta}\right) \tag{4.58}$$

and $O(poly(d))$ classical pre-processing, where ρ_{max} a bound on the dimension of the largest eigenspace and

$$d = O\left(\frac{\rho_{max}\beta\gamma}{\varepsilon} \log \frac{\rho_{max}\beta\gamma}{\varepsilon}\right). \tag{4.59}$$

2. *For any n there exists a quantum algorithm that produces an estimate ζ of μ_n^A with circuit complexity*

$$O\left((Qn + S_B + S_C + R) \cdot \frac{\beta\gamma}{\varepsilon}\right). \tag{4.60}$$

The estimates ξ and ζ have error ε with probability at least $(1-\delta)$ in their real and imaginary parts.

Proof. The algorithm for computing integrals is as follows:

Algorithm: Integrals of Linear Response Functions

1. Use Theorem 4.7 to construct the polynomial $w(x)$ with $\eta := \frac{\varepsilon}{3}$.
2. Use Theorem 3.18 to construct an $\frac{\varepsilon}{3}$-accurate $\frac{1}{2}$-scaled block encoding of $w(H/\alpha)$, and say it is an exact $\frac{1}{2}$-scaled block encoding of $\tilde{w}(H/\alpha)$.
3. Use Theorem 3.8 to construct a $\frac{1}{2}\beta\gamma$-scaled block encoding of $\Xi := B\tilde{w}(H/\alpha)C$.
4. Use Lemma 4.3 to produce an $\frac{\varepsilon}{3}$-accurate estimates of the real and imaginary parts of ξ with probability at least $(1-\delta)$, corresponding to the Hermitian and anti-Hermitian parts of Ξ as in (4.23,4.24).

The accuracy and complexity analysis is almost identical to that in Theorem 4.9, except for the fact that since $|B| \leq \beta$ and $|C| \leq \gamma$ we observe that $A(\alpha x)$ is bounded by $\rho_{\max}\beta\gamma$ when invoking Theorem 4.7. The algorithm for Chebyshev moments is as follows:

Algorithm: Chebyshev Moments of Linear Response Functions

1. Use Lemma 4.8 to construct a block encoding of $T_n(H/\alpha)$.

2. Use Theorem 3.8 to construct a $\beta\gamma$-scaled block encoding of $Z := BT_n(H/\alpha)C$.

3. Use Lemma 4.3 to produce an ε-accurate estimates of the real and imaginary parts of ζ with probability at least $(1-\delta)$, corresponding to the Hermitian and anti-Hermitian parts of Z as in (4.23,4.24).

$\square$

This technique is significantly more versatile than that of [RC18], which only treats the case when $B = C$ and when $\rho = |\psi_0\rangle \langle\psi_0|$. Their algorithm runs Hamiltonian simulation under B for a short amount of time to approximately prepare the state $B|\psi_0\rangle$, which is an additional source of error. Furthermore their work also does not capitalize on accuracy improvements from amplitude estimation.

The classical strategy [Holzner&11] relies on Matrix Product State (MPS) representations of states $|t_n\rangle = T_n(H/\alpha)C|\psi_0\rangle$. When accurate and efficient MPS representations of $|t_n\rangle$ exist (and $|\psi_0\rangle$ can be efficiently obtained - an assumption we also make), then quantum strategies are not needed. Indeed for many physical systems ground states obey area laws (see e.g. [AAG19]), which lends MPS strategies their power. Quantum strategies will still be useful for ground states with large amounts of entanglement where efficient classical representations do not exist.

4.6 Conclusion

We have demonstrated that block encodings provide a powerful framework for the matrix arithmetic on a quantum computer. This modern and versatile toolkit for quantum algorithms encompasses fundamental strategies such as amplitude amplification and estimation, and novel results in active areas like Hamiltonian simulation can be immediately leveraged due to its modularity. Furthermore, once all the necessary tools are assembled, algorithms based on block encodings are trivial to analyze. We believe that block encodings are the state-of-the-art technique for estimating physical quantities on a quantum computer. This claim should be further tested by attempting to quantize other numerical strategies in condensed matter physics.

Chapter 5

Measuring Observables

This chapter is based on [Rall21]. Some theorems were removed since they already appear in Chapter 3.

Phase estimation is one of the most widely used quantum subroutines, because it grants quantum computers two unique capabilities. First, it yields a quadratic speedup in the accuracy of Monte Carlo estimates [BHMT00, Mon15, HM18]. Achieving 'Heisenberg limited' accuracy scaling has numerous applications in physics, chemistry, machine learning, and finance [Wright&20, ESP20, Rall20, An&20]. Second, it allows quantum computers to diagonalize unitaries in a certain restricted sense: if $U = \sum_j e^{2\pi i \lambda_j} |\psi_j\rangle \langle\psi_j|$ then phase estimation performs the transformation

$$\sum_j \alpha_j |0^n\rangle |\psi_j\rangle \to \sum_j \alpha_j |\lambda_j\rangle |\psi_j\rangle \tag{5.1}$$

where $|\lambda_j\rangle$ is an n-bit estimate of λ_j. This access to spectral information enables quantum speedups for linear algebra [HHL08, CKS15], studying physical systems [Temme&09, YA11, Lemi&19, JKKA20], estimating partition functions [Mon15], and performing Bayesian inference [HW19, AHNTW20].

Phase estimation is also a very complicated algorithm. Textbook phase estimation [NC00] requires the Quantum Fourier Transform (QFT), a tool we commonly associate with the exponential speedup of Shor's algorithm [Shor95]. However, phase estimation only delivers a quadratic speedup for estimation and the exponential speedup for linear algebra can sidestep the QFT [CKS15, GSLW18]. When applied to energies, phase estimation also requires Hamiltonian simulation, a quantum subroutine that is an entire subject of study in its own right: it requires recent innovations to apply optimally in a black-box setting [BCK15, LC1606, LC1610, LC17, GSLW18] and optimal Hamiltonian simulation for specific systems is still being actively studied [SHC20, Cam20]. Furthermore, phase estimation demands median amplification to guarantee an accurate answer. This can be challenging to implement coherently on a quantum computer [HNS01, Klauck02, Beals&12], because it requires many ancillae and a quantum sorting network. The probability with which phase estimation gives the correct answer (sometimes referred to as the 'Fejer kernel' [Rogg20]) is not always high enough to be amplified, which must be dealt with either by rounding or adaptivity [AA16]. The conceptual complexity of textbook phase estimation and the resulting computational overhead motivates a search for alternatives.

Fortunately, a lot of simplification is possible if we allow for 'incoherent' algorithms, where incoherence manifests via a variety of assumptions. We could be allowed to measure the quantum state, either because we are given many copies of the input state, or because we have the ability to prepare it inexpensively. Alternatively, we can assume a type of adaptivity where quantum states wait patiently without decohering while a classical computer performs a computation on the side. This assumption sometimes lets us repair the input state after using it [MW05, Temme&09, Poulin&17]. But most importantly, incoherent phase estimation algorithms usually require that the input state is an eigenstate of the unitary or Hamiltonian in question.

If enough of these assumptions hold, then 'iterative phase estimation' [Kit95] removes an enormous amount of conceptual and computational overhead. The estimate can be extracted one bit at a time, thereby removing the QFT. The accuracy of each bit can be amplified individually via many classical samples, removing the need for the quantum sorting network. The awkward notion of an 'n-bit estimate' can be removed and replaced with a traditional additive-error estimate [AA16]. Iterative phase estimation has seen many refinements and has been applied to gate set tomography and ground state energy estimation [Higgins07, KLY15, LT21]. An incoherent iterative approach also permits direct simplification of amplitude estimation [AR19, GGZW19].

However, for some applications of phase estimation maintaining coherence remains crucial. The original strategy for quantum matrix inversion [HHL08], quantum Metropolis sampling [Temme&09, YA11, Lemi&19], and a protocol for partition function estimation [Mon15], thermal state preparation, and Bayesian inference [HW19, AHNTW20] all violate the assumptions above. The eigenvalues must be estimated while preserving the superposition, and there is no guarantee that the input state is an eigenstate. These applications of 'coherent' phase estimation motivate the main question of this chapter: does there exist a conceptually and computationally simpler algorithm for performing the transformation (5.1) while remaining coherent? That is: the input state is not necessarily an eigenstate, we are given exactly one copy, and there is no adaptive interaction with a classical computer.

In this chapter we answer this question in the affirmative. We present simplified coherent algorithms for phase estimation, energy estimation, and amplitude estimation. None of these algorithms require a QFT, and they can be made arbitrarily close to the ideal transformation without a quantum sorting network to compute a median. These algorithms are about 14x to 20x faster than traditional phase estimation in terms of their query complexity, a performance metric that neglects the fact that they also require fewer ancilla qubits.

However, we also observe that there are unavoidable barriers to estimation in superposition. Consider an estimation algorithm that outputs a superposition of two estimates $\hat{\lambda}_j^{(1)}$ and $\hat{\lambda}_j^{(2)}$, both of which could be pretty close to the true value λ_j:

$$\sum_j \alpha_j \left|0^n\right\rangle \left|\psi_j\right\rangle \to \sum_j \alpha_j \left(\xi_j \left|\lambda_j^{(1)}\right\rangle + \zeta_j \left|\lambda_j^{(2)}\right\rangle\right) \left|\psi_j\right\rangle \tag{5.2}$$

However, it is a well known fact [BBBV97] that uncomputation cannot work in such a situation (unless one of ξ_j or ζ_j is ≈ 0). Thus, any algorithm that actually makes use of the estimates must necessarily damage the input superposition and can no longer be considered coherent.

Even worse, we show that *any unitary quantum algorithm* for estimation must perform a map in the form (5.2), and there always exist some values of λ_j where the neither of the ξ_j and ζ_j are ≈ 0. Therefore, the only way to get an algorithm that performs the deterministic transformation (5.1) is to assume certain values of λ_j do not appear. We refer to this as a **rounding promise,** and show that if the rounding promise holds, then our algorithms perform the map (5.1). However, we also construct our algorithms in such a way that they give reasonable non-deterministic estimates as in (5.2) even when no rounding promise holds.

In the following we give a brief outline of the method we use to construct the algorithms. They strongly resemble iterative phase estimation [Kit95], which works roughly like this: the estimate is computed one bit at a time, starting with the least significant bit. A 'Hadamard-test' computes each bit with a decent success probability, which is then amplified to a high probability by taking the majority vote of many samples. This process could be made coherent naively by computing each sample into an ancilla, but this requires so many ancillae that any performance benefit over the QFT- and median-based approach is lost. The key idea is to amplify without using any new ancillae.

To manipulate the probabilities of the Hadamard-test, we use 'block encodings'. block encodings permit quantum computers to manipulate non-unitary matrices. In our case, the matrices' eigenvalues encode our probabilities. A unitary matrix U_A is a block encoding of A if A is in the top-left corner:

$$U_A = \begin{bmatrix} A & \cdot \\ \cdot & \cdot \end{bmatrix} \tag{5.3}$$

Block-encoded matrices can be manipulated in two ways. First, *linear combinations of unitaries* [BCK15, CKS15] allow us to build block encodings $\alpha A + \beta B$ given block encodings of A and B. Linear combinations of unitaries allow us to make block encodings of Hamiltonians presented as a sum of local terms, covering most practical applications. Second, *singular value transformation* [LC1610, LC1606, GSLW18] lets us apply certain polynomials $p(x)$ to the singular values of a block-encoded matrix A, using $\deg(p)$ many queries to controlled-U_A and its inverse.

Together, these two techniques permit the unification of many quantum algorithms into a single framework. For an accessible introduction to these methods, along with a comprehensive review of the most modern versions of these algorithms we refer to [MRTC21]. This work also presents the independent discovery of an algorithm very similar to our improved method of phase estimation, which is also sketched in [Chuang20].

To obtain the k'th bit ($0 \leq k < n-1$), iterative phase estimation performs a

Hadamard-test on $U^{2^{n-k-1}}$, where $U = \sum_j e^{2\pi i \lambda_j} |\psi_j\rangle \langle\psi_j|$. The outcome of the test is a coin toss that is heads with probability $\cos^2(2^{n-k-1}\pi\lambda_i)$. We observe that the quantum circuit for the Hadamard-test resembles a linear combination of unitaries. Therefore, we can construct a block encoding of a matrix whose eigenvalues encode the Hadamard-test probability for each eigenvector of U.

Iterative phase estimation then proceeds to perform the Hadamard-test several times and takes the majority vote. We observe that the probability that the majority vote is 1 is a polynomial in the Hadamard-test probability p:

$$\Pr[\text{majority vote is 1}] = \sum_{k=\lceil M/2 \rceil}^{M} \binom{M}{k} p^k (1-p)^{M-k} \tag{5.4}$$

We can therefore apply such an 'amplifying polynomial' [Diak09] directly to the block encoding using singular value transformation, which requires no new ancillae. In fact, using techniques from [LC17], we can construct a polynomial that performs the same task with much smaller degree.

Now we have amplified the eigenvalues of the block encoding to be close to 0 or 1, so we have a projector. To extract the 0/1-eigenvalue into a qubit we use the following novel technical tool:

Theorem. ***Block-measurement.*** *Say we have an approximate block encoding of a projector* Π. *Then there exists a quantum channel that approximately implements the map:*

$$|0\rangle \otimes |\psi\rangle \to |1\rangle \otimes \Pi\,|\psi\rangle + |0\rangle \otimes (I - \Pi)\,|\psi\rangle \tag{5.5}$$

The channel is based on uncomputation [BBBV97], but the error analysis also features an interesting trick to deal with the case where the uncomputation fails. This theorem may find applications elsewhere.

Repeating the above procedure for each bit while carefully adjusting the phases at every step yields an algorithm that performs coherent phase estimation with no ancillae required. To estimate energies, we can construct a block encoding with eigenvalues $\cos^2(2^{n-k-1}\pi\lambda_j)$ directly using the Jacobi-Anger expansion [GSLW18] rather than going through Hamiltonian simulation, which boosts the performance further, although now the algorithm does require ancillae. Finally, to perform coherent amplitude estimation, we construct a block encoding of a 1x1 matrix containing the amplitude to be estimated and then invoke energy estimation.

A key research question of this chapter is: Do these algorithms perform better than traditional phase estimation in practice? An asymptotic analysis is not sufficient to answer this question. Instead, we carefully bound the query complexity and failure probability of all algorithms involved using the diamond norm and carefully select constants to maximize the performance. Subsequently, we perform a numerical analysis of the query complexity. We find that we improve the query complexity of phase estimation by a factor of about 13x, and the query complexity of energy estimation is improved by about 20x. Our amplitude estimation algorithm inherits the speedup of the energy estimation algorithm.

This chapter is structured as follows. In section 5.1 we carefully define the problem of coherent estimation and establish the notion of a rounding promise. Then we analyze the textbook method. In section 5.2 we present our novel algorithms for phase and energy estimation. In section 5.3 we discuss a numerical analysis of the query complexities of the algorithms. Then, in section 5.4 give a proof of the block-measurement theorem above and then show how to use energy estimation to perform non-destructive amplitude estimation in section 5.5.

5.1 Preliminaries

In this section we formally define the estimation tasks we want to solve (Definition 5.2). This requires setting up the notion of a 'rounding promise' (Definition 5.1) which highlights an inherent complication with coherent estimation that is not present in the incoherent case. We argue that this complication is unavoidable, and should be taken into account when coherent estimation is performed in practice. Next, we set up some simple technical tools for dealing with the error analysis of uncomputation (Lemmas 5.3,5.4). These will be used several times in this chapter. Finally, we give a precise description and analysis of 'textbook' phase estimation (Proposition 5.5). Thus, this section clearly defines the problem and the previous state-of-the-art which we improve.

Our chapter presents algorithms for phase, energy, and amplitude estimation. Amplitude estimation follows as a relativity simple corollary of energy estimation so we will only be talking about the prior two until Section 5.4. To talk about both phases and energies at once, we standardize input unitaries and Hamiltonians into the following form:

$$U = \sum_j e^{2\pi i \lambda_j} |\psi_j\rangle \langle\psi_j| \qquad\qquad H = \sum_j \lambda_j |\psi_j\rangle \langle\psi_j| \tag{5.6}$$

We refer to the $\{\lambda_j\}$ as the 'eigenvalues', and assume they live in the range $[0, 1)$. While any unitary can be put into this form, the form places the constraint $0 \preceq H \prec I$ onto the Hamiltonian.

Our goal is to compute $|\lambda_j\rangle$ an 'n-bit estimate of λ_j'. In this chapter, we take this to be an n-qubit register containing a binary encoding of the number floor$(2^n\lambda_j)$. Since $\lambda_j \in [0, 1)$ we are guaranteed that floor$(2^n\lambda_j)$ is an integer $\in \{0, ..., 2^n - 1\}$ and thus has an n-bit encoding.

Consider phase estimation using a quantum circuit composed of elementary unitaries and controlled-$e^{2\pi i\lambda_j}$. Following an argument related to the polynomial method, we see that the resulting state must be of the form:

$$\sum_{x\in\{0,1\}^n}\sum_{y}\alpha_{x,y}(e^{2\pi i\lambda_j})\,|x\rangle\,|\text{garbage}_{x,y}\rangle \tag{5.7}$$

where $\alpha_{x,y}(e^{2\pi i\lambda_j})$ is some polynomial of $e^{2\pi i\lambda_j}$. We would like $|x\rangle$ to encode floor$(2^n\lambda_j)$, meaning that $\alpha_{x,y}(e^{2\pi i\lambda_j}) = 1$ if $x = \text{floor}(2^n\lambda_j)$ and $\alpha_{x,y}(e^{2\pi i\lambda_j}) = 0$ otherwise. This is impossible: $\alpha_{x,y}(e^{2\pi i\lambda_j})$ is a continuous function of λ_j, but the desired amplitude indicating $x = \text{floor}(2^n\lambda_j)$ is discontinuous. For energy estimation a similar argument applies, just that the amplitudes are of the form $\alpha_{x,y}(\lambda_j)$. This argument even holds in the approximate case when we demand that $\alpha_{x,y} \leq \delta$ or $\geq 1-\delta$ for some small δ - the discontinuity is present regardless.

To some extent, this issue stems from the awkwardness of the notion of an 'n-bit estimate' since it requires rounding or flooring, a discontinuous operation, when only continuous manipulation of amplitudes is possible. A more comfortable notion is that of an additive-error estimate, used by [AA16, KP16] in their estimation algorithms.

However, the primary application of our algorithms is a situation where this simplification is not possible: Szegedy walks based on Hamiltonian eigenspaces [YA11, Lemi&19, JKKA20, WT21]. The original quantum Metropolis algorithm [Temme&09] implements a random walk over Hamiltonian eigenspaces, where the superposition is measured at every step and it is demonstrated that an additive error estimate is sufficient. But in order to harness a quadratic speedup due to quantum walks [Sze04], the measurement of energies must be made completely coherent, which is not achieved by an additive-error estimate unless the error is smaller than the gap between any eigenvalues. Therefore, we retain the notion of an 'n-bit estimate' in this chapter. We could always fall back to an estimate with additive error $\varepsilon/2$ by computing $n = \text{ceil}(\log_2(\varepsilon^{-1}))$ bits of accuracy.

However, the above argument still applies for additive-error estimation: the amplitude of any given estimate $|\hat{\lambda}\rangle$ is a continuous function of the eigenvalue λ_j. This means that the amplitude cannot be 0 or 1 everywhere, there must exist points where it crosses intermediate values in order to interpolate in between the two. In both the 'n-bit estimate' case and the additive-error case, this causes a problem for coherent quantum algorithms since the estimate cannot always be uncomputed. For some eigenvalues λ_j, the algorithm will yield an output state of the form $\alpha |\hat{\lambda}\rangle + \beta |\hat{\lambda}'\rangle$. Even if $\hat{\lambda}, \hat{\lambda}'$ are good estimates of λ_j, the uncompute trick [BBBV97] cannot be used for such an output state. Thus, the damage to the input superposition over $|\lambda_j\rangle$ is irreparable.

The only way to deal with this issue is to assume that the λ_j do not take certain values. Then the amplitudes can interpolate between 0 and 1 at those points. Observe that the discontinuities in the function $\text{floor}(2^n \lambda_j)$ occur at multiples of $1/2^n$. A 'rounding promise' simply disallows eigenvalues near these regions.

Definition 5.1. *Let $n \in \mathbb{Z}^+$ and $\alpha \in (0,1)$. A hermitian matrix H satisfies an (n,α)-* ***rounding promise*** *if it has an eigendecomposition $H = \sum_j \lambda_j |\psi_j\rangle \langle\psi_j|$, all the eigenvalues λ_j satisfy $0 \leq \lambda_j < 1$, and for all $x \in \{0, ..., 2^n\}$:*

$$\lambda_j \notin \left[\frac{x}{2^n}, \frac{x}{2^n} + \frac{\alpha}{2^n}\right] \tag{5.8}$$

Similarly, a unitary matrix U satisfies an (n,α)-rounding promise if it can be written *as* $U = \sum_j e^{2\pi i \lambda_j} |\psi_j\rangle \langle\psi_j|$ *and the phases* λ_j *satisfy the same assumptions.*

We have $\lambda_j \in [0,1)$, and we have disallowed λ_j from certain sub-intervals of $[0,1)$. The definition above is chosen such that the total length of these disallowed subintervals is α, regardless of the value of n. We have essentially cut out an α-fraction of the allowed eigenvalues.

Guaranteeing a rounding promise demands an enormous amount of knowledge about the eigenvalues. We do not expect many Hamiltonians or unitaries in practice to actually provably satisfy such a promise. However, we expect that the estimation algorithms discussed in this chapter will still perform pretty well even if they do not satisfy such a promise. One can increase the chances of success by setting α to be very small, so that the vast majority of the eigenvalues do not fall into a disallowed region. Then, if the input state is close to a uniform distribution over the eigenstates, then one can be fairly confident that only an α fraction of the eigenvalues in the support will be disallowed. The need for a rounding promise can be also sidestepped entirely by avoiding the use of energy estimation as a subroutine and approaching the desired problem directly. This has been accomplished for ground state finding [LT20].

The rounding promise is not just a requirement of our work in particular - it is a requirement for any coherent phase estimation protocol. The fact that coherent phase estimation does not work for certain phases is largely disregarded in the literature: for example, works like [KP16] neglect this issue entirely. However, there are some works that have observed this problem and attempt to mitigate it. Such methods are called 'consistent phase estimation' [TaShma13] since the error of their estimate is supposedly independent of the phase being estimated, thus allowing amplification to make the amplitudes always close to 0 or 1. We claim that all of these attempts fail, and furthermore that achieving coherent phase estimation without some kind of promise is impossible in principle. This is due to the polynomial method argument above: any coherent quantum algorithm's output state's amplitudes must be a continuous function of the phase, and continuous functions that are sometimes ≈ 0 and sometimes ≈ 1 must somewhere have an intermediate value. Recall that uncomputation only works when the amplitudes are close to 0 or 1. The error depends on if the phase is close to this transition point or not, so the error must depend on the phase. This issue was not taken into account by [Ambainis10], [KP17], and [KLLP18], since all of these either implicitly or explicitly state that there is an algorithm that approximately performs $|\psi_i\rangle |0^n\rangle \to |\psi_i\rangle |\lambda_i\rangle$ in superposition while λ_i is a deterministic computational basis state[1]. A more promising approach is detailed in [TaShma13], which, crudely speaking, shifts the transition points by a classically chosen random amount. Now whether or not a phase is close to a transition point is independent of the phase itself, making amplification possible. [Ambainis10] describes a similar idea, calling it 'unique-answer' eigenvalue estimation. However, we claim that this only works for a single phase. If we consider, for example, a unitary whose phases are uniformly distributed in $[0, 1)$ at a sufficiently high density, then there will be a phase near a transition point for any choice of random shift. The only way to avoid the rounding promise is to sacrifice coherence and measure the output state.

All of the algorithms in this chapter, including textbook phase estimation, achieve an asymptotic runtime of $O(2^n \alpha^{-1} \log(\delta^{-1}))$, where δ is the error in diamond norm. Before we move on to the formal definition of the estimation task, we informally argue that the

[1][Ambainis10] makes use of such a map in Algorithm 4. [KP17] states this as Theorem II.2. [KLLP18] sketches but does not analyze a protocol for this after Claim 4.5.

α^{-1} dependence is optimal, via a reduction to approximate counting. We are given N items, K of which are marked. Following the standard method for approximate counting [BHMT00], we construct a Grover unitary whose phases λ_j encode $\arcsin(\sqrt{K/N})$. Given a $(1, \alpha)$-rounding promise, computing floor$(2^1\lambda_j)$ amounts to deciding if $\lambda_j \leq \frac{1-\alpha/2}{2}$ or $\lambda_j \geq \frac{1+\alpha/2}{2}$ given that one of these is the case. By shifting the λ_j around appropriately we can thus decide if $K \geq (1/2 + C\alpha)N$ or $K \leq (1/2 - C\alpha)N$, for some constant C obtained by linearising $\arcsin(\sqrt{K/N})$. We have achieved approximate counting with a promise gap $\sim \alpha$. Thus the $\Omega(\alpha^{-1})$ lower bound on approximate counting [NW98] implies our runtime must be $\Omega(\alpha^{-1})$.

Equipped with the notion of a rounding promise, we can define our estimation tasks. Many algorithms in this chapter produce some kind of garbage, which can be dealt with the uncompute trick [BBBV97]. Rather than repeat the analysis of uncomputation in every single proof, we present a modular framework where we can deal with uncomputation separately. Furthermore, some applications may require computing some function of the final estimate, resulting in more garbage which also needs to be uncomputed. Rather than baking the uncomputation into each algorithm, it is thus more efficient to leave the decision of when to uncompute to the user of the subroutine.

Definition 5.2. *A **phase estimator** is a protocol that, given some $n \in \mathbb{Z}^+$ and $\alpha \in (0,1)$, and any error target $\delta > 0$, produces a quantum circuit involving controlled U and $U^\dagger$ calls to some unitary U. If $U = \sum_j e^{2\pi i \lambda_j} |\psi_j\rangle \langle\psi_j|$ satisfies an (n, α)-rounding promise then this circuit implements a quantum channel that is δ-close in diamond norm to the map:*

$$|0^n\rangle |\psi_j\rangle \rightarrow |\textit{floor}(\lambda_j 2^n)\rangle |\psi_j\rangle \tag{5.9}$$

Similarly, an ***energy estimator*** *is such a protocol that instead involves such calls to U_H which is a block encoding (see Definition 3.6) of a Hamiltonian $H = \sum_j \lambda_j |\psi_j\rangle \langle\psi_j|$ that satisfies an (n, α)-rounding promise.*

The ***query complexity of an estimator*** *is the number of calls to U or U_H in the resulting circuit, as a function of n, α, δ.*

*An estimator is said to '**have garbage**' or '**have** m **qubits of garbage**' if it is instead close to a map that produces some j-dependent m-qubit garbage state in another register:*

$$|0^n\rangle |0...0\rangle |\psi_j\rangle \to |\mathit{floor}(\lambda_j 2^n)\rangle |\mathit{garbage}_j\rangle |\psi_j\rangle \tag{5.10}$$

(Note that the quantum circuit can allocate and discard ancillae, but that does not count as garbage.)

*An estimator is said to '**have phases**' if the map introduces a j-dependent phase φ_j:*

$$|0^n\rangle |\psi_j\rangle \to e^{i\varphi_j} |\mathit{floor}(\lambda_j 2^n)\rangle |\psi_j\rangle \tag{5.11}$$

If an estimator is both 'with phases' and 'with garbage', then we can just absorb the $e^{i\varphi_j}$ into $|\text{garbage}_j\rangle$ so the 'with phases' is technically redundant. However, in our framework it makes more sense to treat phases and garbage independently since some of our algorithms are just 'with phases'.

The reason we measure errors in diamond norm has to do with uncomputation of approximate computations with garbage. Consider for example a transformation V acting on an answer register and a garbage register:

$$V|0\rangle|0...0\rangle = \sqrt{1-\varepsilon}\,|0\rangle|\text{garbage}_0\rangle + \sqrt{\varepsilon}\,|1\rangle|\text{garbage}_1\rangle \tag{5.12}$$

for some small nonzero ε. We copy the answer register into the output register:

$$\to \sqrt{1-\varepsilon}\,|0\rangle \otimes |0\rangle|\text{garbage}_0\rangle + \sqrt{\varepsilon}\,|1\rangle \otimes |1\rangle|\text{garbage}_1\rangle \tag{5.13}$$

and then we project the answer and garbage registers onto $V|0\rangle|0...0\rangle$:

$$\to \sqrt{1-\varepsilon}\,|0\rangle \cdot \sqrt{1-\varepsilon} + \sqrt{\varepsilon}\,|1\rangle \cdot \sqrt{\varepsilon} \tag{5.14}$$

The resulting state $(1-\varepsilon)|0\rangle + \varepsilon|1\rangle$ is not normalized, meaning that the projection $V|0\rangle|0...0\rangle$ succeeds with some probability < 1. Thus the uncomputed registers are not always returned to the $|0\rangle|0...0\rangle$ state.

At this point our options are either to postselect these registers to $|0\rangle|0...0\rangle$ or to discard them. Postselection improves the accuracy, but also implies that the algorithms do not always succeed. Since many applications of coherent energy estimation demand repeating this operation many times, it is important that the algorithm always succeeds. Thus, we need to discard qubits, so we need to talk about quantum channels. Therefore, the diamond norm is the appropriate choice for an error metric. Recall that the diamond norm is defined in terms of the trace norm [AKN98]:

$$|\Lambda|_\diamond := \sup_\rho |(\Lambda \otimes \mathcal{I})(\rho)|_1 \tag{5.15}$$

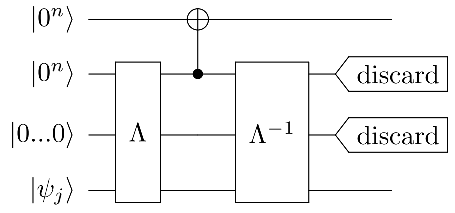
where $\mathcal{I}$ is the identity channel for some Hilbert space of higher dimension than Λ. The following analysis shows how to remove phases and garbage from an estimator, even in the approximate case.

Lemma 5.3. ***Getting rid of phases and garbage.*** *Given a phase/energy estimator with phases and/or garbage with query complexity $Q(n, \alpha, \delta)$ that is unitary, we can construct a phase/energy estimator without phases and without garbage with query complexity $2Q(n, \alpha, \delta/2)$.*

Proof. Without loss of generality, we assume we are given a quantum channel Λ that implements something close in diamond norm to the map:

$$|0^n\rangle |0...0\rangle |\psi_j\rangle \to e^{i\varphi_j} |\text{floor}(\lambda_j 2^n)\rangle |\text{garbage}_j\rangle |\psi_j\rangle \tag{5.16}$$

If the channel is actually without phases then we can just set $\varphi_j = 0$, and if it is actually without garbage then the following calculation will proceed without problems. Our strategy is just to use the uncompute trick, and then to discard the uncomputed ancillae. This requires us to implement Λ^{-1}, so we required that Λ be implementable by a unitary.



(5.17)

First, we consider the ideal case when Λ implements the map (5.16) exactly. If we use $|\psi_j\rangle$ as input and we stop the circuit before the discards, we produce a state $|\text{ideal}_j\rangle$:

$$|0^n\rangle |0^n\rangle |0...0\rangle |\psi_j\rangle \to e^{i\varphi_j} |0^n\rangle |\text{floor}(\lambda_j 2^n)\rangle |\text{garbage}_j\rangle |\psi_j\rangle \tag{5.18}$$

$$\to e^{i\varphi_j} |\text{floor}(\lambda_j 2^n)\rangle |\text{floor}(\lambda_j 2^n)\rangle |\text{garbage}_j\rangle |\psi_j\rangle \tag{5.19}$$

$$\to |\text{floor}(\lambda_j 2^n)\rangle |0^n\rangle |0...0\rangle |\psi_j\rangle =: |\text{ideal}_j\rangle \tag{5.20}$$

Let $\rho_{i,j}^{\text{ideal}} := |\text{ideal}_i\rangle \langle \text{ideal}_j|$, so that we can write down the ideal channel:

$$\Gamma_{\text{ideal}}(\sigma) := \sum_{i,j} \langle \psi_i| \sigma |\psi_j\rangle \cdot \rho_{i,j}^{\text{ideal}} \tag{5.21}$$

If we apply Λ with error $\delta/2$ instead we obtain the actual channel Γ such that:

$$\sup_{\sigma} |(\Gamma \otimes \mathcal{I})(\sigma) - (\Gamma_{\text{ideal}} \otimes \mathcal{I})(\sigma)|_1 \leq \delta \tag{5.22}$$

If A refers to the subsystems that start (and approximately end) in the states $|0^n\rangle |0...0\rangle$ then the final output state satisfies:

$$\sup_{\sigma} |\text{Tr}_A ((\Gamma \otimes \mathcal{I})(\sigma)) - \text{Tr}_A ((\Gamma_{\text{ideal}} \otimes \mathcal{I})(\sigma))|_1 \tag{5.23}$$

$$\leq \sup_{\sigma} |\text{Tr}_A ((\Gamma \otimes \mathcal{I})(\sigma) - (\Gamma_{\text{ideal}} \otimes \mathcal{I})(\sigma))|_1 \tag{5.24}$$

$$\leq \sup_{\sigma} |(\Gamma \otimes \mathcal{I})(\sigma) - (\Gamma_{\text{ideal}} \otimes \mathcal{I})(\sigma)|_1 \leq \delta \tag{5.25}$$

where we have used the fact that for all ρ and subsystems A we have $|\text{Tr}_A(\rho)|_1 \leq |\rho|_1$. Upon plugging in $\sigma = |\psi_j\rangle \langle \psi_j|$ we can see that tracing out the middle register after applying Γ_{ideal} implements the map

$$|0^n\rangle |\psi_j\rangle \to |\text{floor}(\lambda_j 2^n)\rangle |\psi_j\rangle \tag{5.26}$$

so therefore the circuit in (5.17) is an estimator without phases and garbage as desired.

The proof is complete, up to the fact that $|\text{Tr}_A(\rho)|_1 \leq |\rho|_1$ for all ρ and subsystems A. Write ρ in terms of its eigendecomposition $\rho = \sum_i \lambda_i |\phi_i\rangle \langle \phi_i|$, and let $\rho_i := \text{Tr}_A(|\phi_i\rangle \langle \phi_i|)$:

$$|\text{Tr}_A(\rho)|_1 = \left|\text{Tr}_A \left(\sum_i \lambda_i |\phi_i\rangle \langle \phi_i| \right)\right|_1 = \left| \sum_i \lambda_i \rho_i \right|_1 \leq \sum_i |\lambda_i| \cdot |\rho_i|_1 = |\rho|_1 \tag{5.27}$$

□

This establishes that uncomputation works in the approximate case as expected. While we are reasoning about the diamond norm, we also present the following technical tool which will come in handy several times. In particular, we will need it for our analysis of textbook phase estimation.

Lemma 5.4. ***Diamond norm from spectral norm.*** *Say U, V are unitary matrices satisfying $|U - V| \leq \delta$. Then the channels $\Gamma_U(\rho) := U\rho U^\dagger$ and $\Gamma_V(\rho) := V\rho V^\dagger$ satisfy $|\Gamma_U - \Gamma_V|_\diamond \leq 2\delta$.*

Proof. We have that:

$$|\Gamma_U - \Gamma_V|_\diamond := \sup_\rho |(\Gamma_U \otimes \mathcal{I})(\rho) - (\Gamma_V \otimes \mathcal{I})(\rho)|_1 \tag{5.28}$$

$$= \sup_\rho \left|(U \otimes I)\rho(U \otimes I)^\dagger - (V \otimes I)\rho(V \otimes I)^\dagger\right|_1 \tag{5.29}$$

where $\mathcal{I}$ was the identity channel on some subsystem of dimension larger than that of U, V, and $|M|_1$ is the sum of the magnitudes of the singular values of M.

Let $\bar{U} = U \otimes I$ and $\bar{V} = V \otimes I$. Then:

$$\left|\bar{U} - \bar{V}\right| = |U \otimes I - V \otimes I| = |(U - V) \otimes I| = |U - V| \leq \delta \tag{5.30}$$

If we let $\bar{E} := \bar{U} - \bar{V}$ we can proceed with the upper bound:

$$|\Gamma_U - \Gamma_V|_\diamond = \sup_\rho \left|\bar{U}\rho\bar{U}^\dagger - \bar{V}\rho\bar{V}^\dagger\right|_1 \tag{5.31}$$

$$= \sup_\rho \left|\bar{U}\rho\bar{U}^\dagger - \bar{V}\rho\bar{V}^\dagger + \left(\bar{U}\rho V^\dagger - U\rho V^\dagger\right)\right|_1 \tag{5.32}$$

$$= \sup_\rho \left|\left(\bar{U}\rho\bar{U}^\dagger - U\rho V^\dagger\right) - \left(\bar{V}\rho\bar{V}^\dagger - \bar{U}\rho V^\dagger\right)\right|_1 \tag{5.33}$$

$$= \sup_\rho \left|\bar{U}\rho(\bar{U} - \bar{V})^\dagger + (\bar{U} - \bar{V})\rho\bar{V}^\dagger\right|_1 \tag{5.34}$$

$$\leq \delta + \delta \tag{5.35}$$

□

Now we have all the tools required to analyze the textbook algorithm [NC00]. This algorithm combines several applications of controlled-U with an inverse QFT to obtain the correct estimate with a decent probability. One can then improve the success probability via median amplification: if a single estimate is correct with probability $\geq \frac{1}{2} + \eta$ for some η then the median of $\lceil \ln(\delta^{-1})/2\eta^2 \rceil$ estimates is incorrect with probability $\leq \delta$.

However, median amplification alone is not sufficient to accomplish phase estimation as we have defined it in Definition 5.1. This is because any λ_i that is $\approx 10\% \cdot 2^{-n-1}$ close to a multiple of $1/2^n$, the probability of correctly obtaining floor$(2^n \lambda_j)$ is actually *less* than 1/2! This means that no matter how much median amplification is performed, this approach cannot achieve α below $\approx 10\%$. (For reference, a lower bound γ on the probability is plotted in Figure 5.1, although reading this figure demands some notation from the proof of the following proposition. We see that when $|\lambda_j^{(x)} - 1/2| \lesssim 10\%/2$ the probability of obtaining floor$(2^n \lambda_j)$ is less than 1/2.)

Furthermore, even if the signal obtained from the QFT could obtain arbitrarily small α, it would not achieve the desired α^{-1} scaling. This is because the amplification gap η scales linearly with α, and median amplification scales as η^{-2} due to the Chernoff-Hoeffding theorem. Therefore, we would obtain a scaling of α^{-2}.

Fortunately, achieving $\sim \alpha^{-1}$ is possible, using a different method for reducing α: we perform estimation for r additional bits which are then ignored. This makes use of the fact that α is independent of the number of estimated bits, but the gap away from the multiples of $1/2^n$, which is $\alpha/2^{n+1}$, is not. Every time we increase n, the width of each disallowed interval is chopped in half, but to compensate the number of disallowed intervals is doubled. If we simply round away some of the bits, then we do not care about the additional disallowed regions introduced. If we estimate and round r additional bits, we suppress α by a factor of 2^{-r} while multiplying our runtime by a factor of 2^r - so the scaling is $\sim \alpha^{-1}$.

We still need median amplification, though. In order to use the above strategy we need rounding to be successful: if an eigenvalue λ_j falls between two bins floor$(2^n\lambda_j)$ and floor$(2^n\lambda_j) + 1$, then it must be guaranteed to be rounded to one of those bins with failure probability at most δ. Before amplification, the probability that rounding succeeds is $\geq 8/\pi^2$, a constant gap above 1/2 corresponding to $\alpha = 1/2$. We cannot guarantee a success probability higher than $8/\pi^2$ using the rounding trick alone, and thus need median amplification to finish the job.

Below, we split our analysis into two regimes: the $\alpha \leq 1/2$ regime and the $\alpha > 1/2$ regime. When $\alpha > 1/2$ then we do not really need the rounding trick described above, and we can achieve this accuracy with median amplification alone. Otherwise when $\alpha \leq 1/2$ we use median amplification to get to the point where rounding is likely to succeed, and then estimate r additional bits such that $\alpha 2^r \geq 1/2$.

Proposition 5.5. ***Standard phase estimation.*** *There exists an phase estimator with phases and garbage. Consider a unitary satisfying an (n, α)-rounding promise. Let:*

$$\gamma(x) := \frac{sin^2(\pi x)}{\pi^2 x^2} \qquad \eta_0 := \frac{8}{\pi^2} - \frac{1}{2} \qquad \delta_{med} := \frac{\delta^2}{6.25} \tag{5.36}$$

If $\alpha \leq 1/2$,let $r := \left\lceil \log_2\left(\frac{1}{2\alpha}\right)\right\rceil$. Then the phase estimator has query complexity:

$$(2^{n+r} - 1) \cdot \left\lceil \frac{\ln(\delta_{med}^{-1})}{2\eta_0^2} \right\rceil \tag{5.37}$$

and has $(n+r)\left\lceil \frac{\ln(\delta_{med}^{-1})}{2\eta_0^2} \right\rceil$ *qubits of garbage.*

Otherwise, if $\alpha > 1/2$*, let* $\eta := \gamma\left(\frac{1-\alpha}{2}\right) - \frac{1}{2}$*. Then the phase estimator has query complexity:*

$$(2^n - 1) \cdot \left\lceil \frac{\ln(\delta_{med}^{-1})}{2\eta^2} \right\rceil \tag{5.38}$$

and has $n\left\lceil \frac{\ln(\delta_{med}^{-1})}{2\eta^2} \right\rceil$ *qubits of garbage.*

Proof. We begin with the $\alpha > 1/2$ case which is simpler, and then extend to the $\alpha \le 1/2$ case. The following is a review of phase estimation, which has been modified to estimate $\text{floor}(2^n\lambda_j)$ rather than $\text{round}(2^n\lambda_j)$:

1. Prepare a uniform superposition over times: $|+^n\rangle|\psi_j\rangle = \frac{1}{\sqrt{2^n}}\sum_{t=0}^{2^n-1}|t\rangle|\psi_j\rangle$.

2. Apply U to the $|\psi_j\rangle$ register t times, along with an additional phase shift of $\frac{-2\pi t(1-\alpha)}{2^{n+1}}$ to account for flooring:

$$\to \frac{1}{\sqrt{2^n}}\sum_{t=0}^{2^n-1}|t\rangle \otimes U^t e^{-\frac{2\pi i t(1-\alpha)}{2^{n+1}}}|\psi_j\rangle = \frac{1}{\sqrt{2^n}}\sum_{t=0}^{2^n-1} e^{2\pi i t\left(\lambda_j - \frac{1-\alpha}{2^{n+1}}\right)}|t\rangle|\psi_j\rangle \tag{5.39}$$

3. Apply an inverse quantum Fourier transform to the $|t\rangle$ register:

$$\to \frac{1}{2^n}\sum_{t=0}^{2^n-1}\sum_{x=0}^{2^n-1} e^{2\pi i t\left(\lambda_j - \frac{1-\alpha}{2^{n+1}}\right)} e^{-\frac{2\pi i}{2^n}tx}|x\rangle|\psi_j\rangle \tag{5.40}$$

$$= \sum_{x=0}^{2^n-1}\left[\frac{1}{2^n}\sum_{t=0}^{2^n-1} e^{2i\pi t\left(\lambda_j - \frac{x}{2^n} - \frac{1-\alpha}{2^{n+1}}\right)}\right]|x\rangle|\psi_j\rangle \tag{5.41}$$

$$= \sum_{x=0}^{2^n-1}\beta\left(\lambda_j^{(x)}\right)|x\rangle|\psi_j\rangle \tag{5.42}$$

where in the final line we have defined:

$$\lambda_j^{(x)} := 2^n\lambda_j - x - \frac{1-\alpha}{2} \tag{5.43}$$

$$\beta(\lambda_j^{(x)}) := \frac{1}{2^n}\sum_{t=0}^{2^n-1} e^{2\pi i t\lambda_j^{(x)}/2^n}. \tag{5.44}$$

4. Let η and δ_{med} be as in the theorem statement, and let:

$$M := \left\lceil \frac{\log(\delta_{\text{med}}^{-1})}{2\eta^2} \right\rceil \tag{5.45}$$

   Repeat the above process for a total of M times. If we sum $\vec{x}$ over $\{0, ..., 2^n - 1\}^M$, then we obtain:

$$\rightarrow \sum_{\vec{x}} \bigotimes_{l=1}^{M} \beta\left(\lambda_j^{(x_l)}\right) |k_l\rangle \otimes |\psi_j\rangle \tag{5.46}$$

5. Run a sorting network, e.g. [Beals&12], on the M estimates, and copy the median into the output register.

The query complexity analysis is straightforward: each estimate requires $2^n - 1$ applications of controlled-U, and there are M estimates. So all we must do is demonstrate accuracy. It is clear that the process above implements a map of the form:

$$|0^n\rangle |0...0\rangle |\psi_j\rangle \rightarrow \sum_{x=0}^{2^n-1} e^{i\varphi_{j,x}} \sqrt{p_{j,x}}\, |x\rangle |\text{gar}_{j,k}\rangle |\psi_j\rangle \tag{5.47}$$

for some probabilities $p_{j,x}$ and phases $\varphi_{j,x}$, since this is just a general description of a unitary map that leaves the $|\psi_j\rangle$ register intact. This way of writing the map lets us bound the error in diamond norm to the ideal map

$$|0^n\rangle |0...0\rangle |\psi_j\rangle \rightarrow e^{i\varphi_j} |\text{floor}(\lambda_j 2^n)\rangle |\text{gar}_j\rangle |\psi_j\rangle \tag{5.48}$$

(selecting $\varphi_j := \varphi_{j,\mathrm{floor}(\lambda_j 2^n)}$ and $|\mathrm{gar}_j\rangle := |\mathrm{gar}_{j,\mathrm{floor}(\lambda_j 2^n)}\rangle$) without uncomputing while still employing the standard analysis of phase estimation which just reasons about the probabilities $p_{j,x}$. These can be bounded from a median amplification analysis of the probabilities $\left|\beta\left(\lambda_j^{(x)}\right)\right|^2$. Consider a vector of M estimates $\vec{x}$. Then:

$$p_{j,x} := \sum_{\substack{\vec{x} \text{ where} \\ \mathrm{median}(\vec{x})=x}} \prod_{l=1}^{M} \left|\beta(\lambda_j^{(x_l)})\right|^2 \tag{5.49}$$

First we show that probabilities $\left|\beta\left(\lambda_j^{(x)}\right)\right|^2$ associated with the individual estimates are bounded away from $\frac{1}{2}$. Using some identities we can rewrite:

$$\begin{aligned}
|\beta(\lambda_j^{(x)})|^2 &= \left|\frac{1}{2^n}\sum_{t=0}^{2^n-1} e^{2i\pi t\lambda_j^{(x)}/2^n}\right|^2 &(5.50)\\
&= \left|\frac{1}{2^n}\frac{e^{2i\pi\lambda_j^{(x)}}-1}{e^{2i\pi\lambda_j^{(x)}/2^n}-1}\right|^2 &(5.51)\\
&= \frac{1}{4^n}\frac{\sin^2(2\pi\lambda_j^{(x)}) + (\cos(2\pi\lambda_j^{(x)})-1)^2}{\sin^2(2\pi\lambda_j^{(x)}/2^n) + (\cos(2\pi\lambda_j^{(x)}/2^n)-1)^2} &(5.52)\\
&= \frac{1}{4^n}\frac{2-2\cos(2\pi\lambda_j^{(x)})}{2-2\cos(2\pi\lambda_j^{(x)}/2^n)} &(5.53)\\
&= \frac{1}{4^n}\frac{\sin^2(\pi\lambda_j^{(x)})}{\sin^2(\pi\lambda_j^{(x)}/2^n)} &(5.54)
\end{aligned}$$

Using the small angle approximation $\sin(\theta) \leq \theta$ for $0 \leq \theta \leq \pi$, we can derive:

$$|\beta(\lambda_j^{(x)})|^2 = \frac{1}{4^n}\frac{\sin^2(\pi\lambda_j^{(x)})}{\sin^2(\pi\lambda_j^{(x)}/2^n)} \geq \frac{1}{4^n}\frac{\sin^2(\pi\lambda_j^{(x)})}{\pi^2(\lambda_j^{(x)})^2/4^n} = \frac{\sin^2(\pi\lambda_j^{(x)})}{\pi^2(\lambda_j^{(x)})^2} =: \gamma(\lambda_j^{(x)}) \tag{5.55}$$

$\gamma(\lambda_j^{(x)})$ is a very tight lower bound to $|\beta(\lambda_j^{(x)})|^2$, and is plotted in Figure 5.1. We would like this probability to be larger than $\frac{1}{2}+\eta$ for some η when λ_j satisfies a rounding promise. This is guaranteed if we select:

$$\eta := \gamma\left(\frac{1-\alpha}{2}\right) - \frac{1}{2} \tag{5.56}$$

as in the theorem statement. From the figure, we see that when $\alpha > 1/2$ then $\eta > \eta_0$. This fact actually is not necessary for this construction to work, but observing this will be useful for the $\alpha \leq 1/2$ case. However, it is necessary to observe that when $\alpha > 1/2$ we are guaranteed that η is positive.

Then, from the rounding promise and the definition of $\lambda_j^{(x)}$, we are guaranteed that when $x = \text{floor}(2^n\lambda_j)$ then $|\beta(\lambda_j^{(x)})|^2 \geq 1/2 + \eta$. Now we perform a standard median amplification analysis. The probability of a particular estimate x being 'correct' is $\geq 1/2+\eta$, so the probability of being incorrect is $\leq 1/2-\eta$. The only way that a median of M estimates can be incorrect is if more than half of the estimates are incorrect. Let X be the random variable counting the number of incorrect estimates. Then we can invoke the Chernoff-Hoeffding theorem:

$$\Pr[\text{median incorrect}] \leq \Pr\left[X \geq M/2\right] \tag{5.57}$$

$$\leq \Pr\left[X \geq M/2 + \mathbb{E}(X) - (1/2-\eta)M\right] \tag{5.58}$$

$$\leq \exp\left(-2M\eta^2\right) \tag{5.59}$$

We can bound the above by δ_{med}, a constant we will fix later, if we select $M := \left\lceil \frac{\log(\delta_{\text{med}}^{-1})}{2\eta^2} \right\rceil$ as we did above.

We have demonstrated that $p_{\text{j,x}} \geq 1 - \delta_{\text{med}}$ whenever $x = \text{floor}(2^n\lambda_j)$. Now all that remains is a bound on the error in diamond norm. We begin by bounding the spectral norm. Let $p_j := p_{j,\text{floor}(2^n\lambda_j)}$ and observe that $1 \geq p_j \geq 1 - \delta_{\text{med}}$. Taking the difference between equations (5.47) and (5.48) we obtain:

$$\left| (\sqrt{p_j} - 1)e^{i\varphi_j} |\text{floor}(\lambda_j 2^n)\rangle |\text{gar}_j\rangle |\psi_j\rangle + \sum_{x, x \neq \text{floor}(\lambda_j 2^n)} e^{i\varphi_{j,x}} \sqrt{p_{j,x}} |x\rangle |\text{gar}_{j,x}\rangle |\psi_j\rangle \right| \tag{5.60}$$

$$= \sqrt{|(\sqrt{p_j} - 1)e^{i\varphi_j}|^2 + \sum_{x, x \neq \text{floor}(\lambda_j 2^n)} \left| e^{i\varphi_{j,x}} \sqrt{p_{j,x}} \right|^2} \tag{5.61}$$

$$= \sqrt{(\sqrt{p_j} - 1)^2 + \sum_{x, x \neq \text{floor}(\lambda_j 2^n)} p_{j_k}} \tag{5.62}$$

$$\leq \sqrt{(1 - \sqrt{1 - \delta_{\text{med}}})^2 + \delta_{\text{med}}} \tag{5.63}$$

$$\leq \sqrt{2 - 2\sqrt{1 - \delta_{\text{med}}} - \delta_{\text{med}} + \delta_{\text{med}}} \tag{5.64}$$

$$\leq \sqrt{2 - 2\sqrt{1 - \delta_{\text{med}}}} \tag{5.65}$$

Now we apply Lemma 5.4 to bound the diamond norm, and since the error is $\sim \sqrt{\delta_{\text{med}}}$ to leading order, we construct a bound that holds whenever $\sqrt{\delta_{\text{med}}} \leq 1$:

$$2\sqrt{2 - 2\sqrt{1 - \delta_{\text{med}}}} \leq 2.5\sqrt{\delta_{\text{med}}} \tag{5.66}$$

If we select $\delta_{\text{med}} := \delta^2/6.25$ then the diamond norm is bounded by δ.

Having completed the $\alpha > 1/2$ case, we proceed to the $\alpha \leq 1/2$ case. As stated above, the construction we just gave actually also works when $\alpha \leq 1/2$. However, the asymptotic dependence on α is like $\sim \alpha^{-2}$ which is not optimal.

Looking again at Figure 5.1, we see that the sum of the probability of two adjacent bins is at least $8/\pi^2$, even when $\lambda_j^{(x)}$ is in a region disallowed by the rounding promise. Therefore, if we perform median amplification with gap parameter η_0 we are guaranteed that values of $\lambda_j^{(x)}$ that fall into a rounding gap will at least be rounded to an adjacent bin.

We can use this fact to achieve an $\sim \alpha^{-1}$ dependence. Recall that α is the fraction of the range $[0, 1)$ where λ_j are not allowed to appear due to the rounding promise, independent of n. If we perform phase estimation with $n+1$ rather than n, then the width of each disallowed region is cut in half from $\alpha 2^{-n}$ to $\alpha 2^{-n-1}$ - but we also double the number of disallowed regions. However, if we simply ignore the final bit, then, because we are amplifying to η_0, any values of λ_j that fall into one of the newly introduced regions will simply be rounded. Thus, we cut the gaps in half without introducing any new gaps, so α is cut in half. We will make this argument more formal in a moment.

So, in order to achieve a particular α, we observe from Figure 5.1 that if we amplify to η_0 then the gaps have width $\frac{1}{2} \cdot 2^{-n}$. If we estimate an additional r bits, then the gaps have width $\frac{1}{2} \cdot 2^{-n-r}$. Solving $\frac{1}{2} \cdot 2^{-n-r} \leq \alpha 2^{-n}$ for r, we obtain

$$r := \left\lceil \log_2 \left(\frac{1}{2\alpha} \right) \right\rceil . \tag{5.67}$$

We briefly make the n explicit in $\lambda_j^{(x)}$ by writing $\lambda_j^{(n,x)}$. After running the protocol at the beginning of the proof with $n+r$ bits, we obtain, for $\vec{x} \in \{0, ..., 2^n - 1\}^M$, the state:

$$\sum_{\vec{x}} |\text{median}(\vec{x})\rangle \otimes \bigotimes_{l=1}^{M} \sum_{t=0}^{2^r-1} \beta(\lambda_j^{(n+r,\ 2^r x_l + t)}) |2^r k_l + t\rangle \otimes |\psi_j\rangle \tag{5.68}$$

$$= \sum_{x=0}^{2^n-1} |x\rangle \otimes \sum_{\substack{\vec{x} \text{ where} \\ \text{median}(\vec{x}) = kx}} \bigotimes_{l=1}^{M} \sum_{t=0}^{2^r-1} \beta(\lambda_j^{(n+r,\ 2^r x_l + t)}) |2^r x_l + t\rangle \otimes |\psi_j\rangle \tag{5.69}$$

$$= \sum_{x=0}^{2^n-1} \sqrt{p_{j,x}} e^{i\varphi_{j,x}} |x\rangle \otimes |\text{gar}_{j,x}\rangle \otimes |\psi_j\rangle \tag{5.70}$$

where in the last step we have defined the normalized state $|\text{gar}_x\rangle$, the probability $p_{j,x}$, and the phase $\varphi_{j,x}$ via:

$$\sqrt{p_{j,x}} e^{i\varphi_{j,x}} |\text{gar}_{j,x}\rangle := \sum_{\substack{\vec{x} \text{ where} \\ \text{median}(\vec{x}) = x}} \bigotimes_{l=1}^{M} \sum_{t=0}^{2^r-1} \beta(\lambda_j^{(n+r,\ 2^r x_l + t)}) |2^r x_l + t\rangle \tag{5.71}$$

Now if we can demonstrate that $p_{j,x} \geq 1 - \delta_{\text{med}}$ when $x = \text{floor}(2^n \lambda_j)$ then the same argument as in (5.60-5.65) holds and we are done. Observe that:

$$p_{j,x} := \sum_{\substack{\vec{x} \text{ where} \\ \text{median}(\vec{x})=x}} \prod_{l=1}^{M} \sum_{t=0}^{2^r-1} \left| \beta(\lambda_j^{(n+r,\ 2^r x_l + t)}) \right|^2 \tag{5.72}$$

Say $x = \text{floor}(2^n \lambda_j)$. Then, consider $t := \text{floor}(2^{n+r}\lambda_j) - x2^r$, which is an integer $\in \{0, ..., 2^r - 1\}$. Then, looking at Figure 5.1, we see that:

$$|\beta(\lambda_j^{(n+r,\ 2^r x+t)})|^2 + |\beta(\lambda_j^{(n+r,\ 2^r x+t+1)})|^2 \geq \gamma(\lambda_j^{(n+r,\ 2^r x+t)}) + \gamma(\lambda_j^{(n+r,\ 2^r x+t)} - 1) \geq \frac{1}{2} + \eta_0 \tag{5.73}$$

Therefore the probability that the first n bits are x is $\sum_{t=0}^{2^r-1} \left| \beta(\lambda_j^{(n+r,\ 2^r x+t)}) \right|^2 \geq 1/2 + \eta_0$, which is all that was required to perform the median amplification argument above. So we can conclude that $p_{j,k} \geq 1 - \delta_{\text{med}}$, so the modified algorithm retains the same accuracy. □

5.2 Coherent Iterative Estimation

In this section we present the novel algorithms for phase estimation and energy estimation. As described in the introduction, both of these feature a strong similarity to 'iterative phase estimation' [Kit95], where the bits of the estimate are obtained one at a time. Unlike iterative phase estimation however, the state is never measured and the entire process is coherent. We therefore name these algorithms as 'coherent iterative estimators'.

Another similarity that these new algorithms share with the original iterative phase estimation is that the less significant bits are taken into account when obtaining the current bit. This greatly reduces the amount of amplification required for the later bits, so the runtime is vastly dominated by the estimation of the least significant bit. We will go into more detail on this later.

To permit discussion of coherent iterative estimation of phases and energies in a unified manner, we fit this idea into the modular framework of Definition 5.2 and Lemma 5.3. A 'coherent iterative estimator' obtains a single bit of the estimate, given access to all the previous bits. Several invocations of a coherent iterative estimator yield a regular estimator as in Definition 5.2. Furthermore, we can choose to uncompute the garbage at the very end using Lemma 5.3, or, as we will show, we can remove the garbage early, which prevents it from piling up.

Definition 5.6. *A **coherent iterative phase estimator** is a protocol that, given some n, α, any $k \in \{0, ..., n-1\}$, and any error target $\delta > 0$, produces a quantum circuit involving calls to controlled-U and $U^\dagger$. If the unitary U satisfies an (n, α)-rounding promise, then this circuit implements a quantum channel that is δ-close to some map that performs:*

$$|0\rangle |\Delta_k\rangle |\psi_j\rangle \rightarrow |bit_k(\lambda_j)\rangle |\Delta_k\rangle |\psi_j\rangle \tag{5.74}$$

Here $bit_k(\lambda_j)$ is the $(k+1)$'th least significant bit of an n-bit binary expansion, and $|\Delta_k\rangle$ is a k-qubit register encoding the k least significant bits. (For example, if $n = 4$ and $\lambda_j = 0.1011...$ then $bit_2(\lambda_j) = 0$ and $\Delta_2 = 11$.) Note that the target map is only constrained on a subspace of the input Hilbert space, and can be anything else on the rest.

*An **coherent iterative energy estimator** is the same thing, just with controlled-U_H and $U_H^\dagger$ queries to some block encoding U_H of a Hamiltonian H that satisfies an (n, α)-rounding promise.*

A coherent iterative estimator can also be with garbage and/or with phases, just like in Definition 5.2.

Lemma 5.7. ***Stitching together coherent iterative estimators.*** *Given a coherent iterative phase/energy estimator with query complexity $Q(n, k', \alpha, \delta')$, we can construct a non-iterative phase/energy estimator with query complexity:*

$$\sum_{k=0}^{n-1} Q\left(n, k, \alpha, \delta \cdot 2^{-k-1}\right) \tag{5.75}$$

The non-iterative estimator has phases if and only if the coherent iterative estimator has phases, and if the coherent iterative estimator has m qubits of garbage then the iterative estimator has nm qubits of garbage.

Proof. We will combine n many coherent iterative phase/energy estimators, for $k = 0, 1, 2, .., n-1$. The diamond norm satisfies a triangle inequality so if we let the k'th iterative estimator have an error $\delta_k := \delta 2^{-k-1}$ then the overall error will be:

$$\sum_{k=0}^{n-1} \delta_k \leq \frac{\delta}{2} \sum_{k=0}^{n-1} 2^{-k} = \frac{\delta}{2}(2 - 2^{1-n}) \leq \delta \tag{5.76}$$

So now all that is left is to observe that the exact iterative estimators chain together correctly. This should be clear by observing that for all $k > 0$:

$$|\Delta_k\rangle = |\text{bit}_{k-1}(\lambda_j)\rangle \otimes ... \otimes |\text{bit}_0(\lambda_j)\rangle \tag{5.77}$$

and $|\Delta_0\rangle = 1 \in \mathbb{C}$ since when $k = 0$ there are no less significant bits. So the k'th iterative estimator takes the k least significant bits as input and computes one more bit, until finally at $k = n - 1$ we have:

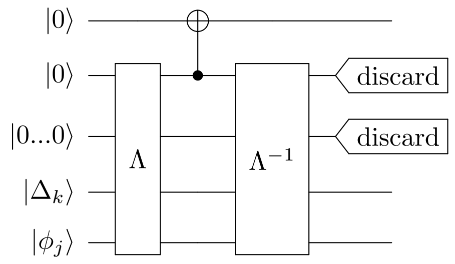
$$|\text{bit}_{n-1}(\lambda_j)\rangle \, |\Delta_{n-1}\rangle = |\text{bit}_{n-1}(\lambda_j)\rangle \otimes ... \otimes |\text{bit}_0(\lambda_j)\rangle = |\text{floor}(2^n \lambda_j)\rangle \tag{5.78}$$

The total query complexity is just the sum of the n invocations of the iterative estimators from $k = 0, ..., n-1$ with error δ_k.

If the iterative estimator has garbage, then the garbage from each of the n invocations just piles up. Similarly, if the estimator is with phases, and has an $e^{i\varphi_{j,k}}$ for the k'th invocation, then the composition of the maps will have a phase $\prod_{k=0}^{n-1} e^{i\varphi_{j,k}}$. □

Lemma 5.8. ***Removing garbage and phases from iterative estimators.*** *Given a coherent iterative phase/energy estimator with phases and/or garbage and with query complexity $Q(n, k, \alpha, \delta')$ that has a unitary implementation, we can construct a coherent iterative phase/energy estimator without phases and without garbage with query complexity $2Q(n, k, \alpha, \delta/2)$.*

Proof. This argument proceeds exactly the same as Lemma 5.3, just with the extra $|\Delta_k\rangle$ register trailing along. If Λ is the channel that implements the iterative estimator with garbage and/or phases, then we obtain an estimator without garbage and phases via:



(5.79)

□

The advantage of the modular framework we just presented is that maximizes the amount of flexibility when implementing these algorithms. How exactly uncomputation is performed will vary from application to application, and depending on the situation uncomputation may be performed before invoking Lemma 5.7 using Lemma 5.8, after Lemma 5.7 using Lemma 5.3, or not at all!

Of course, the idea of uncomputation combined with iterative estimation itself is quite simple, so given a complete understanding of the techniques we present the reader may be able to perform this modularization themselves. However, we found that this presentation significantly de-clutters the presentation of the main algorithms, those that actually implement the coherent iterative estimators from Definition 5.6. While the intuitive concept behind these strategies is not so complicated, the rigorous presentation and error analysis is quite intricate. We therefore prefer to discuss uncomputation separately.

5.2.1 Coherent Iterative Phase Estimation

This section describes our first novel algorithm, presented in Theorem 5.9. Before stating the algorithm in complete detail, performing an error analysis, and showing how to optimize the performance up to constant factors, we outline the tools that we will need and give an intuitive description.

As stated in the introduction, a very similar algorithm was independently discovered by [MRTC21]. The techniques of the two algorithms for phase estimation feature some minor differences: our work is more interested in maintaining coherence of the input state, the algorithm's constant-factor runtime improvement over prior art, and our runtime is $O(2^n)$ rather than $O(n2^n)$ ($O(n)$ vs $O(n \log n)$ respectively in the language of [MRTC21]). On the other hand, [MRTC21] elegantly show how a quantum Fourier transform emerges as a special case of the algorithm, and their presentation is significantly more accessible. Both methods avoid use of ancillae entirely.

A key tool for these algorithms is the block encoding from Definition 3.6, which allows us to manipulate arbitrary non-unitary matrices.

A unitary matrix is a trivial block encoding of itself. In this sense, we already have a block encoding of the matrix:

$$U = \sum_j e^{2\pi i \lambda_j} \left|\psi_j\right\rangle \left\langle\psi_j\right| \tag{5.80}$$

Recall that the goal of the coherent iterative estimator is to compute $\text{bit}_k(\lambda_j)$. The strategy involves preparing an approximate block encoding of:

$$\sum_j \text{bit}_k(\lambda_j) \left|\psi_j\right\rangle \left\langle\psi_j\right| \tag{5.81}$$

We begin by rewriting the above expression a bit. Recall that Δ_k is an integer from 0 to $2^{k+1} - 1$ encoding the k less significant bits of λ_j. If we subtract $\Delta_k/2^n$ from λ_j, we obtain a multiple of $1/2^{n-k}$ plus something $< 1/2^n$ which we floor away. Then, $\text{bit}_k(\lambda_j)$ indicates if this is an even or an odd multiple. That means we can write:

$$\text{bit}_k(\lambda_j) = \text{parity}\left(\text{floor}\left(2^{n-k}\left(\lambda_j - \frac{\Delta_k}{2^n}\right)\right)\right) \tag{5.82}$$

Since $\lambda_j - \frac{\Delta_k}{2^n}$ is a multiple of $1/2^{n-k}$ plus something $< 1/2^n$, we equivalently have that $2^{n-k}\left(\lambda_j - \frac{\Delta_k}{2^n}\right)$ is an integer plus something less than $1/2^k$. For such values, we can write the parity(floor(x)) function in terms of a squared cosine that has been ‘amplified’:

$$\text{parity}(\text{floor}(x)) = \text{amp}\left(\cos^2\left(\frac{\pi}{2}\left[x + \phi\right]\right)\right) \tag{5.83}$$

$$\text{amp}(x) = \begin{cases} 1 \text{ if } x > 1/2 \\ 0 \text{ if } x < 1/2 \end{cases} \tag{5.84}$$

where the shift ϕ centers the extrema of the cosine in the intervals where x occurs. Therefore:

$$\begin{aligned} \mathrm{bit}_k(\lambda_j) &= \mathrm{amp}\left(\cos^2\left(\frac{\pi}{2}\left[2^{n-k}\left(\lambda_j - \frac{\Delta_k}{2^n}\right) + \phi\right]\right)\right) && (5.85)\\ &= \mathrm{amp}\left(\cos^2\left(\pi\left[2^{n-k-1}\left(\lambda_j - \frac{\Delta_k}{2^n}\right) + \phi/2\right]\right)\right) && (5.86)\\ &= \mathrm{amp}\left(\cos^2\left(\pi\lambda_j^{(k)}\right)\right) && (5.87)\end{aligned}$$

Where we have defined:

$$\lambda_j^{(k)} := 2^{n-k-1}\left(\lambda_j - \frac{\Delta_k}{2^n}\right) + \phi_k \tag{5.88}$$

for some k-dependent choice of phase $\phi_k = \phi/2$.

By applying a phase shift conditioned on the $|\Delta_k\rangle$ register, and then iterating it 2^{n-k-1} times, we can construct a block encoding of the unitary with eigenvalues $\lambda_j^{(k)}$. Our goal is now to transform this unitary as follows:

$$\sum_j e^{2\pi i\lambda_j^{(k)}} |\psi_j\rangle\langle\psi_j| \quad \to \quad \sum_j \left[\mathrm{amp}\left(\cos^2\left(\pi\lambda_j^{(k)}\right)\right)\right] |\psi_j\rangle\langle\psi_j| \tag{5.89}$$

To obtain a cosine use linear combinations of unitaries [BCK15, CKS15] to take a take a linear combination with the identity:

$$\begin{aligned} \sum_j \frac{e^{2\pi i\lambda_j^{(k)}} + 1}{2} |\psi_j\rangle\langle\psi_j| &= \sum_j \cos\left(\pi\lambda_j^{(k)}\right)\left[e^{i\pi\lambda_j^{(k)}}|\psi_j\rangle\right]\langle\psi_j| && (5.90)\\ &= \sum_j \left|\cos\left(\pi\lambda_j^{(k)}\right)\right| \cdot \left[\pm e^{i\pi\lambda_j^{(k)}}|\psi_j\rangle\right]\langle\psi_j| && (5.91)\end{aligned}$$

where on the previous line $\pm$ indicates sign $\left(\cos\left(\pi\lambda_j^{(k)}\right)\right)$. This way the above is a singular value decomposition of the block-encoded matrix. So all that is left to do is to approximately transform the singular values a of the matrix above by:

$$a \to \mathrm{amp}(a^2) \tag{5.92}$$

We will accomplish this using singular value transformation. See Theorem 3.17. We will also be making use of Remark 3.19.

Singular value transformation can perform the desired conversion if we can construct a polynomial $A(x)$ such that:

$$A(x) \approx \text{amp}(x) = \frac{1}{2} - \frac{1}{2}\text{sign}(2x-1) \tag{5.93}$$

If we invoke the above lemma with the even polynomial $A(x^2)$ we get an approximation to the desired block encoding of $\sum_j \text{bit}_k(\lambda_j) |\psi_j\rangle \langle\psi_j|$.

In particular, the behavior we must capture in the polynomial approximation is that $A(x) \approx 1$ when $x \in [0, 1/2 - \eta]$ and $A(x) \approx 0$ when $x \in [1/2 + \eta, 1]$ for some gap η away from 1/2. If we view the input x as a probability, then $A(x)$ essentially 'amplifies' this probability to something close to 0 or 1 (and additionally flips the outcome). We hence call $A(x)$ an 'amplifying polynomial'.

One approach to constructing such an amplifying polynomial is to simply adapt a classical algorithm for amplification. We stated this method in the introduction: the polynomial is the probability that the majority vote of several coin tosses is heads, where each coin comes up heads with probability x. Then the desired properties can be obtained from the Chernoff-Hoeffding theorem [Diak09]. The number of coins we have to toss to accomplish a particular η, δ is the degree of the polynomial, which is bounded by $O(\eta^{-2}\log(\delta^{-1}))$.

However, this polynomial does not achieve the optimal η dependence of $O(\eta^{-1})$. This might be achieved by a polynomial inspired by a quantum algorithm for approximate counting, which does achieve the $O(\eta^{-1})$ dependence. But rather than go through such a complicated construction we simply adapt a polynomial approximation to the sign(x)

function developed in [LC17], which accomplishes the optimal $O(\eta^{-1}\log(\delta^{-1}))$. This is presented in Corollary 3.23.

The method for obtaining $M_{\eta\to\delta}$ given η and δ is complicated enough that it is not worth re-stating here. However, the complexity is by all means worth it: in our numerical analyses we found that the polynomials presented in Appendix A of [LC17] feature excellent performance in terms of degree. This is a major source of the query complexity speedup of our algorithms.

The value of δ is chosen such that the final error in diamond norm is bounded. The value of η depends on how far away $\cos^2\left(\pi\lambda_j^{(k)}\right)$ is from $\frac{1}{2}$. Of course, there are several possible values of $\lambda_j^{(k)}$ where $\cos^2\left(\pi\lambda_j^{(k)}\right) = \frac{1}{2}$ exactly, so $\eta = 0$ and amplification is impossible. This is where the rounding promise comes in: it ensures that λ_Δ is always sufficiently far from such values, so that we can guarantee that $\cos^2\left(\pi\lambda_j^{(k)}\right)$ is always either $\geq \frac{1}{2} + \frac{\alpha}{2}$ or $\leq \frac{1}{2} + \frac{\alpha}{2}$. So we select $\eta = \alpha/2$.

When $k = 0$ then indeed the rounding promise is the only thing guaranteeing that amplification will succeed. However, if the less significant bits have already been computed then the set of values that $\lambda_j^{(k)}$ can take is restricted. This is because the previous bits of λ_j have been subtracted off. This widens the region around the solutions of $\cos^2\left(\pi\lambda_j^{(k)}\right) = \frac{1}{2}$ where λ_Δ cannot be found, allowing us to increase η. Furthermore, this can be done without relying on the rounding promise anymore: bits with $k \geq 1$ are guaranteed to be deterministic even if no rounding promise is present. That means that if the rounding promise is violated, then only the least significant bit can be wrong. The polynomials are sketched in Figure 5.2.

After constructing the amplifying polynomial $A_{\eta\to\delta}$, we use singular value transformation to apply $A_{\eta\to\delta}(x^2)$ which is even and therefore fixed parity as required by Theorem 3.17. Now we have an approximate block encoding of $\sum_j \text{bit}_k(\lambda_j) |\psi_j\rangle \langle\psi_j|$, which is in fact a projector. In the introduction we stated that we would use a block-measurement theorem to compute the map:

$$|0\rangle \otimes |\psi\rangle \to |1\rangle \otimes \Pi |\psi\rangle + |0\rangle \otimes (I - \Pi) |\psi\rangle \tag{5.94}$$

given an approximate block encoding of Π. However, this general tool involves uncomputation which we specifically wanted to modularize. The fact that we are already measuring errors in terms of the diamond norm means that Lemmas 5.3 and 5.8 are already capable of dealing with garbage. We therefore defer the proof of this general tool to Section 5.4, specifically Theorem 5.16.

There is another reason to not use Theorem 5.16 in a black-box fashion, specifically for coherent iterative phase estimation. The block encoding of $\sum_j \text{bit}_k(\lambda_j) |\psi_j\rangle \langle\psi_j|$ actually features only one ancilla qubit: the qubit we used to take the linear combination of U and the identity. That means that the block encoding itself is already very close to the map $|0\rangle \otimes |\psi\rangle \to |0\rangle \otimes \Pi |\psi\rangle + |1\rangle \otimes (I - \Pi) |\psi\rangle$ (note the flipped output qubit). The details of this usage of the block encoding will appear in the proof.

This completes the sketch of the procedure to implement the map

$$|0\rangle |\Delta_k\rangle |\psi_j\rangle \to e^{i\varphi_j} |\text{bit}_k(\lambda_j)\rangle |\Delta_k\rangle |\psi_j\rangle \tag{5.95}$$

for some phases φ_j. We can now state the protocol in detail, and perform the accuracy analysis.

Theorem 5.9. ***Coherent Iterative Phase Estimation.*** *There is a coherent iterative phase estimator with phases (and no garbage) and with query complexity:*

$$2^{n-k} \cdot M_{\eta \to \delta_{amp}} \tag{5.96}$$

where in the above $M_{\eta\to\delta}$ is as in Corollary 3.23, $\eta := \frac{1}{2} - \frac{1}{2^k}\left(\frac{1}{2}+\frac{\alpha}{2}\right)$ if $k \geq 1$ and $\eta := \frac{\alpha}{2}$ if $k = 0$, and δ_{amp} can be chosen to be $(1-10^{-m})\delta^2/24$ for any $m > 0$.

Proof. We construct the estimator as follows:

1. Construct a unitary that performs a phase shift depending on Δ_k - we call this $e^{-2\pi i\hat{\Delta}_k/2^n}$ employing some notation inspired by physics literature.

$$e^{-2\pi i\hat{\Delta}_k/2^n} := \sum_{\Delta_k=0}^{2^k-1} e^{-2\pi i\Delta/2^n} \left|\Delta_k\right\rangle \left\langle\Delta_k\right| = \bigotimes_{j=0}^{k-1} e^{-2\pi i\pi 2^{j-n}} \tag{5.97}$$

2. Rewrite the oracle unitary in this notation:

$$e^{2\pi i\hat{\lambda}} := U = \sum_j e^{2\pi i\lambda_j} \left|\psi_j\right\rangle \left\langle\psi_j\right| \tag{5.98}$$

   Then define:

$$\phi_0 := 1 - \text{mean}\left(\frac{1}{2}+\frac{\alpha}{2}, 1\right) \tag{5.99}$$

$$\phi_k := 1 - \text{mean}\left(\frac{1}{2}, \frac{1}{2}+\frac{1}{2^k}\left(\frac{1}{2}+\frac{\alpha}{2}\right)\right) \text{ if } k \geq 1 \tag{5.100}$$

$$\hat{\lambda}^{(k)} := 2^{n-k-1}\left(\hat{\lambda} - \frac{\hat{\Delta}_k}{2^n}\right) + \phi_k, \tag{5.101}$$

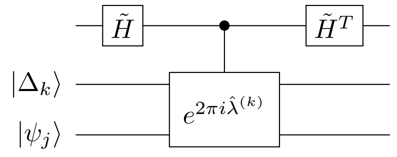
   and implement the corresponding phase shift:

$$e^{2\pi i\hat{\lambda}^{(k)}} := \left(e^{-2\pi i\hat{\Delta}_k/2^n} \otimes e^{2\pi i\hat{\lambda}}\right)^{2^{n-k-1}} \cdot e^{2\pi i\phi_k} \tag{5.102}$$

   This unitary acts jointly on the $\left|\Delta_k\right\rangle$ and $\left|\psi_j\right\rangle$ inputs. Since $k \in \{0, ..., n-1\}$, the exponent 2^{n-k-1} is always an integer.

3. Let $\tilde{H} := \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ i & -i\end{bmatrix}$ be a slightly modified Hadamard gate, and consider the following

unitary, implemented via $e^{2\pi i \hat{\lambda}^{(k)}}$:

$U^{(k)}_{\text{signal}} :=$ 

(5.103)

Observe that:

$$\begin{aligned}
U^{(k)}_{\text{signal}} &= \sum_{\Delta_k}\sum_j \tilde{H}\begin{bmatrix} 1 & 0 \\ 0 & e^{2\pi i \lambda_j^{(k)}} \end{bmatrix} \tilde{H}^T \otimes |\Delta_k\rangle\langle\Delta_k| \otimes |\psi_j\rangle\langle\psi_j| && (5.104)\\
&= \sum_{\Delta_k}\sum_j e^{\pi i \lambda_j^{(k)}} \cdot \tilde{H}\begin{bmatrix} e^{-\pi i \lambda_j^{(k)}} & 0 \\ 0 & e^{\pi i \lambda_j^{(k)}} \end{bmatrix} \tilde{H}^T \otimes |\Delta_k\rangle\langle\Delta_k| \otimes |\psi_j\rangle\langle\psi_j| && (5.105)\\
&= \sum_{\Delta_k}\sum_j e^{\pi i \lambda_j^{(k)}} \cdot \begin{bmatrix} \cos\left(\pi\lambda_j^{(k)}\right) & \sin\left(\pi\lambda_j^{(k)}\right) \\ \sin\left(\pi\lambda_j^{(k)}\right) & -\cos\left(\pi\lambda_j^{(k)}\right) \end{bmatrix} \otimes |\Delta_k\rangle\langle\Delta_k| \otimes |\psi_j\rangle\langle\psi_j| && (5.106)
\end{aligned}$$

In other words, $U^{(k)}_{\text{signal}}$ is a block encoding of:

$$\sum_{\Delta_k}\sum_j \left|\cos\left(\pi\lambda_j^{(k)}\right)\right| \left[\pm e^{\pi i \lambda_j^k} |\Delta_k\rangle|\psi_j\rangle\right]\left[\langle\Delta_k|\langle\psi_j|\right] \tag{5.107}$$

The above is a singular value decomposition of the block-encoded matrix.

4. Choose the amplification threshold via:

$$\eta_k := \frac{1}{2} - \frac{1}{2^k}\left(\frac{1}{2}+\frac{\alpha}{2}\right) \text{ if } k \geq 1 \tag{5.108}$$

$$\eta_0 := \frac{\alpha}{2} \tag{5.109}$$

Also, let $\delta_{\text{amp}} < 1$ be an error threshold we will pick later. Now, let $A_{\eta\to\delta_{\text{amp}}}(x)$ be the polynomial from Corollary 3.23. Viewing $U^{(k)}_{\text{signal}}$ as a block encoding of $\cos\left(\pi\lambda_j^{(k)}\right)$, apply singular value transformation as in Theorem 3.17 to $U^{(k)}_{\text{signal}}$ with a polynomial $\tilde{p}(x)$ approximating $A_{\eta\to\delta_{\text{amp}}}(x^2)$ to accuracy δ_{svt}, which we also pick later.

Theorem 3.17 applies because $A_{\eta\to\delta_{\text{amp}}}(x^2)$ is even. Furthermore, $U^{(k)}_{\text{signal}}$ only has one ancilla qubit, and has the special form where the it implements a reflection on the ancilla (5.106). Thus, by Remark 3.19 the circuit from Theorem 3.17 can be constructed to just have one ancilla.

Call the resulting circuit $U_{\text{svt}}^{(k)}$, which implements the unitary:

$$U_{\text{svt}}^{(k)} = \sum_{\Delta_k} \sum_j \begin{bmatrix} \tilde{p}\left(\cos\left(\pi\lambda_j^{(k)}\right)\right) & \cdot \\ \gamma(\lambda_j^{(k)}) & \cdot \end{bmatrix} \otimes [|\Delta_k\rangle |\psi_j\rangle] [\langle\Delta_k| \langle\psi_j|] \tag{5.110}$$

for some matrix element $\gamma(\lambda_j^{(k)})$.

Now we prove that $U_{\text{svt}}^{(k)}$ is an iterative phase estimator with phases. It implements the map:

$$|0\rangle |\Delta_k\rangle |\psi_j\rangle \to \left(\tilde{p}\left(\cos\left(\pi\lambda_j^{(k)}\right)\right)|0\rangle + \gamma(\lambda_j^{(k)})|1\rangle\right)|\Delta_k\rangle |\psi_j\rangle \tag{5.111}$$

Note that $U_{\text{svt}}^{(k)}$ is a block encoding with only one ancilla, and that ancilla is the output qubit of the map.

We must show that $U_{\text{svt}}^{(k)}$ is close in diamond norm to a map that leaves the first qubit as $|\text{bit}_k(\lambda_j)\rangle$ whenever Δ_k encodes the k least significant bits of an n-bit binary expansion of λ_j.

To study this map we will proceed through the recipe above, proving statements about the expressions encountered along the way. In step 2. we defined:

$$\lambda_j^{(k)} := 2^{n-k-1}\left(\lambda_j - \frac{\hat{\Delta}_k}{2^n}\right) + \phi_k, \tag{5.112}$$

We discuss the relationship between $\lambda_j^{(k)} - \phi_k$ and $\text{bit}_k(\lambda_j)$. If $k = 0$ then $\Delta_k = 0$, and due to the rounding promise we find λ_j in regions of the form $\frac{m}{2^n} + \left[\frac{\alpha}{2^n}, \frac{1}{2^n}\right]$ for integers m. Thus, we find $\lambda_j^{(0)} - \phi_0$ in regions of the form $\frac{m}{2} + \left[\frac{\alpha}{2}, \frac{1}{2}\right]$. The function $\text{bit}_k(\lambda_j)$ just indicates the parity of m. We can also write this as:

$$\mathrm{bit}_0(\lambda_j) = \begin{cases} 0 & \text{if } \lambda_j^{(0)} - \phi_0 \in \mathrm{floor}(\lambda_j^{(0)} - \phi_0) + [\frac{\alpha}{2}, \frac{1}{2}] \\ 1 & \text{if } \lambda_j^{(0)} - \phi_0 \in \mathrm{floor}(\lambda_j^{(0)} - \phi_0) + [\frac{1}{2} + \frac{\alpha}{2}, 1] \end{cases} \tag{5.113}$$

If $k > 0$ then we also can show a similar property. While for $k = 0$ we used the rounding promise to guarantee that $\lambda_j^{(0)} - \phi_0$ only falls into certain regions, for larger k we simply use the fact that Δ_k has been subtracted off in the definition of $\lambda_j^{(k)}$. That means that the regions where $\mathrm{bit}_{<k}(\lambda_j) = 1$ are no longer possible. We find:

$$\mathrm{bit}_k(\lambda_j) = \begin{cases} 0 & \text{if } \lambda_j^{(k)} - \phi_k \in \mathrm{floor}(\lambda_j^{(k)} - \phi_k) + \left[0, \frac{1}{2^k}\left(\frac{1}{2} + \frac{\alpha}{2}\right)\right] \\ 1 & \text{if } \lambda_j^{(k)} - \phi_k \in \mathrm{floor}(\lambda_j^{(k)} - \phi_k) + \left[\frac{1}{2}, \frac{1}{2} + \frac{1}{2^k}\left(\frac{1}{2} + \frac{\alpha}{2}\right)\right] \end{cases} \quad \text{when } k > 0 \tag{5.114}$$

Note that the claim for $k > 0$ did not make use of the rounding promise, and is true regardless of if the rounding promise holds. These regions are shown in Figure 5.3.

In step 3. we defined U_{signal} which is a block encoding of $\cos\left(\pi\lambda_j^{(k)}\right)$. Later, this will approximately be transformed by singular value transformation via $x \to A_{\eta\to\delta_{\mathrm{amp}}}(x^2)$, so we employ a trigonometric identity:

$$\cos^2\left(\pi\lambda_j^{(k)}\right) = \frac{1 + \cos\left(2\pi\lambda_j^k\right)}{2} \tag{5.115}$$

Clearly this is a probability, and since cosine has period 2π, the $\mathrm{floor}(\lambda_j^{(k)} - \phi_k)$ term does not matter. We argue that this probability is either $\geq 1/2 + \eta_k$ or $\leq 1/2 - \eta_k$ depending on $\mathrm{bit}_k(\lambda_j)$. See Figure 5.3. The idea is that the ϕ_k are chosen precisely so that the troughs and peaks of $\cos^2\left(\pi\lambda_j^k\right)$ line up with the centers of the intervals corresponding to $\mathrm{bit}_k(\lambda_j) = 0$ and $\mathrm{bit}_k(\lambda_j) = 1$ respectively. Then, the nodes of $\cos^2\left(\pi\lambda_j^k\right)$ line up with the midpoints of the gaps between the intervals. A line of slope 2 connecting a trough to a peak then forms an upper/lower bound on $\cos^2\left(\pi\lambda_j^k\right)$. If we select $\eta_k := \frac{1}{2} - \frac{1}{2^k}\left(\frac{1}{2} + \frac{\alpha}{2}\right)$ if and $\eta_0 := \frac{\alpha}{2}$ then these bounds show that $\cos^2\left(\pi\lambda_j^{(k)}\right)$ is alternatingly $\leq \frac{1}{2} - \eta_k$ and $\geq \frac{1}{2} + \eta_k$.

Then we have:

$$\cos^2(\pi\lambda_j^{(k)}) \text{ is } \begin{cases} \leq \frac{1}{2} + \eta_k & \text{if } \text{bit}_k(\lambda_j) = 0 \\ \geq \frac{1}{2} - \eta_k & \text{if } \text{bit}_k(\lambda_j) = 1 \end{cases} \tag{5.116}$$

By Corollary 3.23:

$$A_{\eta_k \to \delta_{\text{amp}}}\left(\cos^2(\pi\lambda_j^{(k)})\right) \text{ is } \begin{cases} \leq \delta_{\text{amp}} & \text{if } \text{bit}_k(\lambda_j) = 0 \\ \geq 1 - \delta_{\text{amp}} & \text{if } \text{bit}_k(\lambda_j) = 1 \end{cases} \tag{5.117}$$

And finally, since $\tilde{p}(x)$ approximates $A_{\eta_k \to \delta_{\text{amp}}}(x^2)$ to accuracy δ_{svt}:

$$\tilde{p}\left(\cos(\pi\lambda_j^{(k)})\right) \text{ is } \begin{cases} \leq \delta_{\text{amp}} + \delta_{\text{svt}} & \text{if } \text{bit}_k(\lambda_j) = 0 \\ \geq 1 - \delta_{\text{amp}} - \delta_{\text{svt}} & \text{if } \text{bit}_k(\lambda_j) = 1 \end{cases} \tag{5.118}$$

The circuit for U_{svt} is completely unitary, so the resulting state is normalized. Therefore:

$$\left|\tilde{p}\left(\cos\left(\pi\lambda_j^{(k)}\right)\right)\right|^2 + \left|\gamma(\lambda_j^{(k)})\right|^2 = 1 \tag{5.119}$$

Using this fact we can reason that if $\tilde{p}\left(\cos\left(\pi\lambda_j^{(k)}\right)\right) \leq \delta_{\text{amp}} + \delta_{\text{svt}}$ then $|\gamma(\lambda_j^{(k)})|^2 \geq 1 - (\delta_{\text{amp}} + \delta_{\text{svt}})$.

Similarly, if $\tilde{p}\left(\cos\left(\pi\lambda_j^{(k)}\right)\right) \geq 1 - \delta_{\text{amp}} - \delta_{\text{svt}}$, then:

$$|\gamma(\lambda_j^{(k)})|^2 \leq 1 - (1 - (\delta_{\text{amp}} + \delta_{\text{svt}}))^2 \leq 2(\delta_{\text{amp}} + \delta_{\text{svt}}) - (\delta_{\text{amp}} + \delta_{\text{svt}})^2 \tag{5.120}$$

Now that we have bounds on the amplitudes of the output state, we can bound its distance to $|\text{bit}_k(\lambda_j)\rangle$ for a favorable choice of $e^{i\varphi_j}$. Say $\text{bit}_k(\lambda_j) = 0$. Then we select

$\varphi_j = 0$, so that:

$$\left|\left(\tilde{p}\left(\cos\left(\pi\lambda_j^{(k)}\right)\right)|0\rangle + \gamma(\lambda_j^{(k)})\,|1\rangle\right) - e^{i\varphi_j}\,|\mathrm{bit}_k(\lambda_j)\rangle\right| \tag{5.121}$$

$$\leq \sqrt{\left|\tilde{p}\left(\cos\left(\pi\lambda_j^{(k)}\right)\right) - 1\right|^2 + |\gamma(\lambda_\Delta)|^2} \tag{5.122}$$

$$\leq \sqrt{(\delta_{\text{amp}} + \delta_{\text{svt}})^2 + 2(\delta_{\text{amp}} + \delta_{\text{svt}}) - (\delta_{\text{amp}} + \delta_{\text{svt}})^2} \tag{5.123}$$

$$\leq \sqrt{2(\delta_{\text{amp}} + \delta_{\text{svt}})} \tag{5.124}$$

Otherwise, if $\mathrm{bit}_k(\lambda_j) = 1$, then we define φ_j by $\gamma(\lambda_j^{(k)}) = e^{i\varphi_j}|\gamma(\lambda_j^{(k)})|$. That way:

$$\left|\left(\tilde{p}\left(\cos\left(\pi\lambda_j^{(k)}\right)\right)|0\rangle + \gamma(\lambda_j^{(k)})\,|1\rangle\right) - e^{i\varphi_j}\,|\mathrm{bit}_k(\lambda_j)\rangle\right| \tag{5.125}$$

$$\leq \sqrt{\left|\tilde{p}\left(\cos\left(\pi\lambda_j^{(k)}\right)\right)\right|^2 + \left|e^{i\varphi_j}(|\gamma(\lambda_j^{(k)})| - 1)\right|^2} \tag{5.126}$$

$$\leq \sqrt{|\delta_{\text{amp}} + \delta_{\text{svt}}|^2 + |\delta_{\text{amp}} + \delta_{\text{svt}}|^2} \tag{5.127}$$

$$\leq \sqrt{2(\delta_{\text{amp}} + \delta_{\text{svt}})} \tag{5.128}$$

So either way, the output state is within $\sqrt{2(\delta_{\text{amp}} + \delta_{\text{svt}})}$ in spectral norm of that of the ideal state. Thus, the unitary map U_{svt} is close in spectral norm to some ideal unitary. Invoking Lemma 5.4, the distance in diamond norm is at most:

$$2\sqrt{2(\delta_{\text{amp}} + \delta_{\text{svt}})} \leq \delta \tag{5.129}$$

The inequality above holds if we select, for any $m > 0$:

$$\delta_{\text{amp}} := (1 - 10^{-m}) \cdot \frac{\delta^2}{8} \qquad \delta_{\text{svt}} := 10^{-m} \cdot \frac{\delta^2}{8} \tag{5.130}$$

This solution to the inequality relies on the fact that only δ_{amp} actually enters the query complexity, so if classical resources are cheap but query complexity is expensive then we can make the classical computer do as much work as possible by making m larger.

Finally, we compute the query complexity. An invocation of $e^{2\pi i \hat{\lambda}_j^{(k)}}$ requires 2^{n-k-1} invocations of $U = e^{2\pi i \hat{\lambda}}$. By Theorem 3.17, the number of queries made by the unitary U_{svt} to U_{signal} is the degree of the polynomial. By Corollary 3.23, the polynomial $A_{\eta\to\delta_{\text{amp}}}(x^2)$ has degree $2 \cdot M_{\eta\to\delta}$, so the query complexity is:

$$2^{n-k-1} \cdot 2 \cdot M_{\eta\to\delta_{\text{amp}}} \tag{5.131}$$

□

A really nice feature of the coherent iterative phase estimator we present is that it produces no garbage qubits. All singular value transformation is performed on the final output qubit. It does still produce extra phase shifts between the eigenstates, which in some applications may still need to be be uncomputed. However, in applications where phase differences between eigenstates do not matter, like thermal state preparation, we expect that this uncomputation step can be skipped.

To finish the discussion of coherent iterative phase estimation, we stitch the iterative estimator we just defined into a regular phase estimator. In doing so, we also remark on what happens when no rounding promise is present. A summary of the construction is presented in Figure 5.4.

Corollary 5.10. ***Improved phase estimation.*** *The coherent iterative phase estimator from Theorem 5.9 can be combined with Lemma 5.7 to make a phase estimator with phases with query complexity at most:*

$$O\left(2^n \alpha^{-1} \log(\delta^{-1})\right) \tag{5.132}$$

assuming α is bounded away from 1 by a constant.

Furthermore, if no rounding promise is given, then the estimator δ-approximates in diamond norm a map:

$$|0^n\rangle |\psi_j\rangle \to \left(\xi \,|\mathit{floor}(2^n\lambda_j)\rangle + \zeta\, |\lambda'_j\rangle\right) |\psi_j\rangle \tag{5.133}$$

for some complex amplitudes ξ, ζ and $\lambda'_j = \mathit{floor}(2^n\lambda_j) - 1 \bmod 2^n$ is an erroneous estimate. The performance is the same, except that $0 < \alpha < 1$ can be any constant.

Proof. Write $\eta_k := \frac{1}{2} - \frac{1}{2^k}\left(\frac{1}{2} + \frac{\alpha}{2}\right)$ if $k \geq 1$ and $\eta_0 := \frac{\alpha}{2}$ if $k = 0$, and, following Lemma 5.7, demand an accuracy of $\delta_{\text{amp},k} := (1 - 10^{-m})(\delta 2^{-k-1})^2/8$ for the k'th bit. Recall from Corollary 3.23 that $M_{\eta_k \to \delta_{\text{amp}}} \in O\left(\eta_k^{-1} \log(\delta_{\text{amp}}^{-1})\right)$. Then, the overall query complexity is:

$$\sum_{k=0}^{n-1} 2^{n-k} \cdot M_{\eta_k \to \delta_{\text{amp},k}} \in O\left(\sum_{k=0}^{n-1} 2^{n-k} \eta_k^{-1} \log(\delta_{\text{amp},k}^{-1})\right) \tag{5.134}$$

$$= O\left(\sum_{k=0}^{n-1} 2^{n-k} \eta_k^{-1} \log(2^{k+1}\delta^{-1})\right) \tag{5.135}$$

When $k = 0$ we have $\eta_0 = \frac{\alpha}{2}$, and when $k > 0$ we have $\eta_k > \frac{1-\alpha}{4}$ which is bounded from below by a constant. We can use this to split the sum:

$$\leq O\left(2^{n-0}\eta_0^{-1}\log(2^{0+1}\delta^{-1}) + \sum_{k=1}^{n-1} 2^{n-k}\eta_k^{-1}\log(2^{k+1}\delta^{-1})\right) \tag{5.136}$$

$$\leq O\left(2^n\alpha^{-1}\log(\delta^{-1}) + \sum_{k=1}^{n-1} 2^k(k + \log(\delta^{-1}))\right) \tag{5.137}$$

$$\leq O\left(2^n\alpha^{-1}\log(\delta^{-1}) + 2(2^n - n - 1) + (2^n - 2)\log(\delta^{-1})\right) \tag{5.138}$$

$$\leq O\left(2^n\alpha^{-1}\log(\delta^{-1})\right) \tag{5.139}$$

This completes the runtime analysis. Now we turn to the case when no rounding promise is present. Notice that when $k \geq 1$, the regions where λ_j is assumed not to appear are guaranteed by the previous estimators *definitely* outputting 1 for previous bits regardless of the promise. Thus, the only bit that can be wrong is the first bit. When an eigenvalue λ_j falls into a region disallowed by the rounding promise, the first bit will be some superposition $\xi |0\rangle + \zeta |1\rangle$.

Flipping the final bit of an estimate in general results in an error of ± 1. However, recall that when estimating future bits the value of $\Delta_k/2^n$ is subtracted off from λ_j. That means that when we erroneously measure a 1, the rest of the algorithm proceeds to measure $\lambda_j - 1/2^n$ instead. As a result, if the first bit is wrong, then the algorithm will output $\text{floor}(2^n\lambda_j) - 1$ instead. Since the algorithm is periodic in λ_j with period 1, the output will be $2^n - 1$ if an error occurs when $\text{floor}(2^n\lambda_j) = 0$.

$\square$

Notice that if we had allocated the error evenly between the n steps, then we would have incurred an extra $O(2^n \log n)$ term in the above. Spreading the error via a geometric series avoids this, and we find that we obtain better constant factors with this choice as well. This is because the $k = 0$ term dominates, so we want to make $\delta_{\text{amp},k}$ as large as possible.

5.2.2 Coherent Iterative Energy Estimation

While for phase estimation we are given access to $\sum_j e^{2\pi i \lambda_j} |\psi_j\rangle \langle\psi_j|$, for energy estimation we have a block encoding of $\sum_j \lambda_j |\psi_j\rangle \langle\psi_j|$. For phase estimation we could relatively easily synthesize a cosine in the eigenvalues, just by taking a linear combination with the identity. But for energy estimation we must build the cosine directly.

To do so we leverage a tool employed by [GSLW18] to perform Hamiltonian simulation. The Jacobi-Anger expansion yields highly efficient polynomial approximations to

$\sin(tx)$ and $\cos(tx)$. To perform Hamiltonian simulation one then takes the linear combination $\cos(tx) + i\sin(tx)$. However, we only need the cosine component.

Lemma 5.11. ***Jacobi-Anger expansion.*** *For any $t \in \mathbb{R}^+$, and any $\varepsilon \in (0, 1/e)$, let:*

$$r(t', \varepsilon') := \text{ the solution to } \varepsilon' = \left(\frac{t'}{r}\right)^r \text{ such that } r \in (t', \infty), \tag{5.140}$$

$$R := \left\lfloor r\left(\frac{et}{2}, \frac{5}{4}\varepsilon\right)/2\right\rfloor \tag{5.141}$$

$$J_k(t) := \text{the } k\text{'th Bessel function of the first kind} \tag{5.142}$$

$$T_k(t) := \text{the } k\text{'th Chebyshev polynomial of the first kind} \tag{5.143}$$

$$p_{cos,t}(x) := J_0(t) + 2\sum_{k=1}^{R}(-1)^k J_{2k}(t)T_{2k}(x) \tag{5.144}$$

Then $p_{cos,t}(x)$ is an even polynomial of degree $2R$ such that for all $x \in [-1, 1]$:

$$|\cos(tx) - p_{cos,t}(x)| \le \varepsilon. \tag{5.145}$$

Furthermore:

$$r(t', \varepsilon') \in \Theta\left(t' + \frac{\log \varepsilon'^{-1}}{\log(\log(\varepsilon'^{-1}))}\right) \tag{5.146}$$

Proof. These results are shown in Lemma 57 and Lemma 59 of [GSLW18], outlined in their section 5.1. $\square$

Again, computation of the degree of the Jacobi-Anger expansion is a bit complicated, but the complexity is worth it due to the method's high performance. Our approach is to first synthesize $\cos\left(\pi\lambda_j^{(k)}\right)$ to obtain a signal that oscillates to indicate $\text{bit}_k(\lambda_j)$, and then apply $A_{\eta\to\delta}(x^2)$ to amplify the signal to 0 or 1, as shown in Figure 5.2.

Actually, one might observe that synthesizing $\cos\left(\pi\lambda_j^{(k)}\right)$ first is not necessary to make polynomials that look like those in Figure 5.2. Instead one can take an approach similar to the one used for making rectangle functions in [LC17]: simply shift, scale and add several amplifying polynomials $A_{\eta\to\delta}(x)$ to make the desired shape. None of these operations affect the degree, so this approach also yields the same asymptotic scaling $O(2^n\alpha^{-1}\log(\delta^{-1}))$ as phase estimation. We will see in Corollary 5.13 that the method using the Jacobi-Anger expansion actually achieves the worse scaling of $O(\alpha^{-1}\log(\delta^{-1})(2^n+\log(\alpha^{-1})))$.

The reason why we present the approach using the Jacobi-Anger expansion, despite it having worse asymptotic scaling, is that in the regime of interest ($n\approx 10, \alpha\approx 2^{-10}$) we numerically find that the Jacobi-Anger expansion actually performs better. There may be a regime where it is better to remove the Jacobi-Anger expansion from the construction, in which case the algorithm is easily adapted.

The rest of the construction of Theorem 5.12 strongly resembles coherent iterative phase estimation, so much so that we can re-use parts of the proof of Theorem 5.9. One further difference is that this estimator now has garbage, because we have no guarantee that the block encoding of the Hamiltonian only has one ancilla.

Theorem 5.12. ***Coherent Iterative Energy Estimation.*** *Say the block encoding of H requires a ancillae, that is, U_H acts on $\mathbb{C}^{2^a}\otimes\mathcal{H}$. Then there is an iterative energy estimator with phases and $a+n+3$ qubits of garbage with query complexity:*

$$4\cdot M_{(1-10^{-m_{cos}})\eta_k\to\delta_{amp}}\cdot\left\lfloor r\left(\frac{e}{2}\pi 2^{n-k},\frac{5}{4}\frac{\eta_k}{2}10^{-m_{cos}}\right)\right\rfloor \tag{5.147}$$

where $M_{\eta\to\delta}$ and η_k are as in Corollary 3.23 and δ_{amp} can be chosen to be $(1-10^{-m_{svt}})\delta^2/8$ for any $m_{svt}>0$, and we can choose any $m_{cos}>0$.

Proof. We construct the estimator as follows:

1. Let W_k be a hermitian matrix on k qubits defined by:

$$W_k := 2 \sum_{\Delta_k=0}^{2^k-1} \frac{\Delta_k}{2^n} |\Delta_k\rangle \langle \Delta_k| \tag{5.148}$$

W_k has a block encoding which can be constructed as follows: First, prepare $|+^{n-1}\rangle$:

$$|\Delta_k\rangle \to |\Delta_k\rangle |+^{n-1}\rangle = \frac{1}{\sqrt{2^{n-1}}} \sum_{x=0}^{2^{n-1}-1} |\Delta_k\rangle |x\rangle \tag{5.149}$$

Next, observe that $\Delta_k \leq 2^k - 1 \leq 2^{n-1} - 1$. Into an ancilla register compute $|x < \Delta_k\rangle$, and uncompute any garbage necessary to do so.

$$\to \frac{1}{\sqrt{2^{n-1}}} \sum_{x=0}^{2^{n-1}-1} |\Delta_k\rangle |x\rangle |x < \Delta_k\rangle \tag{5.150}$$

Then postselect that the final register is in the $|1\rangle$ state:

$$\to \frac{1}{\sqrt{2^{n-1}}} |\Delta_k\rangle \sum_{x=0}^{\Delta_k-1} |x\rangle \tag{5.151}$$

Finally, postselect that the x register is in the $|+^{n-1}\rangle$ state:

$$\to |\Delta_k\rangle \sum_{x=0}^{\Delta_k-1} \frac{1}{2^{n-1}} = \frac{\Delta_k}{2^{n-1}} |\Delta_k\rangle \tag{5.152}$$

This process makes use of $n-1$ ancillae initialized and postselected to $|+\rangle$, and one more ancilla that is postselected to $|1\rangle$.

2. Use linear combinations of unitaries to construct a block encoding of:

$$H^{(k)} := \frac{1}{2} \cdot I \otimes H - \frac{1}{4} \cdot W_k \otimes I + \frac{1}{4} \cdot (4\phi_k 2^{k-n}) \cdot I \otimes I \tag{5.153}$$

where the ϕ_k are selected just as in as in Theorem 5.9. Observe that $\phi_k < 1/2$, so therefore $4\phi_k 2^{k-n}$ is a probability which can be block-encoded. Since the block encoding of H has a ancillae, and that of W_k has n ancillae, and the three terms in the linear combination need two control qubits, the block encoding of $H^{(k)}$ has $a+n+2$ ancillae.

3. Use Lemma 5.11 to construct a polynomial approximation $p_{\cos,\pi 2^{n-k}}(x)$ of $\cos(\pi 2^{n-k} x)$ to accuracy $2\delta_{\cos}$, to be picked later.

   Use Corollary 3.23 to construct a polynomial $A_{(\eta-\delta_{\cos})\to\delta_{\text{amp}}}$. We will pick δ_{amp} later and select η_k just as in Theorem 5.9.

   Finally, use Theorem 3.17 to construct U_{svt}, a block encoding of $\tilde{p}(H^{(k)})$ which approximates the even polynomial:

$$\tilde{p}(x) \approx A_{(\eta_k-\delta_{\cos})\to\delta_{\text{amp}}}\left(p^2_{\cos,\pi 2^{n-k}}(x)\right) \tag{5.154}$$

   To perform singular value transformation we needed one extra ancilla, so U_{svt} has $a+n+3$ ancillae - these are the garbage output of this map.

4. Use the modified Toffoli gate $I \otimes |0\rangle\langle 0| + X \otimes (I - |0\rangle\langle 0|)$ to conditionally flip the output qubit.

We rewrite $H^{(k)}$ in terms of its eigendecomposition:

$$H^{(k)} = \sum_j \sum_{\Delta_k} \left[\frac{\lambda_j}{2} - \frac{1}{2}\frac{\Delta_k}{2^n} + 2^{k-n}\phi_k\right] |\Delta_k\rangle\langle\Delta_k| \otimes |\psi_j\rangle\langle\psi_j| \tag{5.155}$$

$$= \sum_j \sum_{\Delta_k} 2^{k-n}\lambda_j^{(k)} |\Delta_k\rangle\langle\Delta_k| \otimes |\psi_j\rangle\langle\psi_j| \tag{5.156}$$

where we defined $\lambda_j^{(k)} := 2^{n-k-1}\left(\lambda_j - \frac{\Delta_k}{2^n}\right) + \phi_k$ just like in Theorem 5.9. This protocol implements a map:

$$|0\rangle|0...0\rangle|\Delta_k\rangle|\psi_j\rangle \to \left(\tilde{p}\left(2^{k-n}\lambda_j^k\right)|0\rangle|\text{gar}_{0,j}\rangle + \gamma(\lambda_j^k|1\rangle|\text{gar}_{1,j}\rangle\right)|\Delta_k\rangle|\psi_j\rangle \tag{5.157}$$

Here $\gamma(\lambda_j^{(k)})$ is defined such that all the other amplitudes for the failed branches of the block encoding are absorbed into the normalized state $|\text{gar}_{1,j}\rangle$.

We see that $\tilde{p}\left(2^{k-n}\lambda_j^k\right)$ approximates

$$\tilde{p}(\lambda_\Delta/2) \approx A_{(\eta_k-\delta_{\cos})\to\delta_{\text{amp}}}\left(\cos^2\left(\pi\lambda_j^{(k)}\right)\right) \tag{5.158}$$

which is the same expression encountered in Theorem 5.9, with the same definition of $\lambda_j^{(k)}$, up to a minor shift on η_k. Therefore, we can follow the same reasoning as in Theorem 5.9 up to two minor differences, and arrive at the exact same conclusion. Namely, if we select some $m_{\text{svt}} > 0$ and then let:

$$\delta_{\text{amp}} := (1-10^{-m_{\text{svt}}})\frac{\delta^2}{8}, \qquad \delta_{\text{svt}} := 10^{-m_{\text{svt}}}\frac{\delta^2}{8}, \tag{5.159}$$

Then the unitary channel we implement is at most δ-far in diamond norm to a channel that implements the map:

$$|0\rangle\,|0...0\rangle\,|\Delta_k\rangle\,|\psi_j\rangle \to e^{i\varphi_j}\,|\text{bit}_k(\lambda_j)\rangle\,|\text{gar}_{\lambda_j}\rangle\,|\Delta_k\rangle\,|\psi_j\rangle \tag{5.160}$$

whenever Δ_k encodes the last k bits of λ_j.

Since the argument in Theorem 5.9 is lengthy and the modifications are extremely minor, we will not repeat the argument here. Instead we will articulate the two things that change.

First, Theorem 5.9 gives an estimator with phases and no garbage, whereas here we also have garbage. The garbage just tags along for the entire calculation, and when we come to selecting φ_j depending on λ_j we can also select $|\text{gar}_{\lambda_j}\rangle = |\text{gar}_{0/1,j}\rangle$ depending on $\text{bit}_k(\lambda_j)$.

Second, we incur an error of $\delta_{\cos}$ in the approximation of $\cos(\pi 2^{n-k}x)$ with $p_{\cos,\pi 2^{n-k}}(x)$. Since we show that $\cos^2(\pi 2^{n-k}x)$ is bounded away from $\frac{1}{2}$ by $\pm\eta$ we therefore have that $p_{\cos,\pi 2^{n-k}}(x)^2$ is bounded away from $\frac{1}{2}$ by $\pm\left(\eta - \frac{2\delta_{\cos}}{2}\right)$. The amplifying polynomial then proceeds to amplify $(\eta - \delta_{\text{cos}}) \to \delta_{\text{amp}}$ appropriately, so the calculation proceeds the same. We just need to ensure that $\eta - \delta_{\text{cos}} > 0$, so we select

$$\delta_{\text{cos}} := 10^{-m_{\text{cos}}} \cdot \eta \tag{5.161}$$

for some $m_{\text{cos}} > 0$. This completes the accuracy analysis.

Finally, we analyze the query complexity. The block encoding of $H^{(k)}$ makes one query to U_H, so by Theorem 3.17 the query complexity is exactly the degree of $\tilde{p}(x)$ which is the degree of $A_{(\eta-\delta_{\text{cos}})\to\delta_{\text{amp}}}\left(p^2_{\text{cos},\pi 2^{n-k}}(x)\right)$. From Lemma 5.11 and Corollary 3.23, the degree is:

$$M_{(\eta-\delta_{\text{cos}})\to\delta_{\text{amp}}} \cdot 2 \cdot 2 \left\lfloor r\left(\frac{e}{2}\pi 2^{n-k}, \frac{5}{4}\frac{\delta_{\text{cos}}}{2}\right)\right\rfloor \tag{5.162}$$

Substituting the definitions for $\eta, \delta_{\text{amp}}$ and δ_{cos} yields the final runtime. As with Theorem 5.9, δ_{amp} can be made larger by increasing m_{amp}. By increasing m_{cos} we can decrease δ_{cos}, which decreases the degree of $A_{(\eta-\delta_{\text{cos}})\to\delta_{\text{amp}}}(x)$ while increasing the degree of $p_{\text{cos},\pi 2^{n-k}}(x)$. The Jacobi-Anger expansion deals with error more efficiently than the amplifying polynomial, so in practice m_{cos} should be quite large. □

As with phase estimation, we also pack the coherent iterative energy estimator into a regular energy estimator using Lemma 5.7. This time, since the iterative estimator produces garbage it makes sense to uncompute the garbage using Lemma 5.8.

Corollary 5.13. ***Improved Energy Estimation.*** *The iterative energy estimator with phases and garbage from Theorem 5.12 can be combined with Lemma 5.8 and Lemma 5.7 to make an energy estimator without phases and without garbage with query complexity bounded* *by:*

$$O\left(\alpha^{-1}\log(\delta^{-1})\left(2^n+\log\left(\alpha^{-1}\right)\right)\right) \tag{5.163}$$

assuming that α is bounded away from 1 by a constant.

Furthermore, even when there is no rounding promise, there exists an algorithm that, given an eigenstate $|\psi_j\rangle$ of the Hamiltonian λ_j, for any $\delta > 0$ performs a transformation δ-close in diamond norm to the map:

$$|\psi_j\rangle\langle\psi_j| \to \left(p\,|\mathit{floor}(2^n\lambda_j)\rangle\langle\mathit{floor}(2^n\lambda_j)| + (1-p)\,|\lambda'_j\rangle\langle\lambda'_j|\right)|\psi_j\rangle\langle\psi_j| \tag{5.164}$$

where p is some probability and $\lambda'_j = \mathit{floor}(2^n\lambda_j) - 1$ mod 2^n is an erroneous estimate. Just as in Corollary 5.10, the performance is the same except that $0 < \alpha < 1$ can be any constant.

Proof. As with Corollary 5.10, we write η_k to make the k dependence explicit and demand accuracy $\delta_{\text{amp},k} = (1-10^{-m_{\text{amp}}})(\delta 2^{-k-1})^2/8$ for the k'th bit. From Lemma 5.11 we obtain an asymptotic upper bound $r(t,\varepsilon) \in O\left(t+\log(\varepsilon^{-1})\right)$. Again, recall from Corollary 3.23 that $M_{\eta_k\to\delta_{\text{amp}}} \in O\left(\eta_k^{-1}\log(\delta_{\text{amp}}^{-1})\right)$. The asymptotic query complexity of the iterative energy estimator from Theorem 5.12 is then bounded by:

$$O\left((\eta_k-\delta_{\cos})^{-1}\log(\delta_{\text{amp},k}^{-1})\cdot(2^{n-k}+\log(\delta_{\cos,k}))\right) \tag{5.165}$$

$$\le O\left((1-10^{-m_{\cos}})^{-1}\eta_k^{-1}\log((1-10^{-m_{\text{amp}}})^{-1}2^{2(k+1)}\delta^{-2}8)\cdot(2^{n-k}+\log(10^{m_{\cos}}\eta_k^{-1}))\right) \tag{5.166}$$

$$\le O\left(\eta_k^{-1}\log(2^{k+1}\delta^{-1})\cdot(2^{n-k}+\log(\eta_k^{-1}))\right) \tag{5.167}$$

Next we invoke Lemma 5.8 to remove the phases and the garbage, doubling the query complexity. We do this before invoking Lemma 5.7, because Lemma 5.7 involves blowing up the number of garbage registers by a factor of n. While we could also invoke Lemma 5.7 and then invoke Lemma 5.3 to obtain a map without garbage, this would involve many garbage registers sitting around waiting to be uncomputed for a long time. If we invoke Lemma 5.8 first we get rid of the garbage immediately.

Finally, we invoke Lemma 5.7 to turn our iterative energy estimator without garbage and phases into a regular energy estimator without garbage and phases. As in Corollary 5.10, we observe that $\eta_0 = \alpha/2$ and for $k > 0$ we have η_k bounded from below by a constant. Then, the total query complexity is:

$$O\left(\eta_0^{-1}\log(2^{0+1}\delta^{-1})\left(2^{n-0}+\log\left(\eta_0^{-1}\right)\right)+\sum_{k=1}^{n-1}\eta_k^{-1}\log(2^{k+1}\delta^{-1})\left(2^{n-k}+\log\left(\eta_k^{-1}\right)\right)\right) \tag{5.168}$$

$$\leq O\left(\alpha^{-1}\log(\delta^{-1})\left(2^{n}+\log\left(\alpha^{-1}\right)\right)+\sum_{k=1}^{n-1}(k+\log(\delta^{-1}))2^{n-k}\right) \tag{5.169}$$

$$\leq O\left(\alpha^{-1}\log(\delta^{-1})\left(2^{n}+\log\left(\alpha^{-1}\right)\right)+2(2^n-n-1)+(2^n-2)\log(\delta^{-1})\right) \tag{5.170}$$

$$\leq O\left(\alpha^{-1}\log(\delta^{-1})\left(2^{n}+\log\left(\alpha^{-1}\right)\right)\right) \tag{5.171}$$

Next, we show that it is possible to implement a map that, given an eigenstate $|\phi_j\rangle$, measures an estimate that is either floor$(2^n\lambda_j)$ or floor$(2^n\lambda_j) - 1 \bmod 2^n$ with some probability. Note that this is not the same algorithm as above. Just as with phase estimation, it is only the first bit that actually needs the rounding promise, and all other bits are guaranteed to be deterministic.

The first bit performs a map of the form:

$$|0^n\rangle|0...0\rangle|\psi_j\rangle \to \left(\sqrt{p}\,|0\rangle|\mathrm{gar}_{0,j}\rangle + \sqrt{1-p}\,|1\rangle|\mathrm{gar}_{1,j}\rangle\right)|\psi_j\rangle \tag{5.172}$$

We immediately see that Lemma 5.8 cannot be used to perform uncomputation here, because uncomputation only works when $p = 1$ or $p = 0$. Instead, we simply measure the output register containing $|0\rangle$ or $|1\rangle$ and discard the garbage. This would damage any superposition over the $|\psi_j\rangle$, which is why this algorithm only works if the input is an eigenstate.

We then use iterative estimators for the remaining bits to compute the rest of the estimate. This time their outputs will be deterministic, but there is no point in doing uncomputation since the superposition has already collapsed. Instead, we do the same thing as for the $k = 0$ estimator: compute the next bit and some garbage, and discard the garbage. The final answer is either $\text{floor}(2^n\lambda_j)$ or $\text{floor}(2^n\lambda_j) - 1$ for the same reason as in Corollary 5.10.

$\square$

5.3 Performance Comparison

Above, we have presented a modular framework for phase and energy estimation using the key ingredients in Theorem 5.9 and Theorem 5.12 respectively. These results already demonstrate several advantages over textbook phase estimation as presented in Proposition 5.5.

First, they eliminate the QFT and do not require a sorting network to perform median amplification. Instead, they rely on just a single tool: singular value transformation.

Second, they require far fewer ancillae. Our improved phase estimation algorithm requires no ancillae at all, and is merely 'with phases' so arguably does not even need uncomputation for some applications. On the other hand, textbook phase estimation requires $O((n + \log(\alpha^{-1}))\log(\delta^{-1}))$ ancillae in order to implement median amplification.

Given a block encoding of a Hamiltonian with a ancillae, then our energy estimation algorithm requires $a + n + 3$ ancillae. But in order to even compare Proposition 5.5 to Theorem 5.12 we need a method to perform energy estimation using textbook phase estimation. This is achieved through Hamiltonian simulation.

Which method of Hamiltonian simulation is best depends on the particular physical system involved. Hamiltonian simulation using the Trotter approximation can perform exceedingly well in many situations [SHC20]. However, in our analysis we must be agnostic to the particular Hamiltonian in question, and furthermore need a unified method for comparing the performance. Hamiltonian simulation via singular value transformation [LC17, GSLW18], lets us compare Proposition 5.5 and Theorem 5.12 on the same footing. After all, this method features the best known asymptotic performance in terms of the simulation time [Childs&19] in a black-box setting.

Singular value transformation constructs an approximate block encoding of e^{iHt} with ancillae. Since e^{iHt} is unitary, in the ideal case the ancillae start in the $|0\rangle$ state and are also guaranteed to be mapped back to the $|0\rangle$ state. But in the approximate case the ancillae are still a little entangled with the remaining registers, so they become an additional source of error. Certainly the ancillae cannot be re-used to perform singular value transformation again, because then the errors pile up with each use. Thus, we are in a similar situation to Lemma 5.3, where we must discard some qubits and take into account the error.

Therefore, we must do some additional work beyond the Hamiltonian simulation method presented in [GSLW18], to turn the approximate block encoding of a unitary into a channel that approximates the unitary in diamond norm. The main trick for this proof is to consider postselection of the ancilla qubits onto the $|0\rangle$ state. Then the error splits into two parts: the error of the channel when the postselection succeeds, and the probability that the postselection fails.

The Hamiltonian simulation method is extremely accurate, letting us obtain block encodings with an error that decays exponentially in the query complexity to U_H. Thus, the error contribution of this channel is almost entirely negligible in the performance analysis. The purpose of the argument below is really to provide a Lemma analogous to Lemma 5.4 that lets us convert error bounds in spectral norms on block encodings to diamond norms. That way, our entire error analysis is completely formal, as it should be for a fair comparison of all algorithms involved. Notably, an ε-accurate block encoding of a unitary is not the same thing as an ε-accurate implementation of that unitary: the latter implies a channel with diamond norm error 2ε while the prior yields an error 4ε due to the ancilla registers.

Lemma 5.14. ***Hamiltonian simulation.*** *Say U_H is a block encoding of a Hamiltonian H. Then, for any $t > 0$ and $\varepsilon > 0$ there exists a quantum channel that is ε-close in diamond norm to the channel induced by the unitary e^{iHt}. This channel can be implemented using*

$$3 \cdot r\left(\frac{et}{2}, \frac{\varepsilon}{24}\right) + 3 \tag{5.173}$$

queries to controlled-U_H or controlled-$U_H^\dagger$.

Proof. This is an extension of Theorem 58 of [GSLW18], which states that there exists a block encoding U_A of a matrix A such that $|A - e^{iHt}| \leq \varepsilon$ with query complexity $3r\left(\frac{et}{2}, \frac{\varepsilon}{6}\right)$. This result leverages the Jacobi-Anger expansion (Lemma 5.11) to construct approximate block encodings of $\sin(tH)$ and $\cos(tH)$ and uses linear combinations of unitaries to approximate $e^{iHt}/2$. Then it uses oblivious amplitude amplification to get rid of the factor of $1/2$, obtaining U_A. If U_H is a block encoding with a ancillae, then U_A has $a + 2$ ancillae.

All that is left to do is to turn this block encoding into a quantum channel that approximately implements e^{iHt}. To do so, we just initialize the ancillae to $|0^{a+2}\rangle$, apply U_A, and trace out the ancillae. We can write this channel Λ as:

$$\Lambda(\rho) := \sum_i (\langle i| \otimes I) U_A (|0^{a+2}\rangle \langle 0^{a+2}| \otimes \rho) U_A^\dagger (|i\rangle \otimes I) \tag{5.174}$$

To finish the theorem we must select an ε so that the error in diamond norm of Λ to the channel implemented by e^{iHt} is bounded by δ. To do so, we write Λ as a sum of postselective channels $\Lambda_i(\rho)$:

$$\Lambda_i(\rho) := (\langle i| \otimes I) U_A (|0\rangle \langle 0| \otimes \rho) U_A^\dagger (|i\rangle \otimes I) \tag{5.175}$$

That way $\Lambda = \sum_i \Lambda_i$. If we let $\Gamma_{e^{iHt}} := e^{iHt} \rho e^{-iHt}$, then we can bound the error in diamond norm using the triangle inequality:

$$|\Lambda - \Gamma_{e^{iHt}}|_\diamond \leq |\Lambda_0 - \Gamma_{e^{iHt}}|_\diamond + \left| \sum_{i>0} \Lambda_i \right|_\diamond \tag{5.176}$$

Now we proceed to bound the two terms individually. Observe that:

$$\Lambda_0(\rho) := (\langle 0| \otimes I) U_A (|0\rangle \langle 0| \otimes \rho) U_A^\dagger (|0\rangle \otimes I) = A \rho A^\dagger \tag{5.177}$$

Since $|A - e^{iHt}| \leq \varepsilon$, we can invoke Lemma 5.4 and see that $|\Lambda_0 - \Gamma_{e^{iHt}}|_\diamond \leq 2\varepsilon$, and we are done with the first term.

To bound $\left|\sum_{i>0} \Lambda_i\right|_\diamond$, we first observe that $\sum_{i>0} \Lambda_i = \Lambda - \Lambda_0$. Second, we observe that the Λ_i are all positive semi-definite, so $|\Lambda_i(\rho)|_1 = \mathrm{Tr}(\Lambda_i(\rho))$. Plugging in the definition of the diamond norm, we can compute:

$$\left|\sum_{i>0}\Lambda_i\right|_\diamond = \sup_\rho \left|\sum_i (\Lambda_i \otimes \mathcal{I})(\rho)\right|_1 \tag{5.178}$$

$$\leq \sup_\rho \operatorname{Tr}\left(\sum_i (\Lambda_i \otimes \mathcal{I})(\rho)\right) \tag{5.179}$$

$$= \sup_\rho \operatorname{Tr}\left((\Lambda \otimes \mathcal{I})(\rho) - (\Lambda_0 \otimes \mathcal{I})(\rho)\right) \tag{5.180}$$

$$= 1 - \inf_\rho \operatorname{Tr}(A\rho A^\dagger) \tag{5.181}$$

If we let $E := e^{iHt} - A$ so that $|E| \leq \varepsilon$, and plug into the above expression, we get:

$$\operatorname{Tr}(A\rho A^\dagger) = \operatorname{Tr}(e^{iHt}\rho e^{-iHt} - E\rho e^{-iHt} - e^{iHt}\rho E^\dagger + E\rho E^\dagger) \geq 1 - 2\varepsilon + \varepsilon^2 \tag{5.182}$$

Putting everything together we obtain $|\Lambda - \Gamma_{e^{iHt}}|_\diamond \leq 2\varepsilon + 2\varepsilon - \varepsilon^2 \leq 4\varepsilon$. So if we select $\varepsilon := \delta/4$ we obtain the desired bound.

□

The above proof uses the trick for the proof of the block-measurement theorem that we mentioned in the introduction. We will need it again in the proof Theorem 5.16. Thus, while we are at it, we may as well state the generalization of this result to any block-encoded unitary as a proposition.

Proposition 5.15. *Say U_A is a block encoding of A, which is ε-close in spectral norm to a unitary V. Then there exists a quantum channel 4ε-close in diamond norm to the channel $\rho \to V\rho V^\dagger$.*

Proof. Lemma 5.14 proved this result with $V = e^{iHt}$. The exact same argument holds for abstract V. □

Returning to the ancilla discussion, say that the block encoding of the input Hamiltonian has a ancillae. Then we see that Theorem 5.12 requires $a+n+3$ ancillae, while textbook phase estimation combined with Lemma 5.14 requires $O(a+(n+\log(\alpha^{-1}))\log(\delta^{-1}))$. This is because the extra ancillae required to perform the procedure in Lemma 5.14 can be discarded and reset after each application of e^{iHt}. Also note that despite the fact that the above implementation of e^{iHt} is not unitary, the overall protocol in Proposition 5.5 is still approximately invertible as required by Lemma 5.3, since we can just use Lemma 5.14 to implement e^{-iHt} instead.

Next, we compare the algorithms in terms of their query complexity to the unitary U or the block encoding U_H. Note that this is *not* the gate complexity: the number of gates from some universal gate set used to implement the algorithm. The reason for this choice is that the gate complexity of U or U_H depends on the application, and is likely much much larger than any of the additional gates used to implement singular value transformation. The algorithms' query complexities depend on three parameters: α, n, and δ. In the following we discuss the impact of these parameters on the complexity and show how we arrive at the 14x and 20x speedups stated in the introduction.

The performance in terms of α is plotted in Figure 5.5, for fixed $n = 10$ and $\delta = 10^{-30}$. This comparison shows several features. First, we see that the novel algorithms in this chapter are consistently faster than the traditional methods in this regime. The only exception is Corollary 5.10 combined with Lemma 5.3 to remove the phases, which is outperformed by traditional phase estimation for some value of $\alpha > 1/2$. Such enormous α is obviously impractical: even traditional phase estimation does not round to the nearest bit in this regime.

Second, the performance of Proposition 5.5 exhibits a ziz-zag behavior. This is because we reduce α by estimating more bits than we need and then rounding them away, a process that can only obtain α of the form 2^{-1-r} for integer r. When $\alpha > \frac{1}{2}$ then we no longer use the rounding strategy since we can achieve these values with median amplification alone. Therefore, for large enough α the line is no longer a zig-zag and begins to be a curve with shape $\sim \alpha^{-2}$.

The zig-zag behavior makes comparison for continuous values of α complicated. In our analysis we would like to compare against the most efficient version of traditional phase estimation, so we continue our performance analysis where α is a power of two, maximizing the efficiency (but minimizing our speedup). In Figure 5.6 we show the performance when vary α, n and δ independently.

We see that once $n \gtrsim 10$, $\alpha \lesssim 2^{-10}$ and $\delta \lesssim 10^{-30}$ the speedup stabilizes at about 14x for phase estimation and 20x for energy estimation. Our method therefore shows an significant improvement over the state of the art.

Of course, several assumptions needed to be made in order to claim a particular multiplicative speedup. Many of these assumptions were made specifically to maximize the performance of the traditional method. For example, we count performance in terms of query complexity rather than gate complexity, which neglects all the additional processing that phase estimation needs to perform for median amplification. This method of comparison favors the traditional method, since it neglects the fact that our new methods require significantly less ancillary processing. Furthermore, we select α to be a power of two, so that phase estimation is maximally efficient. However, we also assume that n is large enough and that α and δ are small enough that the speedup is stable. If the accuracy required is not so large, then the speedup is less significant.

Which method is best in reality will depend on the situation. In particular, the appropriate choice of α when studying real-world Hamiltonians remains an interesting direction of study. In reality it will not be possible to guarantee a rounding promise, so one's only option is to pick a small value of α and hope for the best. How small of an α is required?

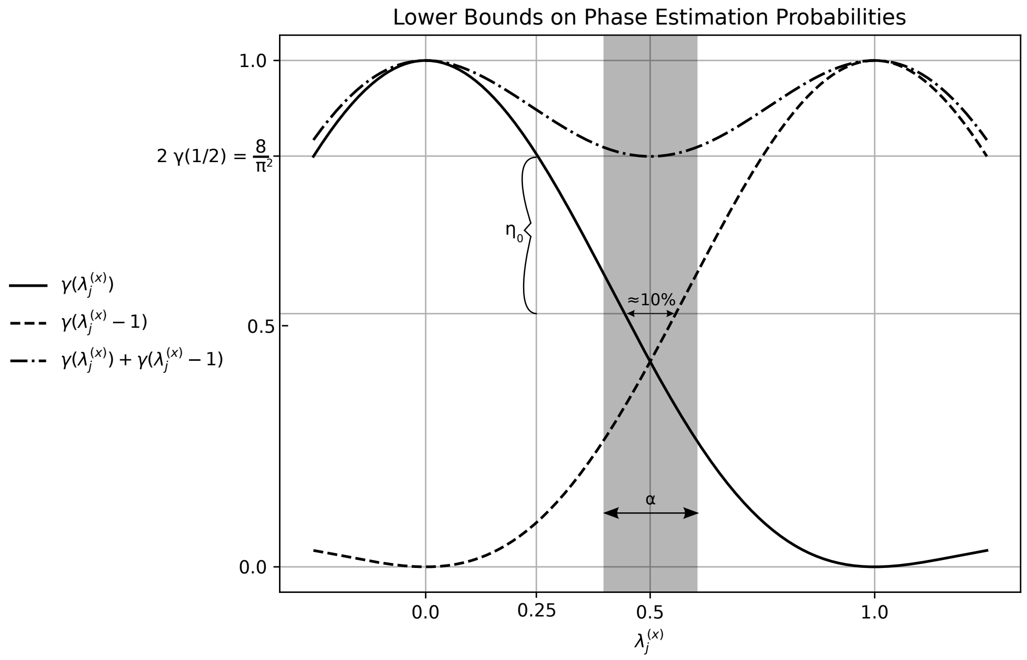


Figure 5.1: Sketch of $\gamma(\lambda_j^{(x)})$, a lower bound on the magnitude of the amplitude of phase estimation $|\beta(\lambda_j^{(x)})|^2$. Recall that the rounding promise guarantees that $\lambda_j \notin \left[\frac{x}{2^n}, \frac{x}{2^n} + \frac{\alpha}{2^n}\right]$ for all x, implying that $\lambda_j^{(x)} \notin [\frac{1-\alpha}{2}, \frac{1+\alpha}{2}]$ as indicated in the shaded region. From this sketch we draw two conclusions necessary for our proof. First, when $\alpha > 1/2$ then $\gamma(\lambda_j^{(x)})$ is guaranteed to be strictly greater than 1/2, so $\eta > 0$ (in fact, $\eta > \eta_0$). Second, even when no rounding promise applies, the sum of the probabilities of two adjacent bins is at least $8/\pi^2$, which we use to define the amplification threshold η_0. Also visible in this figure is the impossibility of reducing α further than $\approx 10\%$ without rounding: several values of $\lambda_j^{(x)}$ near 1/2 have probabilities less than 1/2 and cannot be amplified.

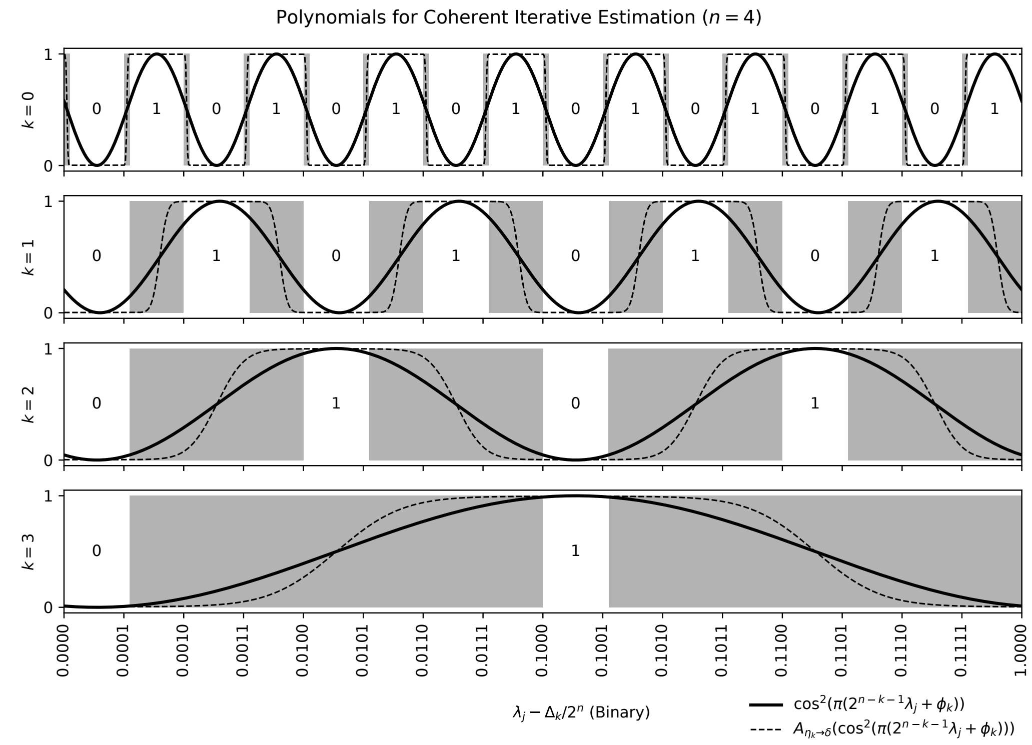


Figure 5.2: Sketch of the polynomials used in Theorems 5.9 and 5.12. In black we show a shifted $\cos^2(x)$ function that indicates if the bit is 0 or 1. Then, the amplifying polynomial from Corollary 3.23 is applied to it to yield the dashed line, which is either $\leq \delta$ or $\geq 1 - \delta$ depending on the bit. For the $k = 0$ bit, the gaps between the allowed intervals are guaranteed by the rounding promise. But as more bits are estimated and subtracted off, the gaps for $k \geq 1$ require no rounding promise, and also become larger and larger so less and less amplification is needed.

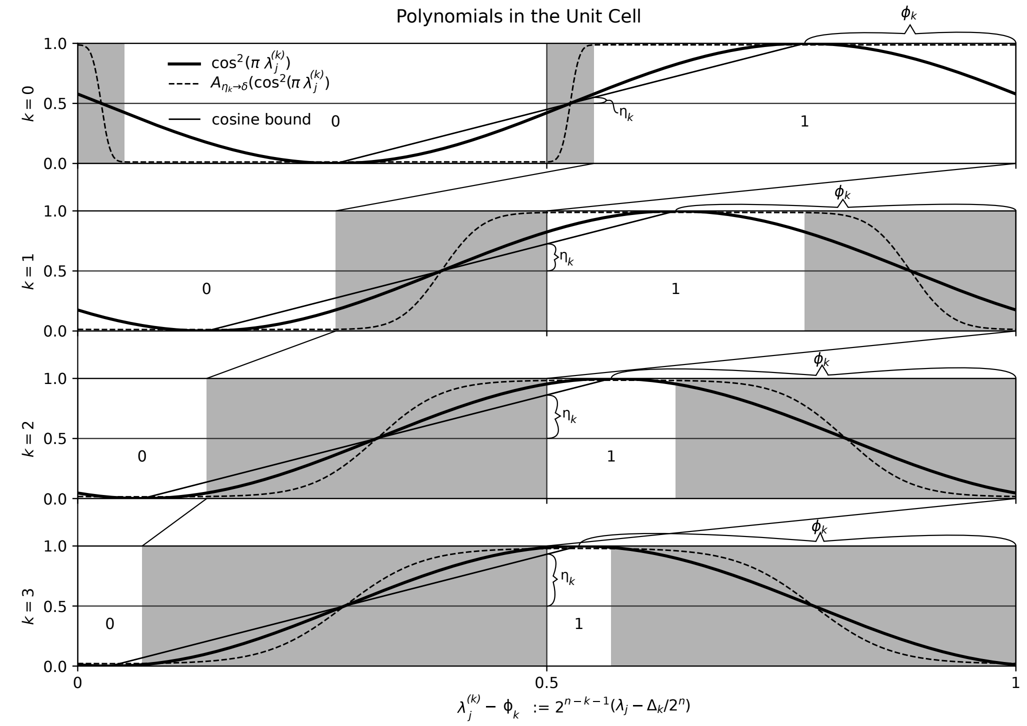
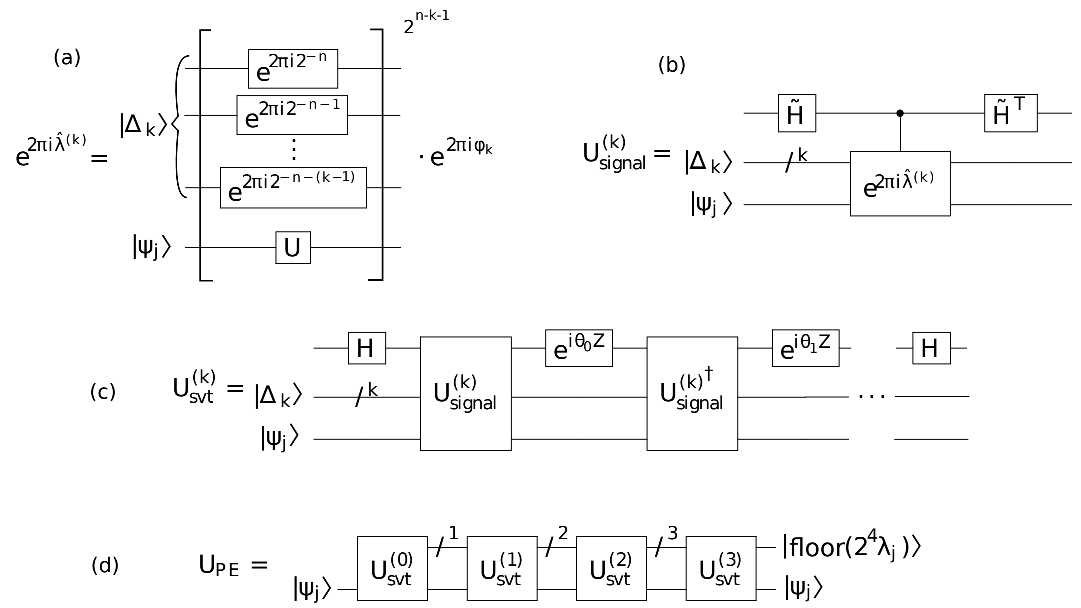


Figure 5.3: Sketch of the probability $\cos^2\left(\pi\lambda_j^{(k)}\right)$ which appears in the proof of Theorem 5.9.
We are guaranteed that $\lambda_j^{(k)}$ can only appear in the un-shaded regions: for $k = 0$ the rounding promise rules out the shaded regions, and for $k \geq 1$ we have subtracted $\Delta_k/2^n$ off of λ_j, preventing regions where previous bits are 1 .
We can see how the probability $\cos^2\left(\pi\lambda_j^{(k)}\right)$ is close to 1 if $\text{bit}_k(\lambda_j) = 0$ and close to 0 if $\text{bit}_k(\lambda_j) = 1$. To make this claim precise, we simply fit a line with slope 2 to the points where the probability intersects 1/2, and see that this line alternatingly gives upper or lower bounds on the probability. So if $\eta_k/2$ is equal to half the distance between allowed intervals, then the probability is either $\geq 1/2 + \eta_k$ or $\leq 1/2 - \eta_k$ in the regions where $\lambda_j^{(k)}$ can appear.
The relationship of this figure to Figure 5.2 is fairly simple: the probability as a function of λ_j can be obtained by just tiling the 'unit cell' shown in this figure 2^{n-k-1} times. This also makes the ratio 2^{n-k-1} between λ_j and $\lambda_j^{(k)}$ intuitive.


(a)
$e^{2\pi i\hat{\lambda}^{(k)}} =$
$|\Delta_k\rangle$
$e^{2\pi i 2^{-n}}$
$e^{2\pi i 2^{-n-1}}$
$e^{2\pi i 2^{-n-(k-1)}}$
$|\psi_j\rangle$
U
2^{n-k-1}
$\cdot\, e^{2\pi i \varphi_k}$
(b)
$U^{(k)}_{\mathrm{signal}} =$
$\tilde{H}$
$\tilde{H}^T$
k
$e^{2\pi i\hat{\lambda}^{(k)}}$
(c)
$U^{(k)}_{\mathrm{svt}} =$
H
$U^{(k)}_{\mathrm{signal}}$
$e^{i\theta_0 Z}$
$U^{(k)\dagger}_{\mathrm{signal}}$
$e^{i\theta_1 Z}$
(d)
$U_{PE} =$
$U^{(0)}_{\mathrm{svt}}$
$U^{(1)}_{\mathrm{svt}}$
$U^{(2)}_{\mathrm{svt}}$
$U^{(3)}_{\mathrm{svt}}$
$|\mathrm{floor}(2^4\lambda_j)\rangle$
Figure 5.4: Circuit diagram for the algorithm in Corollary 5.10. (a) and (b) are depictions of (5.98) and (5.103) respectively. (c) depicts the singular value transformation circuit obtained from Theorem 3.17 which interperses alternating applications of $U^{(k)}_{\text{signal}}$ and $U^{(k)\dagger}_{\text{signal}}$ with phase rotations $e^{i\theta_j Z}$, where the angles θ_j encode the polynomial approximating $A_{\eta\to\delta_{\text{amp}}}(x^2)$. (d) depicts the final circuit assembled via Lemma 5.7, showing how each iterative estimator's output becomes part of the $|\Delta_k\rangle$ register for the next, and how that register finally becomes the output.

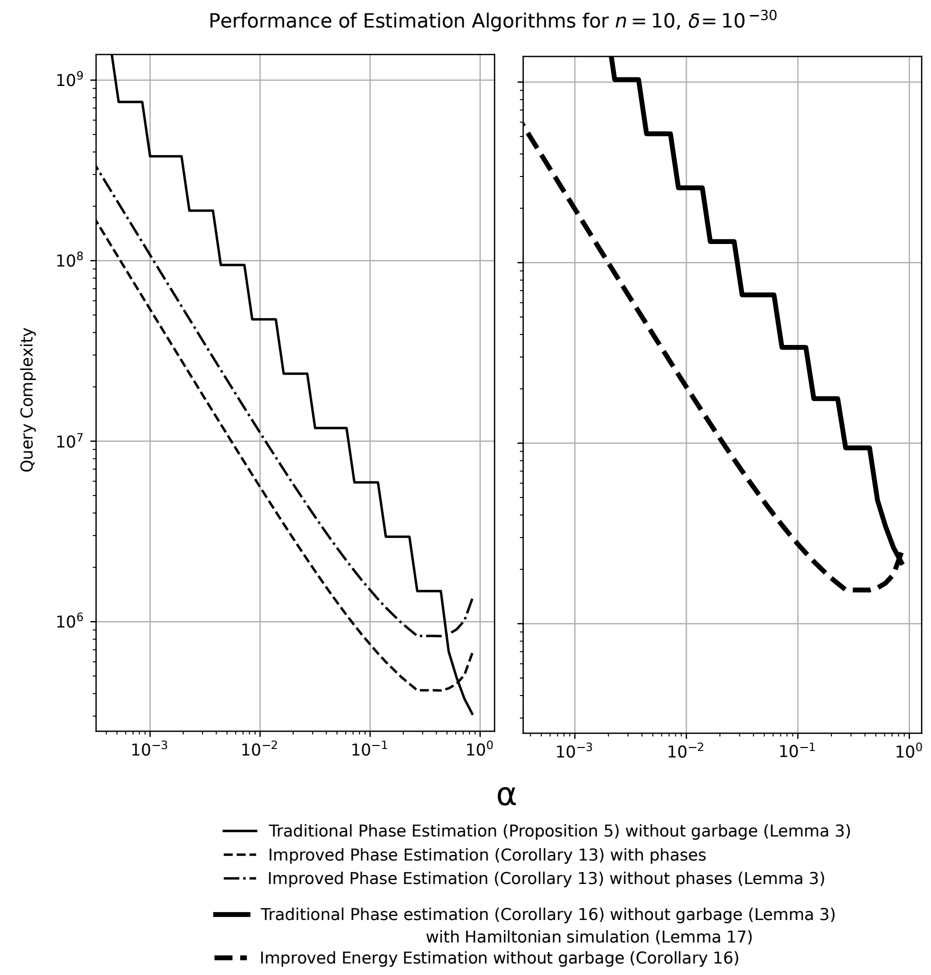


Figure 5.5: Performance of estimation algorithms presented in this chapter. Phase estimation algorithms are presented with slim lines on the left, and energy estimation algorithms are presented with thick lines on the right. The zig-zag behavior of phase estimation is explained by the method by which Proposition 5.5 reduces α: by estimating more bits than needed and then rounding them. Thus, the α achieved by phase estimation is always a power of two.

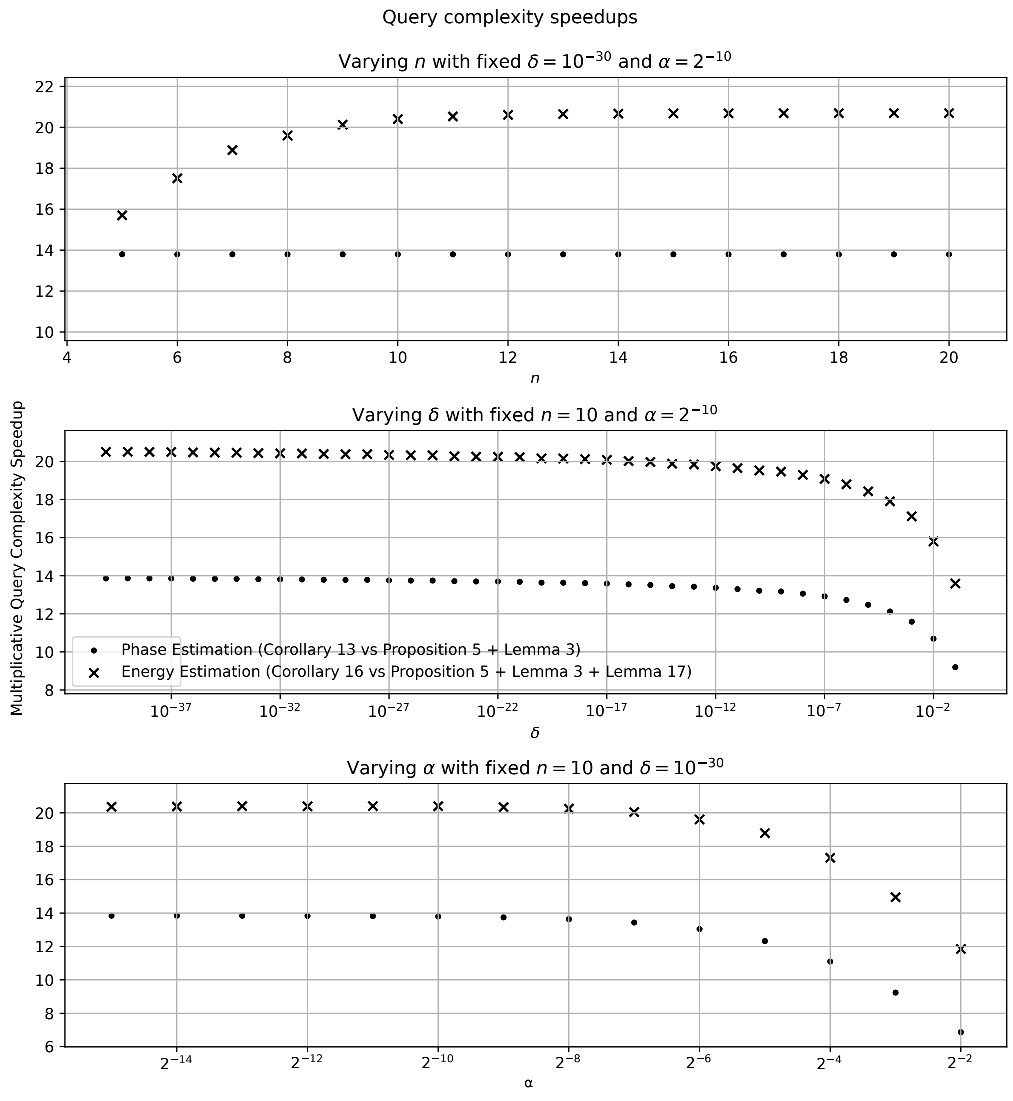


Figure 5.6: Speedup over traditional methods by our new methods for phase and energy estimation. When $n \gtrsim 10$, $\delta \lesssim 10^{-30}$ and $\alpha \lesssim 2^{-10}$ the speedups become stable. Together with Figure 5.5 we can conclude that the speedup for phase estimation is about 14x and the speedup for energy estimation is about 20x.

5.4 Block-Measurement

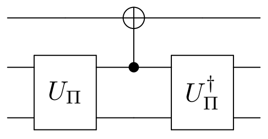
We have presented improved algorithms for phase and energy estimation. In this section we prove the block-measurement theorem from the introduction. The goal of the block-measurement protocol is the following: given an approximate block encoding of a projector Π, implement a channel close to the unitary:

$$|1\rangle \otimes \Pi + |0\rangle \otimes (I - \Pi) \tag{5.183}$$

When the block encoding of Π is exact, then the above is easily accomplished through linear combinations of unitaries and oblivious amplitude amplification. In particular, we can rewrite the above as $|0\rangle \otimes I - \sqrt{2}\,|-\rangle \otimes \Pi$, so we need to amplify away a factor of $1/(1+\sqrt{2})$ from the linear combination. Following Theorem 28 of [GSLW18], we observe that $T_5(x)$ is the first Chebyshev polynomial that has a solution $T_5(x) = \pm 1$ such that $x < 1/(1+\sqrt{2})$. We nudge the factor down to the solution x with some extra postselection, and then we meet the conditions of this theorem. This demands five queries to the block encoding of Π. Then we invoke our Proposition 5.15 to turn the block encoding of $|1\rangle \otimes \Pi + |0\rangle \otimes (I - \Pi)$ into a channel.

However, the above strategy is both more complicated and more expensive than necessary - we can do this in just two queries. Say U_Π is a block encoding of Π with m ancillae. Then, let V_Π:

$V_\Pi :=$



(5.184)

where the CNOT above refers to $X \otimes |0^m\rangle\langle 0^m| + I \otimes (I - |0^m\rangle\langle 0^m|)$. Then, V_Π satisfies:

$$(I \otimes \langle 0^m| \otimes I) V_\Pi (|0\rangle \otimes |0^m\rangle \otimes I) \tag{5.185}$$

$$= |1\rangle \otimes \Pi + |0\rangle \otimes (I - \Pi) \tag{5.186}$$

So viewing the $|0^m\rangle$ register as the postselective part of a block encoding, we see that V_Π is a block encoding of the desired operation. Then we invoke Proposition 5.15 to construct a channel Λ_Π from V_Π. Now all that is left to do is the error analysis.

In fact, when $m = 1$, then there is an extent that we do not even need the modified CNOT gate and can get away with just a single query. We used this fact in Theorem 5.9. This is because if $\Pi = \sum_j \alpha_j |\psi_j\rangle\langle\psi_j|$, then we can write:

$$U_\Pi(|0\rangle \otimes I) = \sum_j (\alpha_j |0\rangle + \beta_j |1\rangle) |\psi_j\rangle\langle\psi_j| \tag{5.187}$$

where β_j is some amplitude satisfying $|\beta_j|^2 + |\alpha_j|^2 = 1$. To compare, the desired operation is:

$$\sum_j (\alpha_j |1\rangle + (1 - \alpha_j) |0\rangle) |\psi_j\rangle\langle\psi_j| \tag{5.188}$$

Since $\alpha_j \in \{0, 1\}$, we know that $\beta_j = e^{i\phi_j}(1 - \alpha_j)$ for some phase ϕ_j that may depend on $|\psi_j\rangle$. So, with the exception of the phase correction, we are already one X gate away from the desired unitary.

Is it possible to remove this phase correction without uncomputation? If U_Π is obtained through singular value transformation of some real eigenvalues λ_j then we actually have more control than Theorem 3.17 would indicate. Looking at Theorem 3 of [GSLW18], we can actually select polynomials P, Q such that $\alpha_j = P(\lambda_j)$ and $\beta_j = Q(\lambda_j)$, provided that

P, Q satisfy some conditions including $|P(x)|^2 + (1-x^2)|Q(x)|^2 = 1$. For what positive-real-valued polynomials $P(x)$ is it possible to choose a positive-real-valued $Q(x)$ that satisfies this relation, thus removing the phases $e^{i\phi_j}$? We leave this question for future work.

Now we proceed to the formal statement and error analysis of the block-measurement theorem.

Theorem 5.16. ***Block-measurement.*** *Say Π is a projector, and A is a hermitian matrix satisfying $|\Pi - A^2| \leq \varepsilon$. Also, A has a block encoding U_A. Then the channel Λ_A, constructed in the same way as Λ_Π above just with U_A instead of U_Π, approximates Λ_Π in diamond norm:*

$$|\Lambda_A - \Lambda_\Pi|_\diamond \leq 4\sqrt{2}\varepsilon \tag{5.189}$$

Proof. Just like V_Π, V_A is a block encoding of the unitary:

$$X \otimes A^2 + I \otimes (I - A^2) \tag{5.190}$$

The distance in spectral norm to $V_\Pi = X \otimes \Pi + I \otimes (I - \Pi)$ is:

$$\begin{aligned}
&\Big| |1\rangle \otimes \Pi + |0\rangle \otimes (I - \Pi) - |1\rangle \otimes A^2 - |0\rangle \otimes (I - A^2) \Big| && (5.191)\\
=&\Big| |1\rangle \otimes (\Pi - A^2) + |0\rangle \otimes (A^2 - \Pi) \Big| && (5.192)\\
=&\sqrt{2}\left| \frac{|0\rangle - |1\rangle}{\sqrt{2}} \otimes (\Pi - A^2) \right| && (5.193)\\
=&\sqrt{2}\left|\Pi - A^2\right| \leq \sqrt{2}\varepsilon && (5.194)
\end{aligned}$$

So we have a block encoding of a matrix $\sqrt{2}\varepsilon$-close in spectral norm to the unitary $X \otimes \Pi + I \otimes (I - \Pi)$. Now we just use Proposition 5.15, which gives a channel with diamond norm accuracy $4\sqrt{2}\varepsilon$. Then we get the desired operation by initializing an extra qubit to $|0\rangle$ before applying the channel. □

What happens when A is not a projector? For simplicity, say the input state to the circuit is $|\Psi\rangle$, which happens to be an eigenstate of A. Then:

$$V_A\left(|0\rangle \otimes |0^m\rangle \otimes |\Psi\rangle\right) = \left(|1\rangle \otimes U_A^\dagger |0^m\rangle \langle 0^m| U_A |0^m\rangle + |0\rangle \otimes U_A^\dagger \left(I - |0^m\rangle \langle 0^m|\right) U_A |0^m\rangle\right) \otimes |\Psi\rangle \tag{5.195}$$

$$= \left(-\sqrt{2}\,|-\rangle \otimes U_A^\dagger |0^m\rangle \langle 0^m| U_A |0^m\rangle + |0\rangle \otimes U_A^\dagger U_A |0^m\rangle\right) \otimes |\Psi\rangle \tag{5.196}$$

We are interested in the state obtained by tracing out the $|0^m\rangle$ register above. For $j \in \{0,1\}^m$, we let γ_j abbreviate a matrix element of U_A:

$$\gamma_j |\Psi\rangle := (\langle 0^m| \otimes I) U_A (|j\rangle \otimes |\Psi\rangle) \tag{5.197}$$

$$(I \otimes \langle j| \otimes I)\, V_A\left(|0\rangle \otimes |0^m\rangle \otimes |\Psi\rangle\right) = \left(-\gamma_j^\dagger \gamma_0 \sqrt{2}\,|-\rangle + \delta_{j,0}\,|0\rangle\right) \otimes |\Psi\rangle \tag{5.198}$$

Note that γ_0 is the eigenvalue of A corresponding to $|\Psi\rangle$. Then, the reduced density matrix after tracing out the middle register is:

$$\sum_{j \in \{0,1\}^m} \left(-\gamma_j^\dagger \gamma_0 \sqrt{2}\,|-\rangle + \delta_{j,0}\,|0\rangle\right)\left(-\gamma_j \gamma_0^\dagger \sqrt{2}\,\langle -| + \delta_{j,0}\,\langle 0|\right) \otimes |\Psi\rangle \langle \Psi| \tag{5.199}$$

$$= \sum_{j \in \{0,1\}^m} \left(2|\gamma_j|^2 |\gamma_0|^2 \,|-\rangle \langle -| - \sqrt{2}\delta_{j,0}\gamma_j\gamma_0^\dagger\,|0\rangle \langle -| - \sqrt{2}\delta_{j,0}\gamma_j^\dagger\gamma_0\,|-\rangle \langle 0| + \delta_{j,0}\,|0\rangle \langle 0|\right) \otimes |\Psi\rangle \langle \Psi| \tag{5.200}$$

$$= \left(2|\gamma_0|^2\,|-\rangle \langle -| - \sqrt{2}|\gamma_0|^2\,|0\rangle \langle -| - \sqrt{2}|\gamma_0|^2\,|-\rangle \langle 0| + |0\rangle \langle 0|\right) \otimes |\Psi\rangle \langle \Psi| \tag{5.201}$$

$$= \left(|\gamma_0|^2\,|1\rangle \langle 1| + (1 - |\gamma_0|^2)\,|0\rangle \langle 0|\right) \otimes |\Psi\rangle \langle \Psi| \tag{5.202}$$

This somewhat complicated calculation produced a simple result: tracing out the $|0^m\rangle$ register collapses the superposition on the output register. A more general version of this calculation where the input state is not an eigenstate of A demonstrates that this collapse also damages the superposition on the input register (although it does not fully collapse it). Intuitively, we can see why this is the case even without doing the full calculation: since measuring the middle register collapses the superposition that encodes the eigenvalues, the environment that traced out the middle register thus learns information about the eigenvalues. But the distribution over eigenvalues also contains information about the input superposition over eigenvectors, so therefore the input state will be damaged.

In essence, we have re-derived the fact that uncomputation is impossible unless the output of the computation is deterministic.

5.5 Non-Destructive Amplitude Estimation

In this section we show how to use our energy estimation algorithm to perform amplitude estimation. The algorithms for energy and phase estimation demanded a rounding promise on the input Hamiltonian, which guarantees that they do not damage the input state even if it is not an eigenstate of the unitary or Hamiltonian of interest. However, as we shall see, this is not a concern for amplitude estimation. This is because we can construct a Hamiltonian whose eigenstate is the input state.

Say Π is some projector and $|\Psi\rangle$ is a quantum state. Non-destructive amplitude estimation obtains an estimate of $a = |\Pi|\Psi\rangle|$ given exactly one copy of $|\Psi\rangle$, and leaves that copy of $|\Psi\rangle$ intact. This subroutine is explicitly required by the algorithms in [HW19, AHNTW20], which can be used to perform Bayesian inference, thermal state preparation and partition function estimation. However, it is also quite useful in general when $|\Psi\rangle$ is expensive to prepare. For example, when estimating the expectation of an observable on the ground state of a Hamiltonian, preparing the ground state can be very expensive. Thus, it may be practical to only prepare it once.

The only previously known algorithm for non-destructive amplitude estimation is given in [HW19]. It works via several invocations of amplitude estimation according to [BHMT00], which is based on phase estimation. We argue that our new algorithm has several performance advantages over the prior art. However, these advantages are not very quantifiable, preventing us from computing a numerical constant-factor speedup. The advantages are as follows:

- Our new algorithm requires dramatically fewer ancillae. This is because [HW19] relies on phase estimation with median amplification. As argued above, median amplification requires $O(n \log(\delta^{-1}))$ ancillae. In contrast, we simply use the protocol for energy estimation which requires $n + O(1)$ ancillae.
- Our new algorithm runs in a fixed amount of time: one application of our energy estimation algorithm suffices. In contrast, [HW19]'s algorithm works by repeatedly attempting to 'repair' the input state. This process succeeds only with probability 1/2, so while the expected number of attempts is constant, the resulting algorithm is highly adaptive and may need a variable amount of time.
- Our new algorithm does not require knowledge of a lower bound on the amplitude a. The 'repair' step in [HW19] itself involves another application of phase estimation, which must produce an estimate with enough accuracy to distinguish $\arcsin(a)$ and $-\arcsin(a)$. This also implies that [HW19] can only produce relative-error estimates, even when an additive-error estimate might be sufficient, as it is in [AHNTW20].
- While, due to the above differences, it is not really possible to perform a side-by-side constant-factor comparison of the algorithms, we do expect to inherit a modest constant-factor speedup from our energy estimation algorithm. [BHMT00] has no need

to perform Hamiltonian simulation, so we expect the speedup to look more like the one in the case of phase estimation. Since no rounding promise is required, making α tiny is not really necessary. But it *is* necessary that $\alpha < 1/2$ since this makes traditional phase estimation round correctly. Looking at Figure 5.5, we already see constant factor speedups in this regime.

Another minor advantage of our method over [HW19] is that it estimates a^2 directly, rather than going through the Grover angle $\theta := \arcsin(a)$. Coherently evaluating a sine function in superposition to correct this issue, while possible, will have an enormous ancilla overhead. a^2 is the probability with which a measurement of $|\Psi\rangle$ yields a state in Π, which may be useful directly.

We note that an additive error estimate of a is slightly stronger than an additive error estimate of a^2. Moreover this accuracy seems to be necessary for some versions of quantum mean estimation: see for example Appendix C of [AHNTW20]. If the amplitude a is desired rather than a^2, then, since the algorithm proceeds by obtaining a block encoding of a^2, one can use singular value transformation to make a block encoding of a instead. A polynomial approximation of $\sqrt{x}$ could be constructed from Corollary 66 of [GSLW18]. We leave a careful error analysis of a direct estimate of a to future work.

Corollary 5.17. ***Non-destructive amplitude estimation.*** *Say* Π *is a projector and* R_Π *is a unitary that reflects about this projector:*

$$R_\Pi := 2\Pi - I \tag{5.203}$$

Let $|\Psi\rangle$ *be some quantum state such that* $a := |\Pi|\Psi\rangle|$. *Let* $M :=$ *floor*$(a^2 2^n)$. *Then for any positive integer* n *and any* $\delta > 0$ *there exists a quantum channel that implements* δ-

approximately in diamond norm the map:

$$|0^n\rangle\langle 0^n| \otimes |\Psi\rangle\langle\Psi| \to (p\,|M\rangle\langle M| + (1-p)\,|M-1 \textit{ mod } 2^n\rangle\langle M-1 \textit{ mod } 2^n|) \otimes |\Psi\rangle\langle\Psi| \tag{5.204}$$

for some probability p (i.e., there is some probability that instead of obtaining floor$(a^2 2^n)$ we obtain floor$(a^2 2^n) - 1$). This channel uses $O\left(2^n \log(\delta^{-1})\right)$ controlled applications of R_Π and $R_{|\Psi\rangle} := 2\,|\Psi\rangle\langle\Psi| - I$.

Proof. We use linear combinations of unitaries to make block encodings of Π and $|\Psi\rangle\langle\Psi|$.

$$\Pi = \frac{I + R_\Pi}{2}, \qquad |\Psi\rangle\langle\Psi| = \frac{I + R_{|\Psi\rangle}}{2} \tag{5.205}$$

Then, we multiply these projectors together to make a block encoding of A:

$$A := |\Psi\rangle\langle\Psi| \cdot \Pi \cdot |\Psi\rangle\langle\Psi| = a^2\,|\Psi\rangle\langle\Psi| \tag{5.206}$$

Now we simply invoke the algorithm described in Corollary 5.13 for the case when no rounding promise is present, and apply it to the input state $|\Psi\rangle$. Since $|\Psi\rangle$ is an eigenstate of A, we are guaranteed to measure either floor$(a^2 2^{n-1})$ or floor$(a^2 2^{n-1}) - 1$ mod 2^n. $\square$

Observe that the random error only occurs for particular values of a^2, which is where the rounding promise is violated. We can ignore the rounding promise precisely because the input state $|\Psi\rangle$ is an eigenstate of the Hamiltonian: an incorrect estimate does not damage the input state.

Chapter 6

Estimating Near-Clifford Quantum Circuits

The work presented in this chapter is based on [RLCK19], which was the result of a collaboration between students in the Aaronson group. Patrick Rall designed and analyzed the algorithms, wrote the text, and contributed theorems and figures throughout. Daniel Liang and William Kretschmer proved many of the theorems in Section 6.2 and prepared some of the figures. Jeremy Cook performed the numerical simulations in Section 6.3.

Simulating quantum circuits on classical hardware requires large computational resources. Near-Clifford simulation techniques extend the Gottesmann-Knill theorem to arbitrary quantum circuits while maintaining polynomial time simulation of stabilizer circuits. Their runtime analysis gives rise to measures of non-Cliffordness, such as the robustness of magic [HC16], magic capacity [SC19], sum-negativity [VMGE13]. These algorithms evaluate circuits by estimating the mean of some probability distribution via the average of many samples, a process with favorable memory requirements and high parallelizability.

Previous work [Bennink&17, HC16] gives an algorithm based on quasiprobability distributions over stabilizer states; we refer to this algorithm as 'stabilizer propagation'. In contrast to techniques based on stabilizer rank [BG16, Bravyi&18], stabilizer propagation is appealing for simulation of NISQ-era hardware [Preskill18] because it can simu:late noisy channels. Moreover, depolarizing noise decreases the number of samples required, measured by robustness of magic and the magic capacity. However, bounding the number of required samples can be expensive: For example, the magic capacity of a 3-qubit channel is defined as a convex optimization problem over 315,057,600 variables [HG18, SC19].

Pashayan et al. [PWB15] showed that in qutrit systems, the discrete Wigner function provides a simpler simulation strategy. This strategy takes linear time to sample, and the number of samples required (measured by the sum-negativity) is tractable to compute for small systems. However, discrete Wigner functions do not yield efficient simulation of qubit Clifford circuits [Raussendorf&15].

Our main result is that Bloch vectors yield simulation strategies for qubit circuits, similar to those in Pashayan et al. We present two algorithms, which we individually call **Schrödinger propagation** and **Heisenberg propagation**, and collectively call **Pauli propagation techniques**. They have several surprising properties:

1. They yield linear time simulation for qubit Clifford circuits without writing down stabilizer states.

2. Schrödinger propagation can efficiently simulate a new family of quantum states called 'hyper-octahedral states' which is significantly larger than the set of stabilizer mixtures in terms of the Hilbert-Schmidt measure.

3. The runtime of Heisenberg propagation does not depend on the input state at all.

4. Non-Cliffordness in both algorithms is measured via the stabilizer norm, which is a lower bound to the robustness of magic. This gives Pauli propagation techniques a strictly lower runtime than stabilizer propagation for all input states and most channels.

We describe these algorithms in Section 6.1. In Section 6.2 we perform a detailed comparison of Schrödinger, Heisenberg and stabilizer propagation which we summarize in the table below. In Section 6.3 we present some numerical results. In Section 6.4 we briefly discuss the implications of the algorithms for resource theories of Cliffordness. In the final two sections we discuss some technical points we allude to throughout the text.

6.1 Algorithms

In this section we describe two algorithms for estimating the expectation value of observables at the end of a quantum circuit. Schrödinger propagation involves propagating states forward though the circuit and taking inner products with the final observables. Heisenberg propagation involves propagating observables backward though the circuit and taking inner products with the initial states. At every step, both procedures sample from an unbiased estimator for the propagated state/observable that is distribution over Pauli matrices.

6.1.1 Sampling Pauli Matrices

The workhorse of both protocols is a subroutine that samples a random scaled tensor product of Pauli matrices as a proxy for an arbitrary n-qubit Hermitian matrix A. Let $\mathcal{P}_n = \{\sigma_1 \otimes \cdots \otimes \sigma_n : \sigma_i \in \{I, \sigma_X, \sigma_Y, \sigma_Z\}\}$ denote the set of n-qubit Pauli matrices. We define a pair of completely dependent random variables $\hat{\sigma} \in \mathcal{P}_n$ and $\hat{c} \in \mathbb{R}$ that satisfy $\mathbb{E}\left[\hat{c} \cdot \hat{\sigma}\right] = A$:

$$\hat{\sigma}(A) = \sigma \text{ with prob. } \frac{|\mathrm{Tr}(\sigma A)|}{2^n \cdot \mathcal{D}(A)} \text{ for each } \sigma \in \mathcal{P}_n, \tag{6.1}$$

$$\hat{c}(A) = \mathrm{sign}\left(\mathrm{Tr}(\hat{\sigma}(A)A)\right) \cdot \mathcal{D}(A). \tag{6.2}$$

The quantity $\mathcal{D}(A)$ is a normalization constant that makes $\frac{|\mathrm{Tr}(\sigma A)|}{2^n \cdot \mathcal{D}(A)}$ for $\sigma \in \mathcal{P}_n$ a probability distribution.

Definition 6.1. *The* ***stabilizer norm*** $\mathcal{D}(A)$ *is:*

$$\mathcal{D}(A) = \frac{1}{2^n} \sum_{\sigma \in \mathcal{P}_n} |\mathrm{Tr}(\sigma A)| \,. \tag{6.3}$$

The product of the random variables $\hat{c}(A) \cdot \hat{\sigma}(A)$ is an unbiased estimator for A because the Pauli matrices form an operator basis for Hermitian matrices:

$$\begin{aligned} \mathbb{E}[\hat{c}(A) \cdot \hat{\sigma}(A)] &= \sum_{\sigma \in \mathcal{P}_n} \frac{|\mathrm{Tr}(\sigma A)|}{2^n \cdot \mathcal{D}(A)} \cdot \mathrm{sign}\left(\mathrm{Tr}(\sigma A)\right) \cdot \mathcal{D}(A) \cdot \sigma \\ &= \sum_{\sigma \in \mathcal{P}_n} \frac{\mathrm{Tr}(\sigma A)}{2^n} \cdot \sigma = A. \end{aligned} \tag{6.4}$$

The time to compute the probabilities and sample from the distributions scales exponentially with the number of qubits of A. We say A has **tensor product structure** if it can be written as a tensor product of several operators, each of which acts on a constant number of qubits:

$$A = A_1 \otimes A_2 \otimes \cdots$$

Then one can observe that:

$$\hat{\sigma}(A) = \hat{\sigma}(A_1) \otimes \hat{\sigma}(A_2) \cdots \text{ and } \hat{c}(A) = \hat{c}(A_1) \cdot \hat{c}(A_2) \cdots$$

Since each A_i acts on a constant number of qubits, each of the probability distributions for $\hat{\sigma}(A_i), \hat{c}(A_i)$ can be computed and sampled from in constant time. So $\hat{\sigma}(A)$ and $\hat{c}(A)$ can be sampled from in linear time if A has tensor product structure, even if A acts on many qubits.

6.1.2 Schrödinger Propagation

Suppose we want to apply a sequence of channels $\Lambda_1, \dots, \Lambda_k$ to an n-qubit state ρ_0. These operations are given as a quantum circuit, so ρ_0 has tensor product structure and each of the Λ_i non-trivially act on a constant-size subset of the qubits. Let ρ_i be the state after applying the first i channels:

$$\rho_i = \Lambda_i(\Lambda_{i-1}(\cdots \Lambda_1(\rho_0))) \tag{6.5}$$

We are given an observable E which also has tensor product structure. We want to estimate the expectation of E on the final state:

$$\langle E \rangle = \operatorname{Tr}(E\rho_k) = \operatorname{Tr}(E\Lambda_k(\Lambda_{k-1}(\cdots \Lambda_1(\rho_0)))) \tag{6.6}$$

We apply the sampling procedure defined by (6.1) and (6.2) to ρ_0. We define $\hat{\sigma}(\rho_0) = \hat{\sigma}_0$ and $\hat{c}(\rho_0) = \hat{c}_0$. Their product $\hat{c}_0 \cdot \hat{\sigma}_0$ is an unbiased estimator for ρ_0.

Given an unbiased estimator $\hat{c}_i \cdot \hat{\sigma}_i$ for ρ_i, we will obtain an unbiased estimator $\hat{c}_{i+1} \cdot \hat{\sigma}_{i+1}$ for ρ_{i+1}. Apply Λ_{i+1} to $\hat{c}_i \cdot \hat{\sigma}_i$ and use linearity of Λ_{i+1}:

$$\mathbb{E}[\Lambda_{i+1}(\hat{c}_i \cdot \hat{\sigma}_i)] = \Lambda_{i+1}(\mathbb{E}[\hat{c}_i \cdot \hat{\sigma}_i]) = \rho_{i+1}$$

We have $\Lambda_{i+1}(\hat{c}_i \cdot \hat{\sigma}_i) = \hat{c}_i \cdot \Lambda_{i+1}(\hat{\sigma}_i)$. Since Λ_{i+1} acts non-trivially on a constant-size subset of the qubits, $\Lambda_{i+1}(\hat{\sigma}_i)$ has tensor product structure and we can sample using (6.1) and (6.2) again. Let:

$$\hat{\sigma}_{i+1} = \hat{\sigma}\left(\Lambda_{i+1}(\hat{\sigma}_i)\right) \text{ and } \hat{c}_{i+1} = \hat{c}_i \cdot \hat{c}\left(\Lambda_{i+1}(\hat{\sigma}_i)\right) \tag{6.7}$$

Now we have $\hat{c}_{i+1} \cdot \hat{\sigma}_{i+1}$, an estimator for ρ_{i+1}, and can recursively obtain $\hat{c}_k \cdot \hat{\sigma}_k$ for ρ_k. Since E and $\hat{\sigma}_k$ have tensor product structure, we can efficiently obtain their trace inner product. The protocol yields a sample from the distribution in time linear in $k+n$:

$$\text{Output: sample from } \hat{c}_k \cdot \mathrm{Tr}(\hat{\sigma}_k E) \tag{6.8}$$

This distribution estimates the target quantity:

$$\mathbb{E}\left[\hat{c}_k \cdot \mathrm{Tr}(\hat{\sigma}_k E)\right] = \mathrm{Tr}(\mathbb{E}\left[\hat{c}_k \cdot \hat{\sigma}_k\right] E) = \mathrm{Tr}\left(\rho_k E\right) = \langle E \rangle$$

We estimate the mean of $\hat{c}_k \cdot \mathrm{Tr}(\hat{\sigma}_k E)$ by taking the average of N samples. The Hoeffding inequality [Hoeffding63] provides a sufficient condition on N for an additive error ε with probability $1-\delta$ in terms of the range of the distribution:

$$N \geq \frac{1}{2\varepsilon^2} \cdot \ln \frac{2}{\delta} \cdot (\text{range})^2 \tag{6.9}$$

The range of the output distribution is bounded by twice the maximum magnitude of the output distribution (6.8).

$$\text{range} \leq 2 \cdot |\hat{c}_k \cdot \mathrm{Tr}(\hat{\sigma}_k E)| \leq 2 \cdot |\hat{c}_k| \cdot \max_{\sigma \in \mathcal{P}_n} |\mathrm{Tr}(\sigma E)| \tag{6.10}$$

Observe that $\hat{c}(A) = \pm \mathcal{D}(A)$, so:

$$\begin{aligned} |\hat{c}_{i+1}| &= |\hat{c}_i| \cdot |\hat{c}\left(\Lambda_{i+1}(\hat{\sigma}_i)\right)| \\ &= |\hat{c}_i| \cdot \mathcal{D}(\Lambda_{i+1}(\hat{\sigma}_i)) \\ &\leq |\hat{c}_i| \cdot \max_{\sigma \in \mathcal{P}_n} \mathcal{D}(\Lambda_i(\sigma)) \end{aligned} \tag{6.11}$$

Intuitively, $\mathcal{D}$ measures the “cost” of a Hermitian matrix in this algorithm. The above motivates a corresponding notion of the “cost” of a channel:

Definition 6.2. *The* ***channel stabilizer norm*** $\mathcal{D}(\Lambda)$ *is defined by:*

$$\mathcal{D}(\Lambda) = \max_{\sigma \in \mathcal{P}_n} \mathcal{D}(\Lambda(\sigma)) \tag{6.12}$$

Expanding the recursion in (6.11) we obtain the bound:

$$|\hat{c}_k \cdot \mathrm{Tr}(\hat{\sigma}_k E)| \leq \underbrace{\mathcal{D}(\rho_0)}_{(1)} \cdot \underbrace{\prod_{i=1}^{k} \mathcal{D}(\Lambda_i)}_{(2)} \cdot \underbrace{\left|\max_{\sigma \in \mathcal{P}_n} \mathrm{Tr}(\sigma E)\right|}_{(3)} \tag{6.13}$$

The number of samples N scales with the square of the above quantity. Thus, the cost of Schrödinger propagation on a circuit breaks into three parts: (1) the cost of the initial state, (2) the cost of each channel, and (3) the cost of the final observable.

Here are two observations:

- Say $\rho_0 = \rho^{\otimes m}$, so $\mathcal{D}(\rho_0) = \mathcal{D}(\rho)^m$. For many ρ with short Bloch vectors, the cost $\mathcal{D}(\rho)$ can be strictly less than 1, meaning more copies of ρ result in an exponential runtime improvement from cost term (1).
- Often we are interested in observables E_{local} that act only on a small subset of the output qubits. Then E is a tensor product of linearly many identity matrices and E_{local}, resulting in an exponential runtime blowup from cost term (3).

Loosely speaking, Schrödinger propagation works well when the input qubits are noisy and all output qubits are measured, like some supremacy circuits [Martinis&16].

6.1.3 Heisenberg Propagation

Heisenberg propagation involves propagating the observable E backwards through the circuit and taking the inner product with the initial state ρ_0. To do so we utilize the **channel adjoint** $\Lambda^\dagger$ which satisfies:

$$\mathrm{Tr}(E\Lambda(\rho)) = \mathrm{Tr}(\Lambda^\dagger(E)\rho) \tag{6.14}$$

Applying this to (6.6), our goal is to estimate:

$$\langle E\rangle = \mathrm{Tr}\left(\rho_0\Lambda_1^\dagger(\cdots\Lambda_{k-1}^\dagger(\Lambda_k^\dagger(E)))\right) = \mathrm{Tr}\left(\rho_0 E_1\right)$$
$$\text{where } E_i = \Lambda_i^\dagger(\Lambda_{i+1}^\dagger(\cdots\Lambda_{k-1}^\dagger(\Lambda_k^\dagger(E)))) \tag{6.15}$$

For Heisenberg propagation we will define $\hat{c}_i, \hat{\sigma}_i$ differently from Schrödinger propagation. We use the sampling procedure defined by (6.1) and (6.2) and obtain $\hat{\sigma}(E) = \hat{\sigma}_{k+1}$ and $\hat{c}(E) = \hat{c}_{k+1}$. Then $\hat{c}_{k+1}\cdot\hat{\sigma}_{k+1}$ is an unbiased estimator for E.

With an unbiased estimator $\hat{c}_{i+1}\cdot\hat{\sigma}_{i+1}$ for E_{i+1} we can obtain an unbiased estimator $\hat{c}_i\cdot\hat{\sigma}_i$ for E_i from $\Lambda_i^\dagger(\hat{c}_{i+1}\cdot\hat{\sigma}_{i+1}) = \hat{c}_{i+1}\cdot\Lambda_i^\dagger(\hat{\sigma}_{i+1})$. Since $\Lambda_i^\dagger(\hat{\sigma}_{i+1})$ has tensor product structure we can sample using (6.1) and (6.2), and obtain:

$$\hat{\sigma}_i = \hat{\sigma}(\Lambda_i^\dagger(\hat{\sigma}_{i+1})) \text{ and } \hat{c}_i = \hat{c}_{i+1}\cdot\hat{c}(\Lambda_i^\dagger(\hat{\sigma}_{i+1})) \tag{6.16}$$

This operation is iterated until we obtain $\hat{c}_1\cdot\hat{\sigma}_1$, an unbiased estimator for E_1. Since ρ_0 has tensor product structure we can compute the trace inner product and produce a sample, again in time linear in $k+n$:

$$\text{Output: sample from } \hat{c}_1\cdot\mathrm{Tr}(\hat{\sigma}_1\rho_0) \tag{6.17}$$

This estimates the target quantity:

$$\mathbb{E}\left[\hat{c}_1\cdot\mathrm{Tr}(\hat{\sigma}_1\rho_0)\right] = \mathrm{Tr}(\mathbb{E}\left[\hat{c}_1\cdot\hat{\sigma}_1\right]\rho_0) = \mathrm{Tr}\left(E_1\rho_0\right) = \langle E\rangle$$

To bound the number of samples N we bound the maximum magnitude of (6.17) and utilize Hoeffding's inequality (6.9). Since ρ_0 is a quantum state, we always have $\max_{\sigma\in\mathcal{P}_n}|\mathrm{Tr}(\sigma\rho_0)| = 1$ since the eigenvalues of σ are ±1. This leaves the recursion relation:

$$\begin{aligned}|\hat{c}_1 \cdot \mathrm{Tr}(\hat{\sigma}_1\rho_0)| \le |\hat{c}_1| = |\hat{c}_{i+1}| \cdot \left|\hat{c}\left(\Lambda_i^\dagger(\hat{\sigma}_{i+1})\right)\right| \\ &= |\hat{c}_{i+1}| \cdot \mathcal{D}(\Lambda_i^\dagger(\hat{\sigma}_{i+1})) \\ &\le |\hat{c}_{i+1}| \cdot \max_{\sigma\in\mathcal{P}_n} \mathcal{D}(\Lambda_i^\dagger(\sigma)) \\ &= |\hat{c}_{i+1}| \cdot \mathcal{D}(\Lambda_i^\dagger) \end{aligned} \tag{6.18}$$

Expanding the recursion we obtain the bound:

$$|\hat{c}_1 \cdot \mathrm{Tr}(\hat{\sigma}_1\rho_0)| \le \underbrace{\mathcal{D}(E)}_{(1)} \cdot \underbrace{\prod_{i=1}^{k} \mathcal{D}(\Lambda_i^\dagger)}_{(2)} \tag{6.19}$$

The number of samples N scales with the square of the cost of the observable (1) and the cost of channel adjoints (2), and is independent of the initial state.

Loosely speaking, Heisenberg propagation is efficient for *any* separable input state or stabilizer mixture and supports a wider range of observables than Schrödinger propagation. However, it cannot capitalize on particularly noisy input states for a runtime improvement.

A version of Heisenberg propagation appears in [PBG17], where they restrict operations to Clifford unitaries. Our work generalizes the technique to arbitrary quantum channels.

6.2 Efficient Circuit Components

In this section we study which input states, channels and observables (collectively 'circuit components') can be simulated by Schrödinger, Heisenberg and stabilizer propagation without increasing runtime. This viewpoint helps address the practical question: "Given a particular quantum circuit, which near-Clifford algorithm is best?"

Straightaway, if the quantum circuit is unitary then stabilizer rank techniques [Bravyi&18] are the best choice due to their superior accuracy and runtime. The primary advantage of propagation algorithms is their ability to support arbitrary circuit components with noise, measurement, and adaptivity. Despite their flexibility, the propagation algorithms vary significantly in their performance.

Since the number of samples scales as the product of the square of the cost of the components, a component occurring linearly many times with cost > 1 demands exponential runtime. In the following, when we say an algorithm **supports** or **can handle** a component, we mean that the cost of the component is ≤ 1, although the protocols can be applied to any component possibly inefficiently.

6.2.1 Efficiency of Stabilizer Propagation

For a self-contained description of stabilizer propagation see [Bennink&17, HC16, SC19]. Just as the algorithms in section II decompose input states into a weighted sum of Pauli matrices, stabilizer propagation decomposes input states into a weighted sum of stabilizer states. A sampling process identical to equations (6.1) and (6.2) results in the number of samples required to be proportional to the square of the following normalization constant:

Definition 6.3. *The* ***robustness of magic*** $\mathcal{R}(\rho)$ *of an* n*-qubit state* ρ *is the outcome of a convex optimization program over real vectors* $\vec{q}$*:*

$$\mathcal{R}(\rho) = \min_{\vec{q}} \sum_i |q_i| \ \textit{s.t.}\ \rho = \sum_i q_i \left|\phi_i\right\rangle \left\langle\phi_i\right| \ \textit{and} \ \sum_i q_i = 1,$$

where $\{\left|\phi_i\right\rangle\}$ *are the* n*-qubit stabilizer states.*

*When $\mathcal{R}(\rho) = 1$ (the minimum value) then ρ is a **stabilizer mixture**, since then the vector $\vec{q}$ is a probability distribution.*

Due to the sheer number of stabilizer states, evaluating $\mathcal{R}(\rho)$ for even small n is very expensive. As stated in [Bennink&17], evaluating the cost function for 3-qubit unitaries is impractical, although the performance can be improved for diagonal gates [SC19].

The performance of stabilizer propagation gives a lens for the non-Cliffordness of channels, studied extensively in [SC19]. In section 6.6 we expand on this work by modifying the protocol to support all **postselective channels** which include all trace preserving channels and all 'reasonable' non-trace-preserving channels. There, we prove the following theorem:

Theorem 6.4. *Let Λ be a postselective channel and let $\bar{\phi}_\Lambda$ be the channel's normalized Choi state. Λ does not increase the number of samples required for stabilizer propagation if and only if $\mathcal{R}(\bar{\phi}_\Lambda) = 1$.*

This establishes simple and flexible criteria for when a circuit component does not increase the runtime of stabilizer propagation: states ρ are cheap when $\mathcal{R}(\rho) = 1$ and channels Λ are cheap if $\mathcal{R}(\bar{\phi}_\Lambda) = 1$.

6.2.2 Observables

Observables encountered in practice are usually computational basis measurements, or operators with bounded norm that can be expressed as sums of not too many Pauli matrices. Sometimes these observables are marginal: many of the qubits are not measured and traced out. Tracing out corresponds to measuring the identity observable, a kind of Pauli observable.

Stabilizer propagation outputs the inner product of the final observable with a stabilizer state. For all of the observables above, calculating inner products with stabilizer states is efficient: inner products with Pauli matrices can be obtained in n^2 time and marginal inner products with other stabilizer states in n^3 time [AG04]. Crucially, these inner products remain bounded by the eigenvalues of the observable and thereby do not exponentially increase the range of the distribution.

Schrödinger propagation, which outputs the inner product with a Pauli matrix, does not have this property: although inner products between Pauli matrices are trivial to compute, the maximum inner product grows like 2^n. Therefore, Schrödinger propagation is only viable when we are interested in the probability of measuring a particular state and only a constant number of discarded qubits. On the other hand, there exist contrived observables that only Schrödinger propagation can handle. If the observable is the tensor product of many non-stabilizer states, then neither Heisenberg propagation nor stabilizer propagation runs efficiently. (Indeed, calculating inner products of stabilizer states with tensor products of many non-stabilizer states is a key slow step in stabilizer rank techniques [BG16, Bravyi&18].)

Heisenberg propagation applies the sampling method (1) (2) to the observable E, so cost is measured by $\mathcal{D}(E)$. The following facts, proven in [HC16], show that Heisenberg propagation can handle the observables most common in quantum circuits.

Proposition 6.5. $\mathcal{D}(\sigma) = 1$ *for* $\sigma \in \mathcal{P}_n$.

Proposition 6.6. *If $|\phi\rangle$ is a stabilizer state, then $\mathcal{D}(|\phi\rangle\langle\phi|) = 1$.*

Proposition 6.7. *$\mathcal{D}$ is multiplicative: $\mathcal{D}(A \otimes B) = \mathcal{D}(A) \cdot \mathcal{D}(B)$.*

6.2.3 Hyper-Octahedral States

A central observation of this work is that Pauli matrix decompositions can produce similar simulational power as decompositions over stabilizer states. Here we show that despite their simplicity, Pauli matrix decompositions are *more* powerful with regards to the input state of the circuit. The number of samples required for Heisenberg propagation does not depend at all on the input state (19). For Schrödinger propagation we observe:

1. there exist states supported by Schrödinger propagation unsupported by stabilizer propagation, and
2. sufficiently depolarized states can actively decrease the number of samples required.

From the definition of the stabilizer norm, $\mathcal{D}$ can be viewed as the L1 norm of the Bloch vector $\vec{x}$ of ρ. The equation $||\vec{x}||_1 \leq 1$ defines the surface and interior of a hyper-octahedron, motivating the following definition.

Definition 6.8. ***Hyper-octahedral states*** *ρ satisfy $\mathcal{D}(\rho) \leq 1$. These states do not increase the number of samples for Schrödinger propagation.*

To see (B), we simply observe that the interior of the octahedron satisfies $\mathcal{D}(\rho) = ||\vec{x}||_1 < 1$. $\mathcal{D}$ is minimized at the n-qubit maximally mixed state where $\mathcal{D}(I/2^n) = 1/2^n$. The following result, proved in [HC16], shows that all stabilizer mixtures are hyper-octahedral.

Proposition 6.9. *For states ρ, $\mathcal{D}(\rho) \leq \mathcal{R}(\rho)$.*

This fact classifies mixed states into three categories: stabilizer mixtures, non-stabilizer hyper-octahedral states, and magic states. For the single qubit, the first two categories coincide (the qubit stabilizer polytope is an octahedron). We plot a cross-section of the two-qubit Bloch sphere in FIG. 6.1, showing that all of these categories are non-empty. FIG. 6.2 shows the relative quantity of these states according to the Hilbert-Schmidt measure [ZS03]. Stabilizer mixtures occupy a tiny fraction of all mixed states, whereas more than half are hyper-octahedral.

From the standpoint of quantum resource theories, hyper-octahedral states are interesting because they are similar to the 'bound' states discussed in [VFGE12, HWVE14, DH15, ACB12]: they contain non-stabilizer mixed states that can be efficiently simulated. But unlike $\mathcal{R}$, tracing out qubits can increase $\mathcal{D}$. Hadamard eigenstates $|H\rangle$ are magic states that let Clifford circuits attain universal quantum computation, but $|H\rangle \otimes (I/2)$ is hyper-octahedral. Hyper-octahedral states are not bound for magic state distillation in the same sense as those in [VFGE12]: there are operations that can be simulated efficiently by stabilizer propagation that increase $\mathcal{D}$. Schrödinger propagation cannot simulate operations that increase $\mathcal{D}$.

6.2.4 Channel Classification

While the classification of states gave rise to only three categories, the classification of channels is not so simple. FIG. 6.3 shows eight categories, all of which are non-empty. Here are examples of each:

M Non-Clifford unitaries, such as the T gate.

CSH Clifford unitaries, measuring a qubit in a Pauli basis (without discarding it), and very

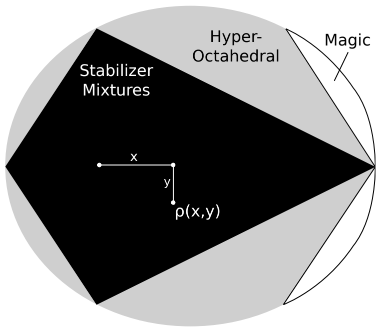


Figure 6.1: Visualization of a cross section of the two-qubit Bloch sphere, given by: $\rho(x,y) = \frac{\sigma_{II}}{4} + x(\sigma_{XX} + \sigma_{ZZ} - \sigma_{YY}) + y(\sigma_{ZI} + \sigma_{IZ})$

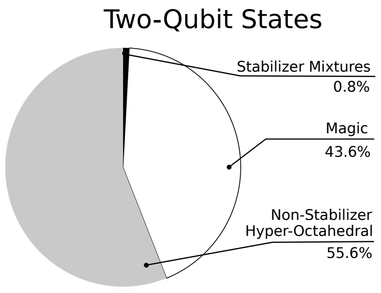


Figure 6.2: Relative quantity of two-qubit mixed states, based on one million samples via the Hilbert-Schmidt measure. Hyper-octahederal states are plentiful for two-qubits, despite not existing for the single qubit.

depolarized non-Clifford unitaries.

SH Mildly depolarized non-Clifford unitaries, e.g. the T gate with fidelity $0.551 \lessapprox f \leq 2^{-1/2}$ (FIG. 6.5).

C Most adaptive Clifford gates: gates performed based on the outcome of a measurement (Proposition 6.15).

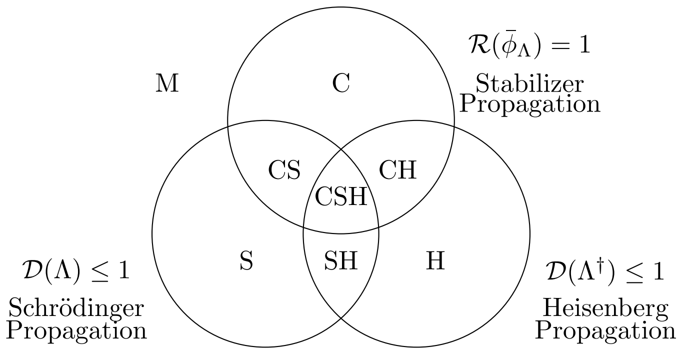
H Any non-Pauli reset channel (Proposition 6.14).

CH Pauli reset channels [Bennink&17].

S, CS Channels adjoints for H, HC, respectively.

To obtain the relative proportions of these categories akin to FIG. 6.2 we leverage channel-state duality. Our definition of postselective channels in section 6.5 is specifically chosen to make the correspondence between two-qubit mixed states and qubit-to-qubit channels a bijection. We sample states according to the Hilbert-Schmidt measure and classify their corresponding channels. Most channels in practice are either unital, trace preserving or both. It is not obvious how to restrict sampling to these measure-zero subspaces. Instead, we sample from the full Hilbert Schmidt measure, and then project onto the Bloch-subspaces corresponding to unital and/or trace preserving channels.

FIG. 6.4 shows the resulting proportions. For qubit-to-qubit channels, Pauli propagation techniques permit simulation of a significant fraction of the circuit components which are a superset of those simulable by stabilizer propagation. As before, it is not clear that this demonstrates that Pauli propagation is significantly more useful in practice, since most quantum circuits are dominated by a few specific types channels.



$\mathcal{R}(\bar{\phi}_\Lambda) = 1$
Stabilizer Propagation
M
C
CS
CH
CSH
S
SH
H
$\mathcal{D}(\Lambda) \leq 1$
Schrödinger Propagation
$\mathcal{D}(\Lambda^\dagger) \leq 1$
Heisenberg Propagation

Figure 6.3: A Venn Diagram of quantum channels that illustrates our naming convention. The channels not efficient under any strategy are category M.

In the following we give evidence for the above examples. To do so, we phrase $\mathcal{D}(\Lambda)$ in terms of the Pauli transfer matrix of Λ.

Definition 6.10. *The Pauli Transfer Matrix (PTM) of a quantum channel Λ taking n qubits to m qubits has elements $(R_\Lambda)_{ij} = 2^{-m}\, Tr(\sigma_i \Lambda(\sigma_j))$ such that $\Lambda(\rho) = 2^{-n} \sum_{i,j} (R_\Lambda)_{ij} \sigma_i\, Tr(\rho\sigma_j)$. We take $\sigma_1 = I$.*

Intuitively, the columns of R_Λ are the Bloch vectors of $\Lambda(\sigma_i)$. The following observations are useful and trivial to prove.

Proposition 6.11. *$D(\Lambda) = \|R_\Lambda\|_1$, where $\|\cdot\|_1$ is the induced L1-norm, i.e. the largest column L1-norm.*

Proposition 6.12. $R_\Lambda^T = R_{\Lambda^\dagger}$

Corollary 6.13. *$D(\Lambda^\dagger) = \|R_\Lambda\|_\infty$, where $\|\cdot\|_\infty$ is the induced L∞-norm, i.e. the largest row L1-norm.*

The PTM of a Clifford gate is a signed permutation matrix and the PTMs of Pauli basis measurements are signed permutations of diag$(1, 1, 0, 0)$. Their Choi states are also readily shown the be stabilizer mixtures, so these channels are CSH as claimed.

6.2.5 Depolarized Rotations

Many useful unitaries take the form $e^{-i\theta\sigma/2}$ with $\sigma \in \mathcal{P}_n$. Via some Clifford transformations these can be obtained from the qubit unitary $e^{-i\theta\sigma_Z/2}$. In this section we consider composing this unitary with depolarizing noise, obtaining a family of channels $\Lambda_{\theta,f}$ where f is the fidelity.

The PTMs of the unitary $e^{-\theta\sigma_Z/2}$ and depolarizing noise are respectively:

$$R_\theta = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad R_f = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & f & 0 \\ 0 & 0 & 0 & f \end{bmatrix}$$

Composing these two channels simply involves multiplying the two PTMs, resulting in:

$$R_{\Lambda_{f,\theta}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & f\cos\theta & -f\sin\theta & 0 \\ 0 & f\sin\theta & f\cos\theta & 0 \\ 0 & 0 & 0 & f \end{bmatrix} \tag{6.20}$$

$$\mathcal{D}(\Lambda_{f,\theta}) = \mathcal{D}(\Lambda^\dagger_{r,\theta}) = \max\left(1, f|\cos\theta| + f|\sin\theta|\right) \tag{6.21}$$

We plot the family in FIG. 6.5, showing that there are channels simulable by Pauli propagation methods that are not simulable by stabilizer propagation. The boundary of $\mathcal{D} \leq 1$ given by $|\cos\theta| + |\sin\theta| = 1$ forms a diamond. The depolarized T gate becomes SH when $f \leq 2^{-1/2} \approx 0.707$, and becomes CSH when $f \lessapprox 0.551$.

6.2.6 Reset Channels

Pauli reset channels can be described as projecting into the +1 eigenspace of some $\sigma \in \mathcal{P}_n$ as in [Bennink&17]. Alternatively we can use Clifford transformations to convert σ to σ_Z, converting the channel to tracing out a single qubit and replacing it with $|0\rangle$. We generalize the notion of a reset channel Λ_ρ to tracing out n qubits and replacing them with an n-qubit state ρ. To make the channel trace preserving we write $\Lambda_\rho(\sigma) = \mathrm{Tr}(\sigma) \cdot \rho$.

Proposition 6.14. *If Λ_ρ is a reset channel, $\mathcal{D}(\Lambda^\dagger) = 1$.*

Proof. The entries of the PTM of Λ_ρ are the following:

$$(R_{\Lambda_\rho})_{ij} = 2^{-n}\mathrm{Tr}(\sigma_i \Lambda_\rho(\sigma_j)) = \begin{cases} 2^{-n}\mathrm{Tr}(\sigma_i \rho) & \sigma_j = I \\ 0 & \sigma_j \neq I \end{cases}$$

All rows except for the first are zero. The entries are bounded $-1 \leq 2^{-n}\mathrm{Tr}(\sigma_i \rho) \leq 1$ and the top left entry is 1. Thus the maximum column L1 norm is 1, and Proposition 6.13 tells us that $\mathcal{D}(\Lambda^\dagger) = 1$. $\square$

Observe that the first row is actually the Bloch vector of ρ (including the identity component) scaled by 2^n. So unless ρ is the maximally mixed state the first row's L1 norm is > 1, so the channel is not simulable by Schrödinger propogation, and its adjoint is not simulable by Heisenberg propogation.

The Choi state of Λ_ρ is $\frac{I}{2^n} \otimes \rho$, so Λ_ρ is simulable by stabilizer propagation when ρ is a stabilizer mixture.

6.2.7 Adaptive Channels

Adaptive channels consist of making a σ_Z measurement, and then conditionally applying a channel based on the measurement outcome. While Pauli propagation techniques are stronger than stabilizer propagation in many respects, adaptive channels are their key weak point. This remains true even if the measured qubit is *not* discarded, so we are not conflating the cost of tracing out qubits with the cost of adaptivity.

Proposition 6.15. *Let Λ be a quantum channel with PTM R_Λ. Let $A(\Lambda)$ be the adaptive channel that conditionally applies Λ based on a σ_Z measurement on some qubit that is not discarded post-measurement. Then:*

$$\begin{aligned}\mathcal{D}(A(\Lambda)) &= 1 + \max_i \sum_{i \neq j} |R_{ij}| \leq 1 + \mathcal{D}(\Lambda^\dagger) && (6.22)\\ \mathcal{D}(A(\Lambda)^\dagger) &= 1 + \max_j \sum_{i \neq j} |R_{ij}| \leq 1 + \mathcal{D}(\Lambda) && (6.23)\end{aligned}$$

Corollary 6.16. *$A(\Lambda)$ is supported by Pauli propagation methods if and only if the PTM of Λ is diagonal.*

So Pauli propagation methods are not 'closed under adaptivity': $A(\Lambda)$ can be non-simulable even if Λ is simulable. Stabilizer propagation on the other hand *is* closed under adaptivity.

Proof of Proposition 6.15. Let Λ take n qubits to m qubits. The measurement of the first qubit projects into the space spanned by I, σ_Z on the first qubit.

$$\begin{aligned}A(\Lambda)(I \otimes \sigma_j) &= \begin{pmatrix}\sigma_j & 0\\ 0 & \Lambda(\sigma_j)\end{pmatrix} &= \begin{pmatrix}\sigma_j & 0\\ 0 & \sum_k R_{kj}\sigma_k\end{pmatrix}\\ A(\Lambda)(\sigma_Z \otimes \sigma_j) &= \begin{pmatrix}\sigma_j & 0\\ 0 & -\Lambda(\sigma_j)\end{pmatrix} &= \begin{pmatrix}\sigma_j & 0\\ 0 & -\sum_k R_{kj}\sigma_k\end{pmatrix}\end{aligned}$$

The output remains in the space spanned by I, σ_Z on the first qubit, so the only nonzero entries of the PTM are:

$$\frac{1}{2^{m+1}}\text{Tr}\left((I\otimes\sigma_i)\cdot A(\Lambda)(I\otimes\sigma_j)\right) = \frac{1}{2}(\delta_{ij}+R_{ij})$$
$$\frac{1}{2^{m+1}}\text{Tr}\left((\sigma_Z\otimes\sigma_i)\cdot A(\Lambda)(I\otimes\sigma_j)\right) = \frac{1}{2}(\delta_{ij}-R_{ij})$$
$$\frac{1}{2^{m+1}}\text{Tr}\left((I\otimes\sigma_i)\cdot A(\Lambda)(\sigma_Z\otimes\sigma_j)\right) = \frac{1}{2}(\delta_{ij}-R_{ij})$$
$$\frac{1}{2^{m+1}}\text{Tr}\left((\sigma_Z\otimes\sigma_i)\cdot A(\Lambda)(\sigma_Z\otimes\sigma_j)\right) = \frac{1}{2}(\delta_{ij}+R_{ij})$$

Applying the definition of channel stabilizer norm:

$$\begin{aligned}\mathcal{D}(A(\Lambda)) &= \frac{1}{2}\max_i\sum_j\left(|\delta_{ij}+R_{ij}|+|\delta_{ij}-R_{ij}|\right)\\ &= 1+\max_i\sum_{i\neq j}|R_{ij}| \quad \square\end{aligned}$$

6.3 Numerical Results

Algorithms based on Monte Carlo averages have favorable memory requirements and admit massive parallelization. We demonstrate these practical advantages via the performance of a GPU implementation written in CUDA [Wilt13].

Following previous tests of near-Clifford algorithms [Bravyi&18] we simulate the Quantum Approximate Optimization Algorithm (QAOA) on E3LIN2 [FG14]. We generate m random independent linear equations acting on three qubits $a, b, c \in [n]$ of the form $x_a \oplus x_b \oplus x_c = d_j$ for $j \in [m]$. Each qubit appears in at most $m/10$ equations. Let $\sigma_Z^{(j)} = \sigma_{Z,a}\otimes\sigma_{Z,b}\otimes\sigma_{Z,c}$ be σ_Z acting on the qubits corresponding to equation j. Our goal is to estimate the observable

$$C = \frac{1}{2}\sum_{j\in[m]}(-1)^{d_j}\sigma_Z^{(j)}$$

since $C + m/2$ is the number of satisfied equations. We estimate the expectation of this observable with the state

$$|\gamma, \beta\rangle = e^{-i\beta B} e^{-i\gamma C} |+^{\otimes n}\rangle$$

where $B = \sum_{i\in[n]} \sigma_{X,i}$ and $\beta = \pi/4$.

Heisenberg propagation is most appropriate for this problem, with performance $\mathcal{D}(C) = m/2$ and $\mathcal{D}(e^{\pm i\gamma\sigma_Z^{(j)}}) = |\sin\gamma| + |\cos\gamma|$. Although the unitary $e^{\pm i\gamma\sigma_Z^{(j)}}$ appears m times in the circuit, at most $3(m/10-1)+1$ can act non-trivially on any term in C. Thus the accuracy of the simulation is given by:

$$\varepsilon_{\text{Heis}} = \frac{m}{\sqrt{2N}} \cdot \sqrt{\ln\frac{2}{\delta}} \cdot (|\sin\gamma| + |\cos\gamma|)^{3(m/10-1)+1}$$

As pointed out by [Bravyi&18], a protocol by van den Nest [VendenNest09] gives an *efficient* Monte Carlo protocol for estimating $\langle C\rangle$ with error $\varepsilon_{\text{Nest}} = \frac{m}{\sqrt{N}} \cdot \sqrt{\ln\frac{2}{\delta}}$. We utilize the van den Nest estimate $\langle C\rangle_{\text{Nest}}$ to verify the Heisenberg propagation estimate $\langle C\rangle_{\text{Heis}}$.

Writing effective CUDA applications demands careful memory management. Implementing stabilizer propagation via the Aaronson-Gottesman tableau algorithm would be a serious computer engineering task. In contrast, the increased simplicity of Pauli propagation algorithms permits a very simple implementation. We furthermore utilize bitwise operations to express the logic in a compact and efficient manner. Despite the better scaling it was ultimately necessary to also implement the van den Nest protocol in CUDA due to the sheer performance improvement over a Python implementation.

For every data point we collected $2^{30} \approx 1$ billion samples in 25 minutes using a laptop GPU (GeForce GTX 1050 Ti). We fix $n = 32$ qubits and $\delta = 0.01$ throughout, and vary γ for a single instance with $m = 40$ equations (Figure 6.6, top). Then we set $\gamma = \pi/8$, maximizing $\mathcal{D}(e^{\pm i\gamma\sigma_Z^{(j)}})$ at $\sqrt{2}$, and perform a scaling analysis with instances up to $m = 80$ (Figure 6.6, bottom).

Hoeffding's inequality gives a worst-case upper bound for the accuracy of the estimate, potentially very far from the actual error. This is the case here: for $m \gtrapprox 60$ we have $\varepsilon_{\text{Heis}} \geq 1$ predicting that $\langle C \rangle_{\text{Heis}}$ is useless, but we observe that the actual error is ≤ 0.01. Furthermore the actual accuracy does not seem to scale proportionally with $\varepsilon_{\text{Heis}}$ as we vary γ and m.

6.4 Conclusion

Recent interest in near-Clifford simulation [Bennink&17, PWB15, BG16, Bravyi&18] and the (non-)contextuality of Clifford circuits [HWVE14, Raussendorf&15, Bermejo-Vega&16, Delfosse&16] demonstrates that there is still much to be learned about embedding symmetry into Hilbert space. The qubit Clifford group appears different from the Clifford group in odd dimensions, where the discrete Wigner function [Gross06] has led to well-behaved resource theories [HC16, VFGE12, VMGE13] and associated simulation algorithms [PBG17]. We observe that the qubit analogue of the Wigner function is just a Bloch vector, and our analysis of the resulting algorithms sheds further light into the differences between the even and odd-dimensional cases. Furthermore, the simplicity of Pauli propagation algorithms along with their improved performance for many quantum channels make them a compelling addition to near-Clifford simulation techniques.

6.5 Postselective Quantum Channels

In this section we define postselective quantum channels. These include all trace preserving channels, and all 'sensible' non-trace-preserving channels. Furthermore, there is a bijection between postselective channels taking $\mathcal{H}^A$ to $\mathcal{H}^B$ and density operators on $\mathcal{H}^B \otimes \mathcal{H}^A$, which is essential for FIG. 5 and Theorem 6.4.

Completely positive maps Λ with $0 \leq \text{Tr}(\Lambda(\rho)) \leq \text{Tr}(\rho)$ have an operational interpretation: the associated channels can 'fail' or 'abort' the computation by yielding 0. For example, let Λ be the channel that measures in the σ_Z basis and postselects on obtaining $|0\rangle$. Then $\Lambda(|1\rangle\langle 1|) = 0$, and $\Lambda(|+\rangle\langle +|) = \frac{1}{2}|0\rangle\langle 0|$.

Definition 6.17. *Let Λ be a completely positive map from $\mathcal{H}^A$ to $\mathcal{H}^B$. Let $|Bell_A\rangle \in \mathcal{H}^A \otimes \mathcal{H}^A$ be a Bell state for $\mathcal{H}^A$, i.e. if $\{|i\rangle\}$ are a basis for $\mathcal{H}^A$ then:*

$$|Bell_A\rangle = \frac{1}{\sqrt{dim(\mathcal{H}^A)}} \sum_i |i\rangle \otimes |i\rangle \tag{6.24}$$

*The **un-normalized Choi state** ϕ_Λ of Λ is the resulting state when Λ is applied to one half of $|Bell_A\rangle$.*

$$\phi_\Lambda = (\Lambda \otimes I)(|Bell_A\rangle\langle Bell_A|) \in \mathcal{H}^B \otimes \mathcal{H}^A \tag{6.25}$$

*$Tr(\phi_\Lambda)$ of Λ can be less than 1 if Λ is not trace preserving. Let $\bar{\phi}_\Lambda = \phi_\Lambda / Tr(\phi_\Lambda)$ be the **normalized Choi state** with trace 1. This distinction is crucial.*

To calculate the output of a channel $\Lambda(\rho)$ given its Choi state ϕ_Λ we compute:

$$\Lambda(\rho) = \text{dim}(\mathcal{H}^A) \cdot \text{Tr}_A\left(\phi_\Lambda(I \otimes \rho^T)\right) \tag{6.26}$$

Crucially we use ϕ_Λ, not $\bar{\phi}_\Lambda$. To explain why, consider a Choi state $\bar{\phi}_\Lambda = |00\rangle\langle 00|$. If we apply the equation above to $\bar{\phi}_\Lambda$ we obtain $\Lambda(\rho) = 2 \cdot |0\rangle\langle 0| \cdot \langle 0| \rho^T |0\rangle$, so $\Lambda(|0\rangle\langle 0|) = 2|0\rangle\langle 0|$ which makes no sense. The fact that ϕ_Λ is under-normalized takes care of this constant.

Given a normalized Choi state $\bar{\phi}_\Lambda$, e.g. $|00\rangle\langle 00|$, how do we determine ϕ_Λ? In general, ϕ_Λ is not unique. Consider channels $\Lambda(\rho)$ and $\Lambda'(\rho) = p \cdot 0 + (1-p)\Lambda(\rho)$, i.e. Λ' aborts with probability p and otherwise applies Λ. Both channels have the same $\bar{\phi}_\Lambda$, but $\phi_{\Lambda'} = p\phi_\Lambda$.

However, Λ' is somewhat silly: aborting the computation should be a tool for postselection and should not happen regardless of the input state. For all sensible channels there should exist an input state where the postselection succeeds with probability 1. To associate all $\bar{\phi}_\Lambda$ to a unique ϕ_Λ we restrict our attention to the following quantum channels.

Definition 6.18. *A completely positive map Λ represents a **postselective quantum channel** if:*

1. *Λ is trace-non-increasing: for all positive-semidefinite ρ, Λ satisfies $0 \leq Tr(\Lambda(\rho)) \leq Tr(\rho)$,*
2. *the postselection can be satisfied: there exists a normalized pure state $|\psi\rangle$ such that $Tr(\Lambda(|\psi\rangle\langle\psi|)) = 1$.*

Among these channels we can uniquely obtain ϕ_Λ from $\bar{\phi}_\Lambda$, so there is a bijection between normalized mixed states and postselective quantum channels. Let $\phi_\Lambda = p_\Lambda \bar{\phi}_\Lambda$. Then:

$$\frac{1}{p_\Lambda} = \dim(\mathcal{H}^A) \cdot \max_{|\psi\rangle} \mathrm{Tr}\left(\bar{\phi}_\Lambda (I \otimes (|\psi\rangle\langle\psi|)^T)\right) \tag{6.27}$$

For example, if $\bar{\phi}_\Lambda = |00\rangle\langle 00|$ then $|\psi\rangle = |0\rangle$ maximizes $1/p_\Lambda$ at 2, so $\phi_\Lambda = \frac{1}{2}|00\rangle\langle 00|$ and $\Lambda(\rho) = |0\rangle\langle 0| \cdot \langle 0|\rho^T|0\rangle$. Incidentally, p_Λ is the probability of postselection succeeding when Λ is applied to the Bell state.

6.6 Simulating Channels whose Choi States are Stabilizer Mixtures

In this section we prove Theorem 6.4: Stabilizer propagation can efficiently simulate a quantum channel Λ if and only if the robustness of its Choi state $\mathcal{R}(\phi_\Lambda)$ is 1. This criterion also captures postselective quantum channels, and thereby all sensible non-trace-preserving channels.

All results of [SC19] generalize neatly to postselective channels. Assuming familiarity with the work, the definition of magic capacity $\mathcal{C}(\Lambda)$ remains identical and the channel robustness $\mathcal{R}_*(\Lambda)$ can be obtained via convex optimization over linear combinations of *un-normalized* Choi states of stabilizer channels. It is easy to see that Theorem 2, $\mathcal{R}(\phi_\Lambda) \leq \mathcal{C}(\Lambda) \leq \mathcal{R}_*(\Lambda)$, still holds. Their Lemma 2, $\mathcal{R}(\bar{\phi}_\Lambda) = 1$ implies $\mathcal{C}(\Lambda) = 1$, is our Theorem 3.2.

Theorem 6.4 (rephrased). *Consider a postselective channel $\Lambda : \mathcal{H}^A \to \mathcal{H}^B$. The following statements are equivalent.*

1. *The channel's normalized Choi state $\bar{\phi}_\Lambda$ is a probabilistic mixture of stabilizer states, so $\mathcal{R}(\bar{\phi}_\Lambda) = 1$.*

2. *If Λ is applied to any subset of the qubits of any large stabilizer state $|\psi\rangle$, one can efficiently sample from a probability distribution over stabilizer states and 'abort' whose mean is the resulting state.*

Proof: 2. implies 1. Say a channel Λ is simulable. Apply Λ to one half of the state $|\text{Bell}_A\rangle$, a stabilizer state. The resulting Choi state is probabilistic mixture of stabilizer states and 'abort':

$$\phi_\Lambda = p_0 \cdot 0 + \sum_i p_i |\phi_i\rangle \langle\phi_i| \tag{6.28}$$

$$\bar{\phi}_\Lambda = \frac{\phi_\Lambda}{\text{Tr}(\phi_\Lambda)} = \frac{1}{1-p_0} \sum_i p_i |\phi_i\rangle \langle\phi_i| \tag{6.29}$$

Since $p_i/(1-p_0)$ is a probability distribution, $\bar{\phi}_\Lambda$ is also a probabilistic mixture of stabilizer states. □

Proof: 1. implies 2. Say we are given

$$\bar{\phi}_\Lambda = \sum_i p_i \bar{\phi}_{\Gamma_i} \tag{6.30}$$

where $\bar{\phi}_{\Gamma_i}$ are *pure* stabilizer states with corresponding pure operations Γ_i. Our goal is to obtain an efficiently computable probability distribution over stabilizer states and 'abort' of Λ applied to some subset of the qubits of a stabilizer state $|\psi\rangle$. The channel acts on a constant number of qubits, so we can compute anything we want about it. The stabilizer state, however, may live in a Hilbert space of exponential dimension. Using $\phi_\Lambda = p_\Lambda \bar{\phi}_\Lambda$:

$$\phi_\Lambda = p_\Lambda \sum_i \frac{p_i}{p_{\Gamma_i}} \phi_{\Gamma_i} \tag{6.31}$$

All of the quantities p_Λ, p_i and p_{Γ_i} can be obtained quickly. Now we apply (6.26), but we extend Λ and Γ_i from the constant size Hilbert space to $\tilde{\Lambda}$ and $\tilde{\Gamma}_i$ which act on the large Hilbert space containing $|\psi\rangle$.

$$\tilde{\Lambda}(|\psi\rangle \langle\psi|) = p_\Lambda \sum_i \frac{p_i}{p_{\Gamma_i}} \tilde{\Gamma}_i(|\psi\rangle \langle\psi|) \tag{6.32}$$

Crucially, $\tilde{\Gamma}_i(|\psi\rangle\langle\psi|)$, the right-hand side of (6.26), is an inner product between pure stabilizer states ϕ_Γ and $|\psi\rangle$ and is therefore a pure stabilizer state that can be computed in polynomial time. Since $\tilde{\Gamma}_i$ may be non-trace-preserving, $\tilde{\Gamma}_i(|\psi\rangle\langle\psi|)$ may not be normalized. Let $|\gamma_i\rangle$ be the normalized pure stabilizer state:

$$|\gamma_i\rangle\langle\gamma_i| = \tilde{\Gamma}_i(|\psi\rangle\langle\psi|)\Big/\mathrm{Tr}(\tilde{\Gamma}_i(|\psi\rangle\langle\psi|)) \tag{6.33}$$

We write $\tilde{\Lambda}(|\psi\rangle\langle\psi|)$ as a weighted sum over normalized pure stabilizer states $|\gamma_i\rangle$.

$$\tilde{\Lambda}(|\psi\rangle\langle\psi|) = \sum_i p_\Lambda \frac{p_i}{p_{\Gamma_i}} \mathrm{Tr}(\tilde{\Gamma}_i(|\psi\rangle\langle\psi|)) \cdot |\gamma_i\rangle\langle\gamma_i| \tag{6.34}$$

The weights are positive and one can see that they sum to less than 1 by taking the trace of both sides. Furthermore since $\bar{\phi}_{\Gamma_i}$ are pure stabilizer states, the number $\mathrm{Tr}(\tilde{\Gamma}_i(|\psi\rangle\langle\psi|))$ and stabilizer state $|\gamma_i\rangle\langle\gamma_i|$ are efficiently computable.

Thus, to simulate Λ acting on $|\psi\rangle$ we sample:

$$|\gamma_i\rangle\langle\gamma_i| \quad \text{w.p.} \quad p_\Lambda \frac{p_i}{p_{\Gamma_i}} \mathrm{Tr}(\tilde{\Gamma}_i(|\psi\rangle\langle\psi|)) \tag{6.35}$$

$$0 \quad \text{w.p.} \quad 1 - \sum_i p_\Lambda \frac{p_i}{p_{\Gamma_i}} \mathrm{Tr}(\tilde{\Gamma}_i(|\psi\rangle\langle\psi|)).\ \Box \tag{6.36}$$

Table: Circuit components that can be simulated efficiently

	[Bennink&17, HC16] **Stabilizer propagation**
What input states are efficient to simulate?	Stabilizer mixtures
Depolarized T gate	Efficient when fidelity $\lessapprox 0.551$
Reset channels	Pauli reset channels efficient
Adaptive gates	Adaptive Cliffords efficient
Marginal observables	Efficient
Pauli observables	

	This work: Pauli propagation algorithms	
	Heisenberg propagation	**Schrödinger propagation**
What input states are efficient to simulate?	Any separable state, Stabilizer mixtures	Hyper-octahedral states, Noisy states reduce runtime
Depolarized T gate	Efficient when fidelity $\leq 2^{-1/2} \approx 0.707$	
Reset channels	All reset channels efficient	Generally inefficient
Adaptive gates	Generally inefficient	
Marginal observables	Efficient	Generally inefficient
Pauli observables		

Table 6.1: Summary of the results of Section 6.2. All algorithms take polynomial time to sample, but the number of samples scales exponentially in the number of *inefficient* circuit components. *Efficient* components do not increase runtime.

Relative Quantity of Qubit-To-Qubit Channels

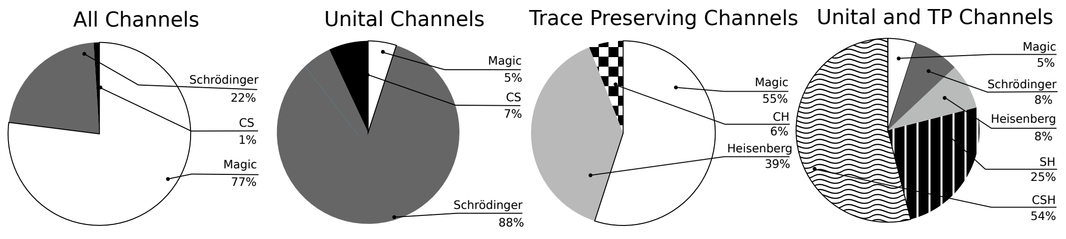


FIG. 4: Relative quantity of of qubit-to-qubit quantum channels, based on 100 000 random two-qubit density matrices obtained via the Hilbert-Schmidt measure. After obtaining the PTM we optionally set the first column or row to [1,0,0,0] to enforce unitality or trace preservation respectively [Greenbaum15]. We utilize the cvxpy library [DB16] to compute $\mathcal{R}$ and use a tolerance of 10^{-6} throughout.

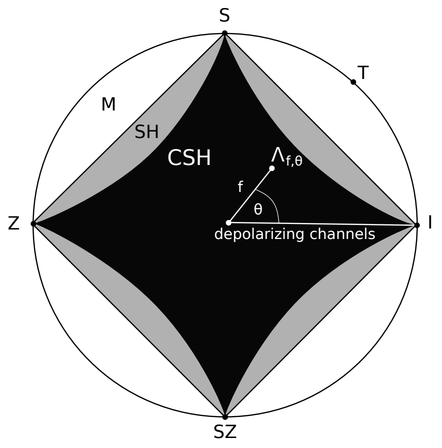


Figure 6.5: Qubit quantum channels $\Lambda_{f,\theta}$ obtained by an application of the unitary $e^{-i\theta\sigma_Z/2}$ followed by depolarizing noise with fidelity f. The region simulable by Pauli propagation (SH) is larger than that simulable by stabilizer propagation (CSH).

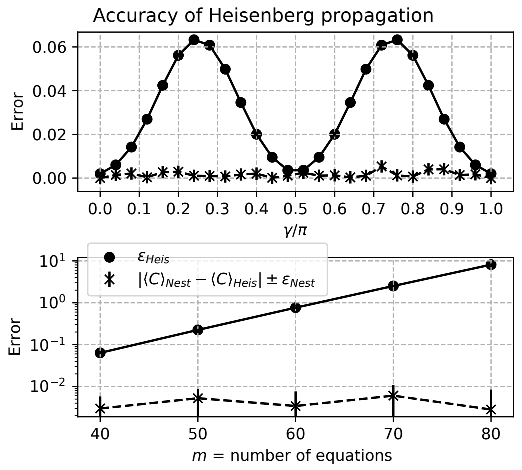


Figure 6.6: Comparison of Hoeffding error bound $\varepsilon_{\text{Heis}}$ to error as estimated by the van den Nest protocol $|\langle C\rangle_{\text{Nest}} - \langle C\rangle_{\text{Heis}}|$ for 32 qubits. Top: $m = 40$ and varying γ. Bottom: $\gamma = \pi/8$ and varying m.

Bibliography

[Zhu&21] Qingling Zhu et al. **Quantum Computational Advantage via 60-Qubit 24-Cycle Random Circuit Sampling** arXiv:2109.03494 (2021)

[WT21] Pawel Wocjan, Kristan Temme **Szegedy Walk Unitaries for Quantum Maps** arXiv:2107.07365 (2021)

[Wu&21] Yulin Wu et al. **Strong quantum computational advantage using a superconducting quantum processor** arXiv:2106.14734 (2021)

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

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

[LT21] Lin Lin, Yu Tong **Heisenberg-limited ground state energy estimation for early fault-tolerant quantum computers** arXiv:2102.11340 (2021)

[Cam20] Earl T. Campbell **Early fault-tolerant simulations of the Hubbard model** arXiv:2012.09238 (2020)

[SHC20] Yuan Su, Hsin-Yuan Huang, Earl T. Campbell **Nearly tight Trotterization of interacting electrons** arXiv:2012.09194 Quantum 5, 495 (2020)

[ESP20] Alexander Engel, Graeme Smith, Scott E. Parker **A framework for applying quantum computation to nonlinear dynamical systems** arXiv:2012.06681 Physics of Plasmas 28, 062305 (2020)

[An&20] Dong An, Noah Linden, Jin-Peng Liu, Ashley Montanaro, Changpeng Shao, Jiasu Wang **Quantum-accelerated multilevel Monte Carlo methods for stochastic differential equations in mathematical finance** arXiv:2012.06283 Quantum 5, 481 (2020)

[Zhong&20] Han-Shen Zhong, et al. **Quantum computational advantage using photons** arXiv:2012.01625 Science, 372, 6545, (948-952) (2020)

[Chuang20] Isaac Chuang. **Grand unification of quantum algorithms.** Seminar presentation at IQC Waterloo. (2020)

[Wright&20] Lewis Wright, Fergus Barratt, James Dborin, George H. Booth, Andrew G. Green **Automatic Post-selection by Ancillae Thermalisation** arXiv:2010.04173 Phys. Rev. Research 3, 033151 (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 (2020)

[GST20] András Gilyén, Zhao Song, Ewin Tang **An improved quantum-inspired algorithm for linear regression** arXiv:2009.07268 (2020)

[JKKA20] Phillip W. K. Jensen, Lasse Bjørn Kristensen, Jakob S. Kottmann, Alán Aspuru-Guzik **Quantum Computation of Eigenvalues within Target Intervals** Quantum Science and Technology 6, 015004 arXiv:2005.13434 (2020)

[Rall20] Patrick Rall **Quantum Algorithms for Estimating Physical Quantities using Block-Encodings** Phys. Rev. A 102, 022408 arXiv:2004.06832 (2020)

[Rogg20] Alessandro Roggero **Spectral density estimation with the Gaussian Integral Transform** Phys. Rev. A 102, 022409 arXiv:2004.04889 (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)

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

[Seddon&20] James R. Seddon, Bartosz Regula, Hakop Pashayan, Yingkai Ouyang, Earl T. Campbell **Quantifying quantum speedups: improved classical simulation from tighter magic monotones** arXiv:2002.06181 PRX Quantum 2, 010345 (2020)

[Childs&19] Andrew M. Childs, Yuan Su, Minh C. Tran, Nathan Wiebe, Shuchen Zhu **A Theory of Trotter Error** Phys. Rev. X 11, 011020 arXiv:1912.08854 (2019)

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

[Google&19] Arute et al 'Quantum supremacy using a programmable superconducting processor' **Quantum supremacy using a programmable superconducting processor** Nature 574, 505–510 (2019)

[Lemi&19] Jessica Lemieux, Bettina Heim, David Poulin, Krysta Svore, Matthias Troyer **Efficient Quantum Walk Circuits for Metropolis-Hastings Algorithm** Quantum

4, 287 arXiv:1910.01659 (2019)

[Chak&19] Shouvanik Chakrabarti, Andrew M. Childs, Shih-Han Hung, Tongyang Li, Chunhao Wang, Xiaodi Wu **Quantum algorithm for estimating volumes of convex bodies** arXiv:1908.03903 QIP '20 (2019)

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

[Somma19] Rolando D. Somma **Quantum eigenvalue estimation via time series analysis** arXiv:1907.11748 New J. Phys. 21 123025 (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)

[Wie19] C. R. Wie **Simpler Quantum Counting** arXiv:1907.08119 Quantum Information & Computation, Vol.19, No.11&12, pp0967-0983 (2019)

[Berry&19] Dominic W. Berry, Andrew M. Childs, Yuan Su, Xin Wang, Nathan Wiebe **Time-dependent Hamiltonian simulation with L^1-norm scaling** arXiv:1906.07115 (2019)

[AAG19] Anurag Anshu, Itai Arad, David Gosset **Entanglement subvolume law for 2D frustration-free spin systems** arXiv:1905.11337 (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)

[RLCK19] Patrick Rall, Daniel Liang, Jeremy Cook, William Kretschmer **Simulation of Qubit Quantum Circuits via Pauli Propagation** arXiv:1901.09070 Phys. Rev. A 99, 062337 (2019)

[SC19] James Seddon, Earl Campbell **Quantifying magic for multi-qubit operations** arXiv:1901.03322 Proc. R. Soc. A 475 (2019)

[KLLP18] Iordanis Kerenidis, Jonas Landman, Alessandro Luongo, and Anupam Prakash **q-means: A quantum algorithm for unsupervised machine learning** arXiv:1812.03584 NIPS 32 (2018)

[Fan&18] Zheyong Fan, Jose Hugo Garcia, Aron W. Cummings, Jose Eduardo Barrios-Vargas, Michel Panhans, Ari Harju, Frank Ortmann, Stephan Roche **Linear Scaling Quantum Transport Methodologies** arXiv:1811.07387 Phys. Rev. B 89, 045123 (2018)

[HL18] Yifei Huang, Peter Love **Approximate stabilizer rank and improved weak simulation of Clifford-dominated circuits for qudits** arXiv:1808.02406 Phys. Rev. A 99, 052307 (2018)

[Bravyi&18] Sergey Bravyi, Dan Browne, Padraic Calpin, Earl Campbell, David Gosset, Mark Howard **Simulation of quantum circuits by low-rank stabilizer decompositions** arXiv:1808.00128 Quantum 3, 181 (2018)

[HG18] Markus Heinrich, David Gross. **Robustness of Magic and Symmetries of the Stabiliser Polytope** arXiv:1807.10296 Quantum 3, 132 (2018)

[HM18] Yassine Hamoudi, Frédéric Magniez **Quantum Chebyshev's Inequality and Applications** ICALP, LIPIcs Vol 132, pages 69:1-99:16 arXiv:1807.06456 (2018)

[Ambainis&18] Andris Ambainis, Kaspars Balodis, Jānis Iraids, Martins Kokainis, Krišjānis Prūsis, Jevgēnijs Vihrovs **Quantum Speedups for Exponential-Time Dynamic Programming Algorithms** arXiv:1807.05209 SODA '19 (2018)

[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 Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing (STOC 2019) Pages 193–204 (2018)

[LW18] Guang Hao Low, Nathan Wiebe **Hamiltonian Simulation in the Interaction Picture** arXiv:1805.00675 (2018)

[KSB18] Maria Kieferova, Artur Scherer, Dominic Berry **Simulating the dynamics of time-dependent Hamiltonians with a truncated Dyson series** arXiv:1805.00582 Phys. Rev. A 99, 042314 (2018)

[RC18] Alessandro Roggero, Joseph Carlson **Linear Response on a Quantum Computer** arXiv:1804.01505 Phys. Rev. C 100, 034610 (2018)

[Preskill18] John Preskill **Quantum computing in the NISQ era and beyond** arXiv:1801.00862 Quantum 2, 79 (2018)

[PBG17] Hakop Pashayan, Stephen D. Bartlett, David Gross **From estimation of quantum probabilities to simulation of quantum circuits** arXiv:1712.02806 Quantum 4, 223 (2017)

[Poulin&17] David Poulin, Alexei Kitaev, Damian S. Steiger, Matthew B. Hastings, Matthias Troyer **Quantum Algorithm for Spectral Measurement with Lower Gate Count** arXiv:1711.11025 Phys. Rev. Lett. 121, 010501 (2017)

[GAW17] András Gilyén, Srinivasan Arunachalam, Nathan Wiebe **Optimizing quantum optimization algorithms via faster quantum gradient computation** arXiv:1711.00465 SODA 2019 (2017)

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

[KP17] Iordanis Kerenidis, Anupam Prakash **Quantum gradient descent for linear systems and least squares** arXiv:1704.04992 Phys. Rev. A 101, 022316 (2017)

[Bennink&17] Ryan S. Bennink, Erik M. Ferragut, Travis S. Humble, Jason A. Laska, James J. Nutaro, Mark G. Pleszkoch, Raphael C. Pooser **Unbiased Simulation of Near-Clifford Quantum Circuits** arXiv:1703.00111 Phys. Rev. A 95, 062337 (2017)

[AA16] Yosi Atia, Dorit Aharonov **Fast-forwarding of Hamiltonians and exponentially precise measurements** Nature Communications volume 8, 1572 arXiv:1610.09619 (2016)

[Bermejo-Vega&16] Juan Bermejo-Vega, Nicolas Delfosse, Dan Browne, Cihan Okay, Robert Raussendorf. **Contextuality as a resource for qubit quantum computation** arXiv:1610.08529 Phys. Rev. Lett. 119, 120505 (2016)

[Delfosse&16] Nicolas Delfosse, Cihan Okay, Juan Bermejo-Vega, Dan Browne, Robert Raussendorf. **Equivalence between contextuality and negativity of the Wigner function for qudits** arXiv:1610.07093 New J. Phys. 19 123024 (2016)

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

[BK16] Fernando G.S.L. Brandao, Michael J. Kastoryano **Finite correlation length implies efficient preparation of quantum thermal states** arXiv:1609.07877 Communications in Mathematical Physics 365.1: 1-16 (2016)

[HC16] Mark Howard, Earl Campbell **Application of a resource theory for magic states to fault-tolerant quantum computing** Phys. Rev. Lett. 118, 090501 arXiv:1609.07488 (2016)

[CF16] Christopher T. Chubb, Steven T. Flammia **Approximate symmetries of Hamiltonians** arXiv:1608.02600 Journal of Mathematical Physics 58, 082202 (2016)

[Martinis&16] Martinis et al. **Characterizing Quantum Supremacy in Near-Term Devices** *Nature Physics 14, 595-600*, arXiv:1608.00263 (2016)

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

[KP16] Iordanis Kerenidis, Anupam Prakash **Quantum Recommendation Systems** arXiv:1603.08675 ITCS 2017, p. 49:1–49:21 (2016)

[CS16] Anirban Narayan Chowdhury, Rolando D. Somma **Quantum algorithms for Gibbs sampling and hitting-time estimation** arXiv:1603.02940 Quant. Inf. Comp.

Vol. 17, No. 1, pp. 0041-0064 (2016)

[BG16] Sergey Bravyi, David Gosset **Improved Classical Simulation of Quantum Circuits Dominated by Clifford Gates** Phys. Rev. Lett. 116, 250501 arXiv:1601.07601 (2016)

[DB16] Steven Diamond, Stephen Boyd **CVXPY: A Python-Embedded Modeling Language for Convex Optimization** Journal of Machine Learning Research (2016)

[Raussendorf&15] Robert Raussendorf, Dan Browne, Nicolas Delfosse, Cihan Okay, Juan Bermejo-Vega. **Contextuality and Wigner function negativity in qubit quantum computation** Phys Rev X 5, 021003, arXiv:1511.08506, (2015)

[CKS15] Andrew M. Childs, Robin Kothari, Rolando D. Somma **Quantum algorithm for systems of linear equations with exponentially improved dependence on precision** SIAM Journal on Computing 46, 1920-1950 arXiv:1511.02306 (2015)

[Greenbaum15] Daniel Greenbaum. **Introduction to Quantum Gate Set Tomography** arXiv:1509.02921 (2015)

[Mon15] Ashley Montanaro **Quantum speedup of Monte Carlo methods** Proc. Roy. Soc. Ser. A, vol. 471 no. 2181, 20150301 arXiv:1504.06987 (2015)

[DH15] Hillary Dawkins, Mark Howard. **Qutrit Magic State Distillation Tight in Some Directions** Phys. Rev. Lett. 115, 030501, arXiv:1504.05965 (2015)

[PWB15] Hakop Pashayan, Joel Wallmann, Steven Bartlett **Estimating outcome probabilities of quantum circuits using quasiprobabilities** Phys. Rev. Lett. 115, 070501 arXiv:1503.07525 (2015)

[KLY15] Shelby Kimmel, Guang Hao Low, Theodore J. Yoder **Robust Calibration of a Universal Single-Qubit Gate-Set via Robust Phase Estimation** Phys. Rev. A 92, 062315 arXiv:1502.02677 (2015)

[BCK15] Dominic W. Berry, Andrew M. Childs, Robin Kothari **Hamiltonian simulation with nearly optimal dependence on all parameters** arXiv:1501.01715 Proc. FOCS, pp. 792-809 (2015)

[FG14] Edward Farhi, Jeffrey Goldstone. **A Quantum Approximate Optimization Algorithm Applied to a Bounded Occurrence Constraint Problem** arxiv:1412.6062 *MIT-CTP/4628* (2014)

[Berry&14] Dominic W. Berry, Andrew M. Childs, Richard Cleve, Robin Kothari, Rolando D. Somma **Simulating Hamiltonian dynamics with a truncated Taylor series** arXiv:1412.4687 Phys. Rev. Lett. 114, 090502 (2014)

[YLC&14] 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)

[HWVE14] Mark Howard, Joel Wallman, Victor Veitch, Joseph Emerson. **Contextuality supplies the magic for quantum computation** doi:10.1038/nature13460, arXiv:1401.4174 (2014)

[Pedernales&14] J. S. Pedernales, R. Di Candia, I. L. Egusquiza, J. Casanova, E. Solano **Efficient Quantum Algorithm for Computing n-time Correlation Functions** arXiv:1401.2430 Phys. Rev. Lett. 113, 020505 (2014)

[TaShma13] Amnon Ta-Shma **Inverting well conditioned matrices in quantum logspace** STOC '13, Pages 881–890 (2013)

[JMdW13] Stacey Jeffery, Frederic Magniez, Ronald de Wolf **Optimal parallel quantum query algorithms** arXiv:1309.6116 Algorithmica volume 79, pages 509-529 (2013)

[YLMV13] Elahe Yeganegi, Ad Lagendijk, Allard P. Mosk, Willem L. Vos **Local density of optical states in the band gap of a finite photonic crystal** arXiv:1309.5730 Phys. Rev. B 89, 045123 (2013)

[VMGE13] Victor Veitch, Seyed Ali Hamed Mousavian, Daniel Gottesman, Joseph Emerson. **The Resource Theory of Stabilizer Computation** New J. Phys. 16, 013009, arXiv:1307.7171 (2013)

[Wilt13] Nicholas Wilt **The CUDA Handbook: A comprehensive Guide to GPU programming** Addison-Wesley Professional; 1st edition, June 22 2013

[Bookatz12] Adam D. Bookatz. **QMA-complete problems** arXiv:1212.6312 Quantum Information and Computation, Vol.14 No.5-6 (2012)

[Beals&12] Robert Beals, Stephen Brierley, Oliver Gray, Aram Harrow, Samuel Kutin, Noah Linden, Dan Shepherd, Mark Stather **Efficient Distributed Quantum Computing** Proc. R. Soc. A 2013 469, 20120686 arXiv:1207.2307 (2012)

[CW12] Andrew M. Childs, Nathan Wiebe **Hamiltonian Simulation Using Linear Combinations of Unitary Operations** arXiv:1202.5822 Quantum Information and Computation 12, 901-924 (2012)

[ACB12] Hussain Anwar, Earl Campbell, Dan Browne. **Qutrit Magic State Distillation** New J. Phys. 14, 063006, arXiv:1202.2326 (2012)

[VFGE12] Victor Veitch, Christopher Ferrie, David Gross, Joseph Emerson. **Negative Quasi-Probability as a Resource for Quantum Computation** New J. Phys. 15 039502, arXiv:1201.1257 (2012)

[JLP11] Stephen P. Jordan, Keith S. M. Lee, John Preskill **Quantum Algorithms for Quantum Field Theories** arXiv:1111.3633 Science Vol 336, Issue 6085, pp. 1130-1133 (2011)

[BDGT11] Gilles Brassard, Frederic Dupuis, Sebastien Gambs, Alain Tapp. **An optimal quantum algorithm to approximate the mean and its application for approximating the median of a set of points over an arbitrary distance** arXiv:1106.4267 (2011)

[ORR11] Maris Ozols, Martin Roetteler, Jérémie Roland **Quantum Rejection Sampling** arXiv:1103.2774 IRCS'12 pages 290-308 (2011)

[Holzner&11] Andreas Holzner, Andreas Weichselbaum, Ian P. McCulloch, Ulrich Schollwöck, Jan von Delft **Chebyshev matrix product state approach for spectral functions** arXiv:1101.5895 Phys. Rev. B 83, 195115 (2011)

[YA11] Man-Hong Yung, Alán Aspuru-Guzik **A Quantum-Quantum Metropolis Algorithm** arXiv:1011.1468 PNAS 109, 754-759 (2010)

[Ambainis10] Andris Ambainis **Variable time amplitude amplification and a faster quantum algorithm for solving systems of linear equations** arXiv:1010.4458 STACS'12, 636-647 (2010)

[BFS10] Brielin Brown, Steven T. Flammia, Norbert Schuch **Computational Difficulty of Computing the Density of States** arXiv:1010.3060 Phys. Rev. Lett. 107, 040501 (2010)

[Temme&09] K. Temme, T.J. Osborne, K.G. Vollbrecht, D. Poulin, F. Verstraete **Quantum Metropolis Sampling** arXiv:0911.3635 Nature volume 471, pages 87–90 (2009)

[VendenNest09] M. Van den Nest **Simulating quantum computers with probabilistic methods** Quant. Inf. Comp. 11, 9-10 pp. 784-812 arXiv:0911.1624 (2009)

[Diak09] Ilias Diakonikolas, Parikshit Gopalan, Ragesh Jaiswal, Rocco Servedio, Emanuele Viola **Bounded Independence Fools Halfspaces** arXiv:0902.3757 FOCS '09, Pages 171–180 (2009)

[DiVentra08] Massimilano Di Ventra **Electrical Transport in Nanoscale Systems** Cambridge University Press, New York. (2008)

[HHL08] Aram W. Harrow, Avinatan Hassidim, Seth Lloyd **Quantum algorithm for solving linear systems of equations** Phys. Rev. Lett. 103, 150502 arXiv:0811.3171 (2008)

[Higgins07] B. L. Higgins, D. W. Berry, S. D. Bartlett, H. M. Wiseman, G. J. Pryde **Entanglement-free Heisenberg-limited phase estimation** Nature.450:393-396 arXiv:0709.2996 (2007)

[Isaacson07] Walter Isaacson **Einstein: His Life and Universe** Simon & Shuster ISBN:9780743264730 (2007)

[EY06] Alexandre Eremenko, Peter Yuditskii **Uniform approximation of sgn(x) by polynomials and entire functions** arXiv:0604324 Journal d'Analyse Mathématique, 101 313-324 (2006)

[Gross06] David Gross **Hudson's Theorem for finite-dimensional quantum systems** Journal of Mathematical Physics 47, 122107 arXiv:quant-ph/0602001 (2006)

[MW05] Chris Marriott, John Watrous **Quantum Arthur-Merlin Games** CC, 14(2): 122 - 152 arXiv:cs/0506068 (2005)

[WWAF05] Alexander Weisse, Gerhard Wellein, Andreas Alvermann, Holger Fehske **The Kernel Polynomial Method** arXiv:cond-mat/0504627 Rev. Mod. Phys. 78, 275 (2005)

[Sze04] Mario Szegedy **Quantum speed-up of Markov chain based algorithms** FOCS '04, Pages 32-41 (2004)

[AG04] Scott Aaronson, Daniel Gottesman **Improved Simulation of Stabilizer Circuits** arXiv:quant-ph/0406196 Phys. Rev. A 70, 052328 (2004)

[KKR04] Julia Kempe, Alexei Kitaev, Oded Regev. **The Complexity of the Local Hamiltonian Problem** arXiv:quant-ph/0406180 Proc. 24th FSTTCS, p. 372-383 (2004)

[HMH04] Michael Hartmann, Guenter Mahler, Ortwin Hess **Spectral densities and partition functions of modular quantum systems as derived from a central limit theorem** arXiv:0406100 Journal of Statistical Physics volume 119, pages1139–1151 (2004)

[Jordan04] Stephen P. Jordan **Fast quantum algorithm for numerical gradient estimation** arXiv:quant-ph/0405146 Phys. Rev. Lett. 95, 050501 (2004)

[AKS04] Agrawal, Manindra; Kayal, Neeraj; Saxena, Nitin **PRIMES is in P** Annals of Mathematics Volume 160, pp. 781-793 (2004)

[SOKG03] Rolando Somma, Gerardo Ortiz, Emanuel Knill, and James Gubernatis **Quantum Simulations of Physics Problems** arXiv:quant-ph/0304063 AeroSense '03 (2003)

[ZS03] Karol Zyczkowski, Hans-Juergen Sommers **Hilbert–Schmidt volume of the set of mixed quantum states** J. Phys. A 36, 10115-10130 arXiv:quant-ph/0302197 (2003)

[Klauck02] Hartmut Klauck **Quantum Time-Space Tradeoffs for Sorting** STOC 03, Pages 69–76 arXiv:quant-ph/0211174 (2002)

[Ambainis02] Andris Ambainis **Quantum lower bounds by quantum arguments** STOC '02 pp 750–767 arXiv:quant-ph/0002066 (2002)

[HNS01] Peter Hoyer, Jan Neerbek, Yaoyun Shi **Quantum complexities of ordered searching, sorting, and element distinctness** 28th ICALP, LNCS 2076, pp. 346-357 arXiv:quant-ph/0102078 (2001)

[NC00] Isaac Chuang and Michael Nielsen **Quantum Computation and Quantum Information** Cambridge University Press. ISBN-13: 978-1107002173 (2000)

[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)

[AW99] Daniel S. Abrams, Colin P. Williams **Fast quantum algorithms for numerical integrals and stochastic processes** arXiv:quant-ph/9908083 (1999)

[ABO99] Dorit Aharonov, Michael Ben-Or. **Fault-Tolerant Quantum Computation With Constant Error Rate** arXiv:/quant-ph/9906129 SIAM Journal on Computing. 38 (4): 1207–1282. (1999)

[Gottesman98] Daniel Gottesman **The Heisenberg Representation of Quantum Computers** arXiv:quant-ph/9807006 ICGTMP '99 (1998)

[AKN98] Dorit Aharonov, Alexei Kitaev, Noam Nisan **Quantum Circuits with Mixed States** STOC '97, pages 20-30 arXiv:quant-ph/9806029 (1998)

[NW98] Ashwin Nayak, Felix Wu **The quantum query complexity of approximating the median and related statistics** arXiv:quant-ph/9804066 STOC '99 pp 384-393 (1998)

[Beals&98] Robert Beals, Harry Buhrman, Richard Cleve, Michele Mosca, Ronald de Wolf arXiv:quant-ph/9802049 FOCS '98 pp 778–797 (1998)

[Grover97] Lov Grover **A framework for fast quantum mechanical algorithms** arXiv:quant-ph/9711043 STOC '98 Pages 53–62 (1997)

[BBBV97] Charles H. Bennett, Ethan Bernstein, Gilles Brassard, Umesh Vazirani **Strengths and Weaknesses of Quantum Computing** arXiv:quant-ph/9701001 SIAM Journal on Computing 26(5):1510-1523 (1997)

[Lloyd96] Seth Lloyd **Universal Quantum Simulators** Science 273, no. 5278, 1073–1078 (1996)

[Grover96] Lov Grover **A fast quantum mechanical algorithm for database search** arXiv:quant-ph/9605043 STOC '96 (1996)

[Kit95] A. Yu. Kitaev **Quantum measurements and the Abelian Stabilizer Problem** arXiv:quant-ph/9511026 (1995)

[Shor95] Peter W. Shor **Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer** SIAM J.Sci.Statist.Comput. 26, 1484 arXiv:quant-ph/9508027 (1995)

[HW91] R. Heiman; A. Wigderson **Randomized vs. deterministic decision tree complexity for read-once Boolean functions** Proceedings of the Sixth Annual Structure in Complexity Theory Conference (1991)

[Sears84] V. F. Sears. **Scaling and final-state interactions in deep-inelastic neutron scattering** Phys. Rev. B **30**, 44 (1984)

[West75] G. B. West **Electron scattering from atoms, nuclei and nucleons** Physics Reports **18**, 263 (1975)

[Rivlin69] Theodore J. Rivlin **An Introduction to the Approximation of Functions** Dover Publications, Inc. New York. ISBN-13:978-0486640693 (1969)

[Bell64] J. S. Bell **On the Einstein Podolsky Rosen paradox** Physics Physique Fizika 1, 195 (1964)

[Hoeffding63] Wassily Hoeffding **Probability Inequalities for Sums of Bounded Random Variables** Journal of the American Statistical Association, Vol. 58, No. 301, pp. 13-30 (1963)

[Dolph46] C. L. Dolph. **A Current Distribution for Broadside Arrays Which Optimizes the Relationship Between Beam Width and Side-Lobe Level** Proceedings of the IRE, 34(6):335–348, 1946.

[EPR35] A. Einstein, B. Podolsky, and N. Rosen **Can Quantum-Mechanical Description of Physical Reality Be Considered Complete?** Phys. Rev. 47, 777 (1935)

[Born26] Max Born **Zur Quantenmechanik der Stoßvorgänge** Princeton University Press pp. 863–867 (1926)