Senior Thesis in Physics

Random Quantum Circuits, Quantum t-Designs, and Emergent Irreversibility

Author:
Jacob Hauser

Advisors:
Professor Alioscia Hamma
Professor Eli Quetin

Submitted to Pomona College in Partial Fulfillment
of the Degree of Bachelor of Arts

April 30, 2020

Contents

Abstract

Although irreversibility is a well-understood classical phenomenon, it is a much more elusive notion in quantum mechanics. Random quantum circuits provide a very general way to model the dynamics of quantum many-body systems. Two groups of circuits are of particular interest: random Clifford circuits and random universal circuits. Previous work has suggested that the transition from reversible to irreversible quantum dynamics connects to the transition between these two classes of circuits. Clifford circuits are known to be 3-designs – that is, they reproduce the Haar measure up to the third moment – but, unlike random universal circuits, they are not a 4-design. In this thesis, we attempt to use computational methods to study the transition from 3-design to 4-design as non-Clifford gates are added to Clifford circuits. To accomplish this, we build a robust and efficient engine for generating random quantum circuits of various architectures. We find that our original approach cannot yield useful insights; nevertheless, we propose and begin to explore a more promising twist on our original method. Additionally, our random quantum circuit engine has broad utility. We demonstrate this by using it to visualize Clifford and random universal circuits in a way that may lend itself to the application of machine learning at this boundary in the future.

Acknowledgements

This thesis project would not have been as meaningful or educational – or even possible – without the help and support of many individuals. I am incredibly thankful that Professor Hamma agreed to supervise this project from afar. I have learned so much from him this year and deeply appreciate his guidance and support throughout the year. Likewise, I want to thank Professor Quetin for agreeing to supervise this project from the Pomona side. It's been very reassuring to have his unwavering support through the ups and downs of research. Asya Shklyar's help was indispensable. Her kindness and knowledge are remarkable, and much of this project rests on computational improvements she advised on.

I also want to thank Professor Moore for teaching me physics, Professor Whitaker for teaching me to be a physicist, and Natalie Hughes for making the Pomona physics department so special. Professor Stecklein's wise advice and support throughout the year were also invaluable. This is an odd time and place to be completing a thesis, but, even far away from Pomona, I feel supported by this physics department and thankful that I've been able to be part of it for the past four years. The physics lounge, the students, and the department's snack budget were all integral to this thesis.

Finally, I want to my parents for all of their love and support from afar – and then, unexpectedly, from quite near after all.

Chapter 1

Introduction

The classical world is manifestly irreversible: a piece of paper can catch on fire and crumble into ash, but a pile of ash cannot reassemble into a burning piece of paper.

The same phenomenon can be formulated in more precise terms by considering an insulated container partitioned into two parts, as in Figure 1.1. Initially, the left side contains a gas and the right side contains only vacuum. Then the partition is removed so that the gas expands into the entire container.

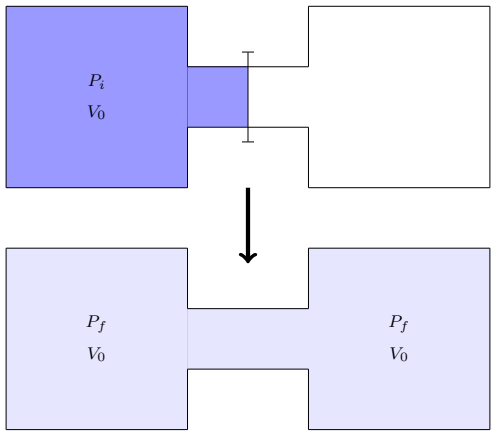


Figure 1.1: In the upper image, a partition separates two regions of an insulated container of volume $2V_0$ so that one (on the left, with volume V_0) has gas at pressure P_i and the other (on the right) is empty. In the lower image, the partition has been removed and the gas now fills the entire container with pressure $P_f < P_i$. This process, called Joule expansion, is irreversible.

This expansion occurs spontaneously, but the reverse process does not [1, p. 144–146]. This is because the system's entropy increases during the expansion,[1] and, according to the second law of thermodynamics, no process can reduce the entropy of a closed system.

Such irreversibility is a natural consequence of statistical mechanics, where probabilities are assigned to certain states and a system naturally tends towards states with maximal probability. However, it is not an obvious consequence of quantum mechanics. In quantum mechanics, the evolution of states is driven by unitary operators, in order to preserve probability [31, p. 112]. To demonstrate this, suppose that some normalized state $|\psi\rangle$ is evolved by some operator $\hat{U}$:

$$|\psi\rangle \to |\psi'\rangle = \hat{U}\,|\psi\rangle$$

Then,

$$\langle\psi'|\psi'\rangle = \langle\psi|\,\hat{U}^\dagger\hat{U}\,|\psi\rangle$$

This must be equal to $\langle\psi|\psi\rangle$ in order for the state to still be normalized and thus for probability to be conserved. This is the case if and only if $\hat{U}^\dagger\hat{U} = \mathbb{1}$, the definition of a unitary matrix. As a corollary, every unitary matrix has an inverse – its Hermitian conjugate.

With this in mind, it's clear that quantum mechanics doesn't facially share the irreversible characteristics of the classical world; every probability-preserving evolution is reversible. Where, then, does irreversibility come from?

In classical physics, explanations of irreversibility rely on statistical assumptions. This research aims to explore how irreversibility emerges from quantum mechanics. Specifically, we study the boundary between the Clifford group, a set of particularly well-behaved unitary operators, and the full set of unitary operators. Related research has studied three ideas at this boundary: learnability, entanglement spectrum statistics (ESS), and quantum t-designs. Here, we primarily study quantum t-designs, seeking to connect their behaviour with learnability and ESS. This research is intriguing for two reasons: all three ideas are naturally connected to irreversibility through notions of complexity, and the boundary between the Clifford group and the full set of unitary operators is relevant for realizing practical quantum computing.

This chapter provides a foundation for understanding learnability, ESS, and quantum t-designs and connecting them to the broader questions of emergent irreversibility and practical quantum computing. First, we introduce basic concepts in classical computing and extend them to quantum computing. With this foundation, we also introduce the Clifford group and notions of universal quantum computing. Next, we briefly discuss the three characteristics we

[1] $\Delta S = Nk_b \log\left(\frac{V_f}{V_i}\right) = Nk_b \log 2$ is the entropy change associated with Joule expansion, where $V_i = V_0$ is the volume of the left region of the container and $V_f = 2V_0$ is the volume of the entire container.

study, contextualizing them in terms of irreversibility and the current literature. Finally, we connect these ideas to the goal of realizing practical quantum computing, and provide a roadmap for the remainder of this thesis.

1.1 Background

1.1.1 The Past and Future of Quantum Computing

Computers are ubiquitous in the modern world, but the history of computing stretches far earlier than the founding of today's Silicon Valley giants. The notion of an algorithm – a "well-defined computational procedure that takes some value, or set of values, as input and produces some value, or set of values, as output" – was explored even before computers existed [7]. Over time, engineering advances allowed computers to realize their theoretical potential.

Similarly, the theoretical framework for quantum computing has preceded its practical realization. The field of quantum information began to gather steam in the 1970s, and the idea of quantum computing was popularized by Richard Feynman in 1981 [19]. As will be discussed in Section 1.1.6, quantum systems are generally too complex to simulate on a classical computer. With this in mind, Feynman explored the possibility of using a quantum system itself for computations in order to efficiently model quantum systems [11].

Since then, a wide range of powerful uses for quantum computers have been studied. Although they will not fill the same role as classical computers – "a quantum computer will not necessarily be faster, bigger, or smaller than an ordinary computer" – they will allow exponentially faster algorithms for certain purposes, with revolutionary implications for cryptography, convex optimization, and quantum simulation [17].

1.1.2 From Classical Computing to Quantum Computing

The electronics underlying standard computers enforce a natural binary: low voltages correspond to 0 and high voltages correspond to 1. Therefore, information in computers is built from *bits*, short for binary digits. Clever circuitry allows computers to store binary data (in memory) and perform arithmetic on them (in processors). This brief description contains kernels of several key ideas:

1. Our model for computation is *discrete*. A reasonable computer has finite storage, which ensures that only a finite set of values are possible. Quan-

tum mechanical systems are also discrete (hence "quantum"), suggesting a possible extension to quantum systems.

2. Information is stored in *states* composed of bits and processed through *circuits* which operate on these bits. The same constructions will be integral to quantum computing.

3. *Space* and *time* are important, limited resources. Storing information in states requires space and performing computations requires time. Effectively navigating the use of these resources is important in classical computing, and also in quantum computing.

1.1.3 Bit to Qubit

The quantum mechanical analog to a bit is a qubit: a superposition of some orthonormal states $|0\rangle$ and $|1\rangle$. The space of possible states for a qubit is larger than that for a bit. While $|0\rangle$ and $|1\rangle$ are possible states for the qubit, so is any state

$$a\,|0\rangle + b\,|1\rangle \text{ where } a, b \in \mathbb{C} \text{ and } |a|^2 + |b|^2 = 1 \tag{1.1}$$

and where $\mathbb{C}$ is the set of complex numbers. The space of these states is geometrically equivalent to a sphere, called the Bloch sphere, depicted in Figure 1.2. This representation provides useful intuition for the differences between the space associated with a bit and the space associated with a qubit. A bit may attain only the values on the poles of the Bloch sphere whereas a qubit may attain any value on the surface of the sphere.

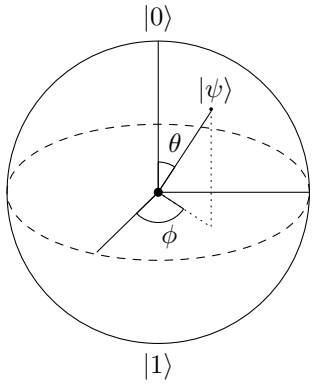


Figure 1.2: The Bloch sphere illustrates the relationship between the possible states of one qubit and the surface of a sphere. To translate between the variables used in Equation 1.1 and those here, let $a = \cos\frac{\theta}{2}$ and $b = e^{i\phi}\sin\frac{\theta}{2}$.

Nevertheless, when observed, the qubit will collapse into either $|0\rangle$ or $|1\rangle$. Thus, although a continuous set of values are possible for a qubit, its observable values are discrete.

Qubits are simplest when viewed in this theoretical framework, but they have a solid foundation in reality. Canonical examples of qubits – which are, in general, two-level quantum systems – include the spin of an electron and the polarization of a photon [10].

1.1.4 Composite Systems

It is often necessary for classical computers to work with numbers other than 0 or 1. Computers can represent such numbers with multiple bits. For example, using two bits, the set of possible states is $\{0,1\} \times \{0,1\}$, where:

$$\begin{aligned} 0 = 00 &:= (0,0) \\ 1 = 01 &:= (0,1) \\ 2 = 10 &:= (1,0) \\ 3 = 11 &:= (1,1) \end{aligned}$$

In general, the set of possible states for a system of N bits is

$$\underbrace{\{0,1\} \times \{0,1\} \times \cdots \times \{0,1\}}_{N \text{ times}} = \{0,1\}^N$$

which has 2^N elements.

Similarly, we can build new states out of multiple qubits. If a single qubit can attain values in a two-dimensional space $\mathcal{H}$, then the set of possible states for two qubits is $\mathcal{H} \otimes \mathcal{H}$, where:

$$\begin{aligned} |0\rangle = |00\rangle &:= |0\rangle \otimes |0\rangle \\ |1\rangle = |01\rangle &:= |0\rangle \otimes |1\rangle \\ |2\rangle = |10\rangle &:= |1\rangle \otimes |0\rangle \\ |3\rangle = |11\rangle &:= |1\rangle \otimes |1\rangle \end{aligned}$$

forms a basis for $\mathcal{H} \otimes \mathcal{H}$. In general, the set of possible states for a system of N qubits is

$$\underbrace{\mathcal{H} \otimes \mathcal{H} \otimes \cdots \otimes \mathcal{H}}_{N \text{ times}} = \mathcal{H}^{\otimes N}$$

which is 2^N-dimensional.

1.1.5 Logic Gates and Unitary Gates

In classical computing, bits are manipulated via gates which operate on one or more bits. An example of a gate which acts on one bit[2] is the `NOT` gate, which is defined by:

$$0 \mapsto 1$$
$$1 \mapsto 0$$

It is a general property of quantum mechanics that operators act linearly on states [19, p. 18]. That is, an operator $\hat{O}$ must satisfy

$$\hat{O}\left(a\left|0\right\rangle + b\left|1\right\rangle\right) = a\,\hat{O}\left|0\right\rangle + b\,\hat{O}\left|1\right\rangle.$$

Thus, we can construct an analogous `NOT` gate for quantum computing by defining the action of the operator on $|0\rangle$ and $|1\rangle$:

$$|0\rangle \mapsto |1\rangle$$
$$|1\rangle \mapsto |0\rangle$$

Using this basis, there is a natural matrix representation for these states and this operator where:

$$|0\rangle \to \begin{pmatrix}1\\0\end{pmatrix}, \quad |1\rangle \to \begin{pmatrix}0\\1\end{pmatrix}, \quad \text{and} \quad \texttt{NOT} \to \begin{pmatrix}0 & 1\\1 & 0\end{pmatrix}.$$

In this representation, it is clear that `NOT` acts linearly (every matrix corresponds to a linear transformation) and that it performs as promised on the basis states $|0\rangle$ and $|1\rangle$.

Classical computing also allows gates which operate on multiple bits. Consider, for example, the `NAND` gate, ("not-and"), defined by:

$$(0,0) \mapsto 1, \quad (0,1) \mapsto 1$$
$$(1,0) \mapsto 1, \quad (1,1) \mapsto 0$$

We seek to define analogous gates for quantum computing. However, since quantum gates evolve states, we are limited to the set of unitary operators described at the beginning of this chapter. Thus, the matrix representations of our operators must be square, so each maps a set of N states to N new states (unlike `NAND` which maps two states to only one state).

In general, any quantum gate can be fully described by its action on basis states, and it turns out that all finite unitary operators are legitimate quantum circuits, where a circuit is the operator that results from the composition of various quantum gates [19, p. 20]. This allows a more general perspective than

[2] In fact, this is the only non-trivial gate acting on one bit.

the limited study of unitary evolution due to a particular Hamiltonian $\hat{H}$ where $\hat{U}(t) = e^{-i\hat{H}t/\hbar}$.

This is why quantum computing is an appropriate lens through which to study the question of emergent irreversibility. Qubits are simple quantum systems to work with and can model arbitrarily complicated systems for sufficiently large N, and the quantum gates which evolve qubits can represent any probability-preserving evolution.

1.1.6 Complexity and Entanglement

In order to fully describe a system of N bits, N pieces of information are required. However, the same is not true for systems of N qubits. Such systems correspond to 2^N-dimensional vector spaces, as discussed in Section 1.1.4. Thus, 2^N pieces of information are required to describe an arbitrary state.

This is a fundamental difference between the *complexity* of classical systems and quantum systems. In both classical and quantum computing, complexity describes how space and time requirements for computations scale with the size of inputs. The space required to store states and time required to evolve them scales exponentially with N for quantum systems, even though it scales linearly with N for classical systems.

This difference is a result of entanglement – the use of shared sections of the tensor product structure of our N-qubit system. For example, if a state in $\mathcal{H}^{\otimes N}$ can be described as the tensor product of pure states, then it can be completely described by $2N$ pieces of information – 2 for each qubit. Only for states which are not the tensor product of pure states do quantum systems become more complex than classical systems.

1.1.7 The Clifford Group

In classical computing, the set of gates which evolve an N-bit initial state into an N-bit final state is the set of maps from $\{0,1\}^N$ to itself. This set is very large – in fact, there are $\left(2^N\right)^{\left(2^N\right)}$ such functions – but it is clearly finite.

Noting that combining two logic gates results in a new logic gate, it is possible that this large set of maps can be *generated* by a smaller set. That is, perhaps there is a smaller set of gates which can be combined in various ways to produce any circuit. Such a set of gates is called a *universal set*. It turns out that `NAND` alone is a universal set for classical computing: any circuit you can imagine can be constructed from purely `NAND` gates operating on various pairs of bits [27].

It's not immediately clear that a similar universal set exists for quantum gates. The set of possible gates evolving an N-qubit state is $U(2^N)$, the set of $2^N \times 2^N$ unitary operators. This set is 2^{2N}-dimensional and has an infinite number of elements.

In our search for a universal set of operations, consider the following gates:

- `H` (Hadamard) gate, which operates on one qubit, mapping:

$$|0\rangle \mapsto \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$$
$$|1\rangle \mapsto \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$$

- Phase gates $\mathtt{P}_\delta$, which operate on one qubit, mapping:

$$|0\rangle \mapsto |0\rangle$$
$$|1\rangle \mapsto e^{i\delta}|1\rangle$$

  In particular, we label the phase gate where $\delta = \frac{\pi}{4}$ as `T` and that where $\delta = \frac{\pi}{2}$ as `S`.

- `CNOT` (controlled-NOT) gate, which operates on two qubits, mapping:

$$|00\rangle \mapsto |00\rangle$$
$$|01\rangle \mapsto |01\rangle$$
$$|10\rangle \mapsto |11\rangle$$
$$|11\rangle \mapsto |10\rangle$$

Consider the set $G_C = \{\mathtt{H}, \mathtt{S}, \mathtt{CNOT}\}$. This set of gates generates a set of operators called *the Clifford group* [13]. Each element of the Clifford group is called a Clifford operator[3] and can be generated by applying elements of G_C to various qubits. Note, however, that although the number of gates required to generate the Clifford group is small, generating any individual element of the Clifford group may require a large number of these gates [4].

[3]So far, we have used the terms "operator", "gate", and "circuit" in similar ways, and we continue to do so throughout the rest of this thesis. Although the boundaries between the terms are blurry, we mean to distinguish them in the follow ways:

- **Operator** is the most general term, encompassing any linear map applied to quantum mechanical states, including both gates and circuits.
- **Gate** is mostly used to refer to important, ideally physically realizable operators in quantum computing; e.g., the `CNOT` or `T` gates.
- **Circuit** is mostly used to refer to operators expressed as products of gates; a circuit is sometimes viewed as a list of these gates and sometimes as the single operator that results from composing them.

However, our usage doesn't perfectly conform to that described above.

Interestingly, quantum circuits composed only of Clifford gates can be simulated efficiently by a classical computer [3]. It follows that the Clifford group cannot be the entire group of unitary operators $U(2^N)$ because evolution under arbitrary circuits cannot be efficiently simulated using classical computers. Thus, it turns out that the Clifford group is a particularly well-behaved subset of $U(2^N)$.

Now, consider the set $G_U = \{\texttt{H}, \texttt{T}, \texttt{CNOT}\}$. This set is sufficient for universal quantum computing. By composing these gates, applied to various qubits, it is possible to approximate any unitary operator arbitrarily well.

1.2 Three Important Ideas

1.2.1 Learnability

Returning to our original discussion of emergent irreversibility, suppose that some normalized state $|\psi\rangle$ is evolved by some operator $\hat{U}$:

$$|\psi\rangle \to |\psi'\rangle = \hat{U}\,|\psi\rangle$$

In particular, suppose that $\hat{U}$ is the composition of gates chosen from a particular finite set G – perhaps $G = G_C$ or $G = G_U$. This construction is realistic: although all gates in the set generated by G are possible operators, a practical $\hat{U}$ (realized in a physical system) must be built from a finite number of gates.

Let $M \in \mathbb{N}$ be the number of gates composing $\hat{U}$. If the individual gates composing $\hat{U} = \hat{U}_M \dots \hat{U}_1$ are known, then it is easy find $\hat{U}^\dagger = \hat{U}_1^\dagger \dots \hat{U}_M^\dagger$. However, if the various $\hat{U}_i$ are randomly selected and not recorded, then it is generally difficult to find $\hat{U}^\dagger$. This mimics the randomness inherent in statistical mechanics.

With this in mind, an interesting question arises: when is it possible to efficiently learn how to reverse the evolution due to $\hat{U}$? It is certainly possible to do this efficiently when the evolved state is completely unentangled. Then, the state can be described by only $2N$ pieces of information and an operator which evolves this state to the original state can be found efficiently.

Therefore, an equivalent question is whether it is possible to efficiently disentangle a state. If the circuits generated by a set of gates generally evolve a state which can be efficiently disentangled, that set of gates is *learnable*. If not, the set of gates is *unlearnable*.

In [4], the learnability of Clifford gates was compared to that of $U(2^N)$. In particular, states were evolved under random circuits composed of either Clifford gates or universal gates – hereafter called Clifford circuits and random universal

circuits[4] respectively – until their entanglement was maximal. Then, a classical algorithm was applied to disentangle the state. It was shown that Clifford circuits are learnable, but random universal circuits are not.

1.2.2 Entanglement Spectrum Statistics (ESS)

The reliance of learnability on entanglement suggests that the structure of entanglement in a state may contain important information. One way to explore this is through ESS.

A detailed explanation of ESS requires material beyond this chapter. However, there are some useful things we can say about ESS at this time. First, two states which appear maximally entangled can have very different ESS [4]. This indicates a deeper layer of complexity than mere entanglement: there are different ways to be entangled, and we can characterize these ways as more or less complex. Second, a very intriguing correspondence was demonstrated in [4] between learnability and ESS. In particular, one ESS distribution was associated with learnable circuits and another with unlearnable circuits.

As a result, Clifford circuits and random universal circuits are separated by both learnability and ESS. A natural subsequent question is where this transition takes place. If `T` is added to G_C, a set which generates the Clifford group, then the resulting set generates $U(2^N)$. Therefore, one can study the behaviour of learnability and ESS as the density of `T` gates is increased in a circuit primarily composed of Clifford gates. As the density of `T` gates is increased, we expect the circuit to behave more like one composed of gates from $U(2^N)$.

This question was explored in [34]. There, it was found that in the large N limit (also called the thermodynamic limit), a single `T` gate embedded in a circuit composed of Clifford gates was sufficient to transition from Clifford-like ESS to unitary-like ESS. The thermodynamic limit is particularly interesting because we seek to understand many-body (large N) quantum systems, where we expect irreversible tendencies will begin to arise (because they certainly don't for very simple systems).

1.2.3 Quantum t-Designs

The final idea, which is particularly important for our work, is that of quantum t-designs, which help quantify how evenly distributed a set of quantum states or operators is. In particular, certain well-distributed sets can be identified as t-designs where larger t indicates stronger agreement with a uniform distribution.

[4]These circuits are intended to approximate the Haar distribution, which is discussed briefly in Section 2.3.3.

We elaborate rather extensively on quantum t-designs in Section 2.3, but hopefully some essence of complexity is already apparent: a circuit which distributes states less evenly has more discrete behaviour and is less complex, while the opposite is true for a circuit which distributes states more evenly. The former case seems potentially easier to reverse (and perhaps, thus, to learn) whereas the latter case seems potentially irreversible.

It is known that the Clifford group is only a 3-design whereas random universal circuits are a 4-design (and higher) [36]. However, when we began this work it was not known at what density of `T` gates the transition from 3-design to 4-design occurs.[5] It was conjectured in [34] that the transition point aligns with that of learnability and ESS.

1.2.4 Our Work

Our original goal was to characterize this t-design transition between Clifford gates and $U(2^N)$ using computational methods. This was unsuccessful in two ways. First, we determined that the problem was not tractable using our original method. Second, early in 2020, another research group succeeded in analytically characterizing this transition in [16], as we discuss in Section 5.1.

However, the work we have conducted is still useful for several reasons. Although [16] determined how the number of `T` gates required for mostly-Clifford circuits of a particular architecture to comprise a 4-design varies with t and N, it did not find any concrete values for this number of gates. Insofar as we are interested in practical applications of mostly-Clifford circuits, determining these values is useful. Furthermore, while the analytical methods employed in [16] allow for a deep and convincing result, a robust computational method for analyzing t-designs would be useful in complementary ways, allowing the analysis of other circuit architectures and more complicated systems.

Additionally, although our original method was unsuccessful, we have proposed a new approach which could yet yield useful results. Regardless, over the course of this project we built a robust engine for generating and working with random quantum circuits. This engine has broad utility; as an example, we use it to visualize quantum circuits in a way that seems to produce different behaviour for Clifford and random universal circuits.

[5] Since then, an independent research group has analytically characterized this transition. Nevertheless, there is still interesting work to be done in this area.

1.2.5 Implications for Practical Quantum Computing

Despite significant experimental advances over the past decade, quantum computers are not yet sufficiently powerful to provide a meaningful speedup compared to classical computers. We are particularly interested in two major obstacles: limiting *decoherence* and achieving *universal quantum computing*. Decoherence is the process by which errors are introduced to quantum computations through interactions with the surrounding environment [17]. Although methods have developed to correct for such errors, thereby producing *fault-tolerant quantum computing*, such methods make it more difficult to implement `T` gates than to implement Clifford gates [21]. Furthermore although `T` gates are necessary for universal quantum computing (the ability to carry out any arbitrary computation), they are difficult to implement regardless of decoherence [10].

A better understanding of the transition between the Clifford group and $U(2^N)$ may allow the development of strategic circuit designs relying primarily on Clifford gates, with judicious use of `T` gates.

1.3 A Roadmap

Having introduced a broad array of important ideas in the chapter, we continue by addressing several in more detail in Chapter 2. In particular, we build up the theory underlying quantum t-designs and present a framework for analyzing them computationally. We also discuss quantum states, operators, and their representations, basic group theory in the context of quantum circuits, and computational complexity. In Chapter 3, we describe our methods for generating and analyzing quantum circuits, and our approaches to optimizing these methods so that we can probe larger systems of qubits.

In Chapter 4, we present and discuss our findings on three topics. First, we explain the intractability of our original method for determining the t-design transition and propose an adaptation of this method that may allow success in the future. Second, we study the performance of three different implementations of our code, discussing the progress we have made in optimizing our random circuit methods. Third, we apply these random circuit methods to a related research question, demonstrating the broader applicability of our work.

Finally, in Chapter 5, we briefly discuss related work by a different research group that recently used analytical tools to answer the key question we were exploring in this thesis. Then we summarize the progress we made, discuss future steps, and present some closing thoughts.

Chapter 2

Theory

In this chapter, we lay useful groundwork for the rest of this thesis. We are primarily concerned with three topics. First, in Section 2.1, we put in more rigorous terms how tensor product spaces work, in particular when states and operators are viewed as vectors and matrices, and when they are viewed as tensors. Second, in Section 2.2, we introduce a couple of important ideas in group theory and discuss the Clifford and unitary groups. Third, in Section 2.3 we build up the notion of t-designs, providing some theoretical background and connecting them to properties we can analyze computationally. Lastly, we close in Section 2.4 by briefly describing big O notation and computational complexity.

2.1 States and Operators

In quantum mechanics, states can be represented as vectors in Hilbert spaces.[1] As discussed briefly in Section 1.1.4, composite states – including multiple qubit systems – are elements of a tensor product space. To illustrate this, suppose that $\{|d_1\rangle, |d_2\rangle, \ldots, |d_m\rangle\}$ is an orthonormal basis for an m-dimensional Hilbert space $\mathcal{H}_A$ and $\{|e_1\rangle, |e_2\rangle, \ldots, |e_n\rangle\}$ is an orthonormal basis for an n-dimensional Hilbert space $\mathcal{H}_B$, so that

$$|\psi_A\rangle = \sum_{i=1}^{m} a_i |d_i\rangle \tag{2.1}$$

[1] Hilbert spaces are vector spaces with extra structure. Among other things, Hilbert spaces introduce an inner product.

is an arbitrary state in $\mathcal{H}_A$ and

$$|\psi_B\rangle = \sum_{j=1}^{n} b_j |e_j\rangle \tag{2.2}$$

is an arbitrary state in $\mathcal{H}_B$. Then, when the two systems are considered together, $\mathcal{B} = \{|d_i\rangle \otimes |e_j\rangle \mid 1 \leq i \leq m, 1 \leq j \leq n\}$ is a basis for $\mathcal{H} = \mathcal{H}_A \otimes \mathcal{H}_B$ and

$$|\psi\rangle = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij}(|d_i\rangle \otimes |e_j\rangle)$$

is an arbitrary state in $\mathcal{H}_A \otimes \mathcal{H}_B$. Note from the definition of $\mathcal{B}$ above that $\mathcal{H}_A \otimes \mathcal{H}_B$ has dimension mn. Also, note that just as we built up this composite system $\mathcal{H}$ out of $\mathcal{H}_A$ and $\mathcal{H}_B$, we could also take some system $\mathcal{H}$ and partition it into two regions, as we do in Section 4.3.

If $|\psi\rangle \in \mathcal{H} = \mathcal{H}_A \otimes \mathcal{H}_B$, it's not guaranteed that we can write $|\psi\rangle = |\psi_A\rangle \otimes |\psi_B\rangle$ for some $|\psi_A\rangle \in \mathcal{H}_A$ and $|\psi_B\rangle \in \mathcal{H}_B$. In fact, this cannot be possible in all cases: if all states could be decomposed in this way, then $\mathcal{H}$ would have dimension $m+n$, but we already argued that it has dimension mn. Thus, such "inseparable" states must exist when $mn > m + n$. We call these states *entangled*.

Operators work in a similar way. In general, operators are linear maps on Hilbert spaces and if $\hat{A}$ and $\hat{B}$ operate on states in $\mathcal{H}_A$ and $\mathcal{H}_B$ respectively, then $\hat{A} \otimes \hat{B}$ is an operator on states in $\mathcal{H} = \mathcal{H}_A \otimes \mathcal{H}_B$. However, just like entangled states, an operator on states in $\mathcal{H} = \mathcal{H}_A \otimes \mathcal{H}_B$ cannot generically be written as a tensor product of operators on states in $\mathcal{H}_A$ and $\mathcal{H}_B$ respectively.

When states and operators are viewed as vectors and matrices respectively, the *Kronecker product* provides a way to explicitly determine the tensor product of states or operators. If A is an $m \times n$ matrix and B is a $p \times q$ matrix, then $A \otimes B$ is an $mp \times nq$ matrix that can be written in block form as follows:

$$A \otimes B := [a_{ij}B] = \begin{bmatrix} a_{11}B & \dots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \dots & a_{mn}B \end{bmatrix}$$

The rule for multiplying vectors follows from the general Kronecker product for matrices, with $n = q = 1$.

Index notation also offers a compelling way to view states and operators, so we offer a brief introduction here. A familiar reader may skip the remainder of this section and an unfamiliar one may wish to explore [18, ch. 4-6] for a talented presentation of this information. While this notation is not heavily used in this thesis, it indispensable in several instances.

If we take the liberty of shifting the subscript on a_i in Equation 2.1 to become a superscript (and, to be clear, not an exponent), then we have

$$|\psi_A\rangle = \sum_{i=1}^{m} a^i |d_i\rangle$$

Next, we can adopt the *Einstein summation convention* and omit the "$\sum_{i=1}^{m}$", understanding that it's implied by the paired superscript on a^i and subscript on $|d_i\rangle$. Then we have,

$$|\psi_A\rangle = a^i |d_i\rangle$$

Finally, we note that the coefficients a^i contain all the information necessary to describe $|\psi_A\rangle$ in a particular basis. Thus, we can use a^i to represent the state $|\psi_A\rangle$. Similarly, we can use b^i to represent the state $|\psi_B\rangle$. Then,

$$\begin{aligned}
|\psi_A\rangle \otimes |\psi_B\rangle &= \left(\sum_{i=1}^{m} a^i |d_i\rangle\right) \otimes \left(\sum_{j=1}^{n} b^j |e_j\rangle\right) \\
&= \sum_{i=1}^{m}\sum_{j=1}^{n} a^i b^j |d_i\rangle \otimes |e_j\rangle \\
&= a^i b^j |d_i\rangle \otimes |e_j\rangle
\end{aligned}$$

Since $\{|d_i\rangle \otimes |e_j\rangle\}$ is a basis for $\mathcal{H}_A \otimes \mathcal{H}_B$, we can now use $a^i b^j$ to represent $|\psi_A\rangle \otimes |\psi_B\rangle$ in this space.

Similarly, we can write generic operators as

$$\hat{A} = \sum_{i=1}^{m}\sum_{j=1}^{m} A^i{}_j |d_i\rangle \langle d^j|$$

and, omitting the explicit sums and basis vectors, represent $\hat{A}$ as $A^i{}_j$.[2] Then, when $\hat{A}$ operates on $|\psi_A\rangle$:

$$\begin{aligned}
\hat{A}|\psi_A\rangle &= \left(\sum_{i=1}^{m}\sum_{j=1}^{m} A^i{}_j |d_i\rangle \langle d_j|\right)\left(\sum_{k=1}^{m} a^k |d_k\rangle\right) \\
&= \sum_{i=1}^{m}\sum_{j=1}^{m}\sum_{k=1}^{m} A^i{}_j a^k |d_i\rangle \langle d_j|d_k\rangle \\
&= \sum_{i=1}^{m}\sum_{j=1}^{m} A^i{}_j a^j |d_i\rangle
\end{aligned}$$

[2] In general, our convention is to leave indices on kets as subscripts but to move indices on bras to be superscripts instead. Then, maintaining the Einstein summation convention, the opposite is true for the matching indices on coefficients.

where the last step follows since $\langle d_j|d_k\rangle = 0$ when $j \neq k$ and $\langle d_j|d_k\rangle = 1$ when $j = k$ for orthonormal basis vectors.

Then, dropping the explicit sums and the basis vectors, we can represent $\hat{A}\,|\psi_A\rangle$ as $A^i{}_j a^j$. Again, the subscript/superscript pair of js stands in for a sum over this index, which is called a tensor contraction.

Index notation is a particularly good way to depict the action of local operators – operators that only act on one or two qubits. Let $\mathcal{H} \cong \mathbb{C}^2$ be the Hilbert space corresponding to a single qubit. Then, in a five-qubit system, $|\psi\rangle \in \mathcal{H}^{\otimes 5}$. Let $\hat{U}$ be an operator that acts on the second and fifth qubits in the system. Then, following the above examples, we can represent $|\psi\rangle$ as $\psi^{\nu_1\nu_2\nu_3\nu_4\nu_5}$ where the ν_i are indices like i, j, and k above; we can also represent $\hat{U}$ as $U^{\mu_2}{}_{\nu_2}{}^{\mu_5}{}_{\nu_5}$. Then, just as the operator and state were combined with a tensor contraction above, we can represent $\hat{U}\,|\psi\rangle$ as $U^{\mu_2}{}_{\nu_2}{}^{\mu_5}{}_{\nu_5}\psi^{\nu_1\nu_2\nu_3\nu_4\nu_5}$ where the index notation makes it clear exactly which parts of the system are being acted on: there is an implicit sum over ν_2 and over ν_5.

2.2 Circuits and Group Theory

Up to this point, we've referred to the Clifford group and the unitary group without further explanation. In this section, we define groups in general, the Clifford and unitary groups in particular, and the notion of group orbits. Further elementary group theory can be found in [26], among other places.

Definition 2.2.1. *A tuple $(G, \times)$, where G is a non-empty set and $\times$ is a binary operation on G, is a* group *if the following hold:*

(a) $(a \times b) \times c = a \times (b \times c)$ for all $a, b, c \in G$.
(b) There exists $e \in G$ such that $e \times a = a \times e = a$ for all $a \in G$.
(c) For all $a, b \in G$, $a \times b \in G$.
(d) For all $a \in G$ there exists $b \in G$ such that $a \times b = b \times a = e$.

In other words, a group is a set with an associative binary operation and an identity element, where the set is closed under the operation and under taking inverses.

Both the Clifford group and the unitary group are, as expected, examples of groups. As we suggested in Chapter 1, the unitary group can be defined as follows:

$$U(d) = \{U \in \mathrm{Mat}_{d\times d}(\mathbb{C}) \mid U^\dagger U = \mathbb{1}_d\}$$

where $\mathrm{Mat}_{d\times d}(\mathbb{C})$ is the set of $d \times d$ matrices with complex entries.

We have not provided an analogous definition for the Clifford group. In this thesis, we are satisfied with the definition suggested in Chapter 1: the Clifford group is the set of circuits composed of gates in $G_C = \{\texttt{H}, \texttt{S}, \texttt{CNOT}\}$ applied to various qubits. In other words, the Clifford group is the set of circuits generated by G_C. However, it is also possible to construct the Clifford group using more group-theoretic tools. In particular, the Clifford group is defined as the *normalizer*[3] of the Heisenberg-Weyl group, which in turn is constructed from Pauli gates.[4]

The way the Clifford group is generated by G_C is slightly different from the way $U(d)$ is generated by $G_U = \{\texttt{H}, \texttt{T}, \texttt{CNOT}\}$. Although the Clifford group for N qubits grows large very quickly with N, it is nevertheless a *finite* subgroup[5] of $U(2^N)$. In particular, the size of the N-qubit Clifford group[6] is given by

$$|\mathcal{C}_N| = 2^{N^2+2N} \prod_{i=1}^{N} (4^i - 1) \tag{2.3}$$

Table 2.1 illustrates this growth for small N.

N	1	2	3	4	5
$\lvert\mathcal{C}_N\rvert$	24	11520	92897280	12128668876800	2541082267845918720 0

Table 2.1: The size of the N-qubit Clifford group for small N.

In contrast, $U(d)$ is truly infinite. Thus, when we say that G_U generates this group, we mean that for any $\hat{U} \in U(d)$ and $\epsilon > 0$, we can produce a circuit $\hat{V}$ using gates in G_U such that $|\hat{U} - \hat{V}| < \epsilon$ where $|\cdot|$ is a suitable norm on $U(d)$.[7]

The last piece of group theory machinery we need is the idea of *orbits*. Some groups act in natural ways on certain sets. The unitary group is a great example of this: unitary operators can not only be composed with one another, but they can also be applied to quantum states. When this is the case, an element's orbit is the set of places it can be sent by the group.

Definition 2.2.2. *Let $(G, \times)$ be a group which acts on a set X, and let $x \in X$. Then, the* orbit *of x under the action of G is*

$$Gx = \{g \cdot x \mid g \in G\}$$

[3] In general, the normalizer in a group G of a subgroup $H \leq G$ is defined by [26, p. 95] as:

$$N_G(H) = \{g \in G \mid gHg^{-1} = H\}$$

[4] For more information, see [30].

[5] A subgroup is a subset H of a group $(G, \times)$ where $(H, \times)$ is also a group in its own right.

[6] This equation, from [5, 20], is only consistent with similar equations up to some factor, because sources disagree about whether certain global phases and such should be modded out.

[7] For example, the Hilbert-Schmidt inner product introduced in Section 2.3.3 should suffice.

Therefore, given a group of unitary operators G and a quantum state $|\psi\rangle$, we can produce a set of quantum states by taking the orbit of $|\psi\rangle$ under G:

$$G\,|\psi\rangle = \{\hat{U}\,|\psi\rangle \mid \hat{U} \in G\}$$

2.3 Quantum t-Designs

While *design* is a familiar word, its use in the context of quantum information is likely more foreign. The implicated area of mathematics is called combinatorial design theory, which studies the arrangement of finite sets in ways that satisfy certain notions of "balance" or "symmetry" [29]. Here, we are particularly interested in how "balanced" a distribution of quantum states or operators is in the space of states or operators.

2.3.1 A Useful Analogy: Spherical t-Designs

In this section we are concerned with particular designs called *t-designs*. As an introduction to this idea, we first describe *spherical t-designs*.[8] These can be visualized in multi-dimensional real spaces, which makes them easier to understand than the complex versions we will develop shortly (and which are naturally more useful for quantum mechanics).

Here, we follow the explanation provided in [15]. Consider a function $f : \mathcal{S}^d \to \mathbb{R}$ where $\mathcal{S}^d$ denotes the d-dimensional real sphere.[9] Suppose one wants to know the average value f attains over $\mathcal{S}^d$. In general, this requires knowledge of the function at all points and a potentially difficult integral over $\mathcal{S}^d$. Instead, one could calculate the value of f at some finite set of points in the domain and use this to estimate the average value of f.

Of course, this may or may not provide a useful estimate depending on the points one selects. Furthermore, for any finite set of points, there exist exotic functions whose true average differs an arbitrarily large amount from the estimated average. Such a function could behave one way near the sampled points and behave a very different way in gaps between them. However, the larger the set of sampled points, the more exotic such a function must be. Expressed as a polynomial via a Taylor expansion, significant contributions from higher-order terms would be required. Recognizing that more exotic functions must be

[8] Although the notion of spherical t-designs was understood some time earlier, and the entire field of combinatorial design theory actually arises from recreational mathematics, the concept of spherical t-designs was first put forth by Delsarte, Goethals, and Seidel [9].

[9] Note that the dimension here is the dimension of the surface. Thus, a 2-sphere – perhaps the most familiar example – is the boundary of a 3-dimensional ball. In other words, $\mathcal{S}^d$ is topologically equivalent to $\{(x_1, x_2, \ldots, x_d, x_{d+1}) \mid x_1^2 + x_2^2 + \cdots + x_d^2 + x_{d+1}^2 = 1\}$.

higher degree polynomials, we formalize the quality of finite sets for estimating the average of functions on the real sphere as follows.

Definition 2.3.1. *A set of K points $\{x_i\} \subset \mathcal{S}^d$ is a d-dimensional* spherical t-design *if for every degree t polynomial $p : \mathcal{S}^d \to \mathbb{R}$,*

$$\frac{1}{K}\sum_{i=1}^{K} p(x_i) = \frac{1}{|\mathcal{S}^d|}\int_{\mathcal{S}^d} p(x)\, dA_d$$

where $|\mathcal{S}^d|$ is the area of $\mathcal{S}^d$ and dA_d is the d-dimensional area element on this surface [35].

Remark. *In other words, a set D of K elements in the space $\mathcal{S}^d$ composes a spherical t-design if for polynomials of degree less than or equal to t (polynomials no more exotic than tth order dependence), the estimated average found by measuring p at the points in D is equal to the true average found by measuring p at all points in $\mathcal{S}^d$.*

This notion – and, in particular, that of spherical t-designs in one or two dimensions – is a good mental model for the quantum t-designs we define in the next section and use throughout this thesis.

2.3.2 Projective t-Designs

It is straightforward to imagine this construction being extended to complex numbers instead of real numbers. What results are *complex projective t-designs* (referred to hereafter simply as *projective t-designs*). In particular, d-dimensional projective t-designs are sets of vectors which are evenly distributed on the complex unit sphere in $\mathbb{C}^d$ [36]. This is immediately relevant for quantum mechanical states. Since such states are normalized and can be represented by a vector of complex numbers, we can ask whether a set of quantum states composes a projective t-design.

This is precisely what we intend to do for sets of states evolved by circuits composed primarily of Clifford gates. It is known that the orbits of Clifford circuits compose projective 3-designs but not, in general, projective 4-designs [36]. We seek to understand how many `T` gates must be added to circuits composed mostly of Clifford gates in order for the orbits of these circuits to constitute projective 4-designs.

In our work, we are interested in states of N-qubit systems: states $|\psi\rangle \in \mathcal{H}^{\otimes N}$ for a two-dimensional Hilbert space $\mathcal{H}$. Since such tensor product spaces are 2^N-dimensional, we are generally interested in d-dimensional projective t-designs for $d = 2^N$.

Definitions 2.3.2 and 2.3.3 follow those in [36] and Theorem 2.3.4 is also proved in the same paper. Since a state $|\psi\rangle$ in a d-dimensional Hilbert space is represented by a vector $\psi \in \mathbb{C}^d$, we freely move between the two viewpoints for the remainder of this section.

As promised, the definition of projective t-designs closely mirrors that of spherical t-designs in 2.3.1.

Definition 2.3.2. *A set of K normalized vectors $\{\psi_i\} \subset \mathbb{C}^d$ is a d-dimensional* projective t-design *if for every degree t polynomial p,*

$$\frac{1}{K}\sum_{i=1}^{K} p(\psi_i) = \int p(\psi)\, d\psi$$

where the integral is over the complex unit sphere in $\mathbb{C}^d$ – the space of normalized states.[10]

The core reason to construct t-designs is to be able to work with the sum on the left side of Definition 2.3.2 instead of grappling with the integral on the right side. Thus, using the above definition to show that a given set is a t-design by proving that it matches a full integral defeats the purpose of constructing the set in the first place. Therefore, we need a different route for determining whether a set is a t-design, preferably one amenable to computational analysis.

The *frame potential* of a set of vectors provides a different way to identify projective t-designs. Frame potentials are related to the broader mathematical concept of frames, elaborated on in the context of frame potentials in [23].

Definition 2.3.3. *The tth frame potential of a set of K normalized vectors $\{\psi_i\} \subset \mathbb{C}^d$ is defined as:*

$$\Phi_t(\{\psi_i\}) := \frac{1}{K^2}\sum_{i,j} |\langle\psi_i|\psi_j\rangle|^{2t}$$

Describing this construction as a "potential" makes some sense. It is a sum over all pairs of vectors in a set, adding up powers of the inner product of each pair. All other things being equal, the frame potential is increased if two vectors have a greater overlap. Conversely, it is minimized when the vectors are as far apart from one another as possible. Like the distribution of charge over the surface of the sphere, this occurs when the vectors are evenly distributed around the complex unit sphere.

[10] Although this definition mirrors that of spherical t-designs in Definition 2.3.1, we have been less rigorous here. Thus, the interested reader should note that (1) the measure $d\psi$ is the normalized uniform measure, removing the need for a normalization factor in front of the integral and (2) the set of degree t polynomials considered is $\mathrm{Hom}_{(t,t)}(C^d)$, the space of polynomials homogeneous of degree t in both the coordinates of ψ and the coordinates of ψ^*. More information on (2) can be found in Appendix G of [36].

This is the same notion of balance that t-designs seek to capture. The correspondence between the two is formalized by the following theorem.

Theorem 2.3.4. *$\{\psi_i\}$ is a d-dimensional projective t-design if and only if*

$$\Phi_t(\{\psi_i\}) = \frac{1}{W_t}$$

where W_t is shorthand for the Welch bound, $\binom{d+t-1}{t}$ [32].[11]

Further matching our notion of a projective t-design minimizing the tth frame potential, $\Phi_t(\{\psi_i\}) \geq \frac{1}{W_t}$ in general with the equality only arising in the case where $\{\psi_i\}$ composes a projective t-design [36].[12]

Using this theorem, we can computationally generate finite sets of states evolved by quantum circuits and, if a set's frame potential saturates the Welch bound, conclude that we have produced a projective t-design.

2.3.3 Unitary t-Designs

The notion of t-designs for quantum states was extended to quantum operators in [8]. In this construction, the unitary group $U(2^N)$ replaces the real unit sphere (in the case of Section 2.3.1) or the complex unit sphere (in the case of Section 2.3.2) [15].

The definition of *unitary t-designs* takes the same form as Definitions 2.3.1 and 2.3.2.

Definition 2.3.5. *A set of K unitary operators $\{U_i\} \subset U(d)$ is a d-dimensional* unitary t-design *if for every degree t polynomial p,*[13]

$$\frac{1}{K}\sum_{i=1}^{K} p(U_i) = \int p(U)\, d\mu_H(U)$$

where the integral is over $U(d)$ with respect to the normalized Haar measure.[14]

As before, we can define a frame potential for a set of unitary operators.

[11] And, for reference, $\binom{n}{k} := \frac{n!}{k!(n-k)!}$.

[12] Like our binomial coefficient above, this is sometimes called the Welch bound [32]. It is also called the Sidelnikov inequality [28].

[13] As in Definition 2.3.2, the set of degree t polynomials considered is $\mathrm{Hom}_{(t,t)}(U(d))$, the space of polynomials homogeneous of degree t in both the elements of U and the elements of $U^\dagger$ [36].

[14] In general, a measure is a map from subsets of some space to positive real numbers, providing a way to associate lengths (or areas, volumes, etc.) to regions of some space [24]. In physics, we explicitly construct various measures for integrals in one or more dimensions. It is more difficult to construct an appropriate volume element for groups. The Haar measure

Definition 2.3.6. *The tth frame potential of a set of K unitary operators* $\{U_i\} \subset U(d)$ *is defined as:*

$$\Phi_t(\{U_i\}) := \frac{1}{K^2} \sum_{i,j} \left| \mathrm{Tr}\left[U_i^\dagger U_j\right] \right|^{2t}$$

This definition is directly analogous to that in Definition 2.3.3. In fact, the only notable difference – the replacement of the inner product $\langle \psi_i | \psi_j \rangle$ with the trace $\mathrm{Tr}\left[U_i U_j^\dagger\right]$ – is actually a natural substitution. The trace $\mathrm{Tr}\left[U_i U_j^\dagger\right]$ is called the *Hilbert-Schmidt inner product* and it is a natural inner product for matrices, especially in physics [19].

Theorem 2.3.7. $\{U_i\}$ *is a d-dimensional unitary t-design if and only if*

$$\Phi_t(\{U_i\}) = \gamma(t, d)$$

where $\gamma(t, d)$ *is "the number of permutations of* $\{1, 2, \ldots, t\}$ *with no increasing subsequence of length larger than d" [36].*

Remark. *Although* $\gamma(t, d)$ *is hard to write down explicitly in general, it is easy when* $d \geq t$*. And, since* $d = 2^N$ *for N qubits, all multi-qubit systems satisfy* $d \geq t$ *for* $t = 3$ *or* $t = 4$*. In this case, no subsequences can have length larger than d, so* $\gamma(t, d)$ *merely counts the number of permutations of* $\{1, 2, \ldots, t\}$*. Therefore,* $\gamma(t, d) = t!$ *in this case.*

Like the Welch bound for projective t-designs, $\gamma(t, d)$ is a lower bound for frame potentials of sets of unitary operators [36].

Now, we have two different perspectives on t-designs in quantum mechanics: one describes the distribution of states and the other describes the distribution of operators. Both are valuable individually, and a connection is provided by the following theorem due to [36].

Theorem 2.3.8. *Any orbit of a unitary group t-design forms a projective t-design.*

Therefore, if a unitary t-design also forms a group, we can analyze its distribution by analyzing the circuits themselves – measuring their frame potential – or by analyzing states evolved by the circuits.

---

is notable because it assigns an *invariant* volume element; that is, for $u \in U(d)$,

$$\mu_H(u) = \mu_H(uv) = \mu_H(vu) \text{ for all } v \in U(d)$$

as further elaborated on in [6]. Like the measure used for the complex unit sphere in Definition 2.3.2, this Haar measure is normalized. As a result, it is suitable for treating $U(d)$ as a probability distribution. This, in addition to the Haar measure's invariance property, makes it the suitable measure for selecting random quantum gates for many purposes in quantum information [8].

2.4 Big O Notation and Complexity

We close this chapter with a brief discussion of big O notation and complexity.

Definition 2.4.1. *Let $f, g : \mathbb{R} \to \mathbb{R}$. Then, $f(x) = O(g(x))$ if there exist $N, C \in \mathbb{R}$ such that*

$$|f(x)| \leq C|g(x)| \text{ for all } x > N$$

This captures the notion that, asymptotically, f grows no faster than g. This is a very important idea in computer science. One is often interested in the time or space complexity of an algorithm – that is, how the length of time a process needs to run or the space it needs to use varies with the size of the input.[15]

For example, consider an algorithm which multiplies two $d \times d$ matrices A and B together in the obvious way: by taking the dot product of the ith row of A and the jth row of B for each $1 \leq i, j \leq d$. The time cost of each dot product – which is composed of d addition and d multiplication operations – is proportional to d. Since there are d^2 of these operations, the time complexity is $O(d^3)$. Also, the algorithm needs to store the two matrices, which have d^2 elements each, so the space complexity is $2d^2 = O(d^2)$.

When assessing algorithms, it makes sense to only consider the asymptotic behaviour. We're most interested in how the algorithms behave when given difficult tasks – large inputs – and behaviour for small inputs can be dependent on factors that are unimportant asymptotically. This is illustrated by our benchmarking results in Figure 4.5 (in Section 4.2), which depicts how simulation runtimes vary with system size for our unitary frame potential algorithms.

[15] Although big O notation technically asserts only an upper bound on growth, we strive to use it as a tight bound in this thesis. That is, although $4x^2 + 10 = O(x^3)$ technically, we'd rather write $4x^2 + 10 = O(x^2)$.

Chapter 3

Methods

3.1 Framing Our Methods

3.1.1 Primary Tools

Since quantum mechanical objects like states and operators can be represented as vectors and matrices, many of our methods rest on a foundation of linear algebra tools. Gates in a circuit can be composed using matrix multiplication, and important aspects of our system are embedded in traces and eigenvalues of these circuits or in properties of the states they evolve.

While it's appealing to derive analytical results using abstract properties of these matrices, that's a very difficult task for many of the characteristics we're interested in studying. Whereas the Clifford group and $U(2^N)$ are both groups, with lots of helpful mathematical structure, the space of circuits with varying `T` gate density between these groups lacks this helpful group structure. Thus, it is difficult to make analytical conclusions about this space. Therefore, we rely primarily on computational methods, and the emphasis on linear algebra drives us to make heavy use of well-established linear algebra packages.

Our programming language of choice is Python. While strong linear algebra capabilities are present in most standard languages, Python has several benefits. It offers a convenient balance of object-oriented and functional programming, and it is especially notable for its flexibility and readability. However, these strengths come at the cost of speed, suggesting that a language optimized for matrices like MATLAB or a relatively low-level language like C++ could be effective. Nevertheless, Python packages like `numpy` allow the use of more prim-

itive C functionality, providing speed in Python comparable to traditionally faster languages.

In what follows, we translate the discussion of Chapter 2 into practical terms, discussing the relevant methods for studying random quantum circuits. We first describe how quantum states and circuits are generated, and then discuss how these objects are used to study t-designs. To complete the chapter, we discuss the goals and challenges underlying this work in Section 3.4 and explain how to overcome these challenges in Section 3.5.

3.2 Generating States and Circuits

For the computations we perform, it is necessary to generate quantum states and quantum circuits, and to evolve states through circuits. In many cases, each realization of a measurement requires a new random state and circuit. We describe the methods underlying these processes below. In this section and throughout this chapter, we take inspiration from [25].

In [25], the initialized state was selected to be a random, unentangled state. Thus, N random phases θ_j were selected and a state was prepared as

$$|\Psi_0\rangle = |\psi_0\rangle \otimes |\psi_1\rangle \otimes \cdots \otimes |\psi_{N-1}\rangle$$

where

$$|\psi_j\rangle = \cos(\theta_j)\,|0\rangle_j + \sin(\theta_j)\,|1\rangle_j$$

and where the subscript on the state (and later on operators) denotes the region of the Hilbert space this space lives in.[1] If storing states as vectors, this can be accomplished in Python using the `numpy.random` package to generate random phases and `numpy.kron()` to combine the vectors corresponding to each qubit into a total state representing $|\Psi_0\rangle$.[2] Alternatively, as discussed in Section 3.5.1, it is beneficial to instead leverage `numpy`'s tensor capabilities, storing states as $2^{\otimes N}$ tensors instead of as 2^N-length vectors.

There are other possible choices for initial states beyond random unentangled states. In [34], initial states for a different but related process are selected to be random states regardless of entanglement. However, this makes it more difficult to isolate the distinctions between the Clifford group and the Unitary group, as a random entangled state could already contain much of the complexity produced by unitary matrices outside of the Clifford group. Alternatively, a *fiducial* state

[1] See Section 2.1 for a further discussion of composite Hilbert spaces and their component spaces.

[2] The vectors are combined via a Kronecker product, performed by `numpy.kron()`, which performs tensor products for vector and matrix representations. More information can be found in Section 2.1.

can be used, which refers to the state:

$$|\Psi_0\rangle = |0\rangle_0 \otimes |0\rangle_1 \otimes \cdots \otimes |0\rangle_{N-1}\,.$$

Each of these alternative initial states is straightforward to implement using random numbers and Kronecker or tensor products.

Next, we initialize a circuit. Each circuit is composed of some predefined number M of gates – often $M = bN^2$ is appropriate, where b is some constant scale factor.[3] Using `numpy.random` functionality, we select M gates from our generating set. Recall that this set may be

$$G_C = \{\mathtt{H}, \mathtt{S}, \mathtt{CNOT}\}$$

for the Clifford group, or

$$G_U = \{\mathtt{H}, \mathtt{T}, \mathtt{CNOT}\}$$

for the Unitary group, or a different generating set. Each of the gates in these sets is a local operator, meaning that it acts on only one or two qubits. Thus, when we generate circuits acting on N qubits, we not only select M gates from the sets above, but also the qubits that they act on. With this information, we can construct full operators on the space of N qubits by taking the tensor product of the selected gate with the identity matrix acting all other qubits. For example, for a `T` gate acting on qubit j, we construct the following operator:

$$\hat{U} = \hat{\mathbb{1}}_0 \otimes \ldots \hat{\mathbb{1}}_{j-1} \otimes \hat{\mathtt{T}}_j \otimes \hat{\mathbb{1}}_{j+1} \cdots \otimes \hat{\mathbb{1}}_{N-1}$$

where $\hat{\mathbb{1}}_j$ denotes the identity operator on the jth qubit. Performing this tensor product in Python yields an operator acting on N qubits.

Having multiplied M operators of this form together, we attain a quantum circuit of length M. At this stage, the quantum circuit can be represented by a single matrix, the product of all the component matrices selected from our generating set (though we improve on this in Section 3.5.1). To evolve a state through a circuit, we simply apply the circuit matrix to the state vector, yielding a final state $|\Psi_1\rangle$.

3.3 Quantum t-Designs

Our overall task with regards to t-designs is to determine the density of `T` gates required for circuits composed primarily of Clifford gates to comprise a unitary 4-design or for their orbit to comprise a projective 4-design. As demonstrated in Theorem 2.3.8, the state perspective and the circuit perspective are related. Thus, we should pursue the easier method.

[3] An appropriate M is one that sufficiently samples the space of possible circuits. In related literature ([25], [34]) this scaling is used, and it is motivated by work in [2].

For most of this project, we thought the easier method was to identify the unitary 4-design transition. This is because we determined Theorem 4.1.1 (discussed later in this thesis) early on, which establishes that a prohibitively large number of states are required to produce a projective 4-design. Unfortunately, we later determined Theorem 4.1.2 (also discussed later), which establishes the same result for unitary 4-designs. So both methods are similarly difficult. Nevertheless, there is some hope – see Section 4.1.3, for example – that an altered version of frame potentials could help overcome Theorems 4.1.1 and 4.1.2.

As a result of all of this, most of the methods developed here are focused on studying unitary t-designs, but the aspects which apply to projective t-designs may yet be useful, as we now understand that projective t-designs are no harder to identify than unitary t-designs.

With this in mind, the most important equation is the definition of the unitary frame potential, our computational foothold in the otherwise abstract world of t-designs.

Definition 2.3.6. *The tth frame potential of a set of K unitary operators $\{U_i\} \subset U(d)$ is defined as:*

$$\Phi_t(\{U_i\}) := \frac{1}{K^2} \sum_{i,j} \left| \mathrm{Tr}\left[U_i^\dagger U_j\right] \right|^{2t}$$

This equation makes clear the tasks we'll need to perform. We need to generate and store K circuits, compute Hermitian conjugates and traces, multiply together gates in circuits to produce a single operator, and multiply pairs of these operators together.

Using the methods for generating and combining random operators developed in Section 3.2, calculating the frame potential and comparing it to $\gamma(t, d)$ is merely an exercise in linear algebra – and really, in correctly applying `numpy` linear algebra tools. However, as emphasized below in Section 3.4.2, multiplying the matrices corresponding to these operators is also computationally intensive. Thus, performing this calculation efficiently is the motivation behind many of the optimizations explained in Section 3.5. For example, with K circuits and K^2 pairs of circuits, calculating the unitary frame potential requires us to multiply a significant number of operators, scaling with at least K^2.

Before we can meaningfully explore the space of mostly-Clifford circuits with judicious numbers of `T` gates, we need to be able to computationally distinguish the frame potentials of Clifford circuits from random universal circuits. In fact, that is still a work in progress – therefore, a necessary step is to calculate the frame potential of Clifford circuits for $t = 4$ to a sufficient precision that it's clear it doesn't saturate the lower bound. As discussed in Section 2.2, the Clifford group is actually a finite subgroup of the unitary group, so it's theoretically

possible to calculate its unitary frame potential explicitly. However, Table 2.1 demonstrates how quickly the size of the Clifford group grows.

Therefore, we study the frame potentials of far smaller subsets $\{\hat{U}_i\}$ of the Clifford group. Since such a set will imperfectly represent the entire space of circuits we seek to measure, we expect to observe a trend towards saturation or its absence as we increase K, for appropriately long circuits. This is a way we hope to get around the scaling of complexity with K^2 mentioned above: ideally, a pattern will become apparent for sufficiently small K that we can extrapolate to larger K.

3.4 Benchmarking

3.4.1 Computational Goals

As outlined in Section 1.1.6, the complexity of our computations scales exponentially with the number of qubits N. Furthermore, it scales with the size of our circuits (how many gates they're composed of) and with the number of samples we take for a given measurement. However, it's appealing to increase these numbers as much as possible while our computations can still be achieved in reasonable time scales.

Large N is important because we want to explore the properties of quantum many-body systems: we're interested in how irreversibility emerges from complex quantum systems. $N \to \infty$ represents the *thermodynamic limit* where we expect to see the same properties – like irreversibility – that are apparent in classical systems. Of course, $N \to \infty$ is difficult to achieve. Instead, we settle for N between 8 and 16, and ideally as high as 20 where possible.[4]

Large circuits are important to ensure that we appropriately sample the spaces of circuits we explore. As discussed in Section 2.2, a space of circuits G is *generated* by a set a gates S_G when circuits composed of gates in S_G can arbitrarily approximate any gates in G. This says nothing of how many gates from S_G are required to sufficiently approximate a given circuit in G. If we use too few gates, we may only sample a small – and potentially unrepresentative – portion of G. It is usually sufficient to use a number of gates proportional to N^2 (or, in some cases, another power of N) [2].

[4] This will allow us to compare to similar results in [25] and [34]. These goals are bold for circuits, which are much harder to simulate than states; we have so far only studied up to $N = 13$.

A large number of samples is important for more intuitive reasons. The more times we perform a measurement, the more certainty we can have about its result!

3.4.2 Computational Challenges

Performing computations about quantum mechanical systems is more challenging than analogous computations for classical systems for several reasons.

The largest challenge in performing these computations is the exponential dependence of quantum systems' complexity on N. For this reason, the space complexity of arbitrary states is $O(2^N)$ and the space complexity of arbitrary operators is $O(2^N \times 2^N) = O(2^{2N})$. Furthermore, this exponential complexity propagates through other processes. For example, matrix multiplication for $n \times n$ matrices has a time complexity of $O(n^k)$ for some k, so the time complexity of multiplying $2^N \times 2^N$ matrices is $O(2^{kN})$. This exponential complexity is a major difference between quantum and classical systems, as discussed in 1.1.6. In classical systems, tasks whose time complexity is $O(2^N)$ for an N-bit system are understood to be difficult to compute.

Complexity aside, these $2^N \times 2^N$ matrix operations require a massive amount of storage space. If $N = 20$, the largest value we aim for, then a matrix composed of 64-bit numbers takes up:

$$\left(2^{20} \times 2^{20} \text{ matrix elements}\right) \times \frac{64 \text{ bits}}{\text{matrix element}} \times \frac{1 \text{ byte}}{8 \text{ bits}} \times \frac{1 \text{ TB}}{(1024)^4 \text{ bytes}} = 8 \text{ TB}$$

(where a byte is 8 bits and SI prefixes are used for larger numbers of bits, with factors of $2^{10} = 1024$ replacing factors of 1000). Even if $N = 15$, and the storage space required is $2^5 \times 2^5 = 2^{10} = 1024$ times smaller, 8 GB are still required. The RAM – *Random Access Memory*, the temporary storage space of a computer, which can be accessed much faster than permanent storage (like a hard disk) – of modern computers, while much larger than that of older computers, is nevertheless often only 8 or 16 GB. Consequently, we need to be wary of even storing such large matrices in memory.

This time and space complexity is further compounded by the need for many gates in each circuit and for many samples discussed in 3.4.1. A quantum circuit is the result of multiplying various individual matrices corresponding to gates, and the time required for producing these circuits scales with the number of circuits we sample.

3.5 Overcoming Computational Challenges

Over the course of this thesis work, we have implemented several methods for reducing the space and time complexity of our computations and encountered a couple of interesting dead ends. These successful and unsuccessful attempts are described here in two sections. In Section 3.5.1, we introduce improved methods for representing and storing quantum gates and circuits. In Section 3.5.2, we discuss ways to improve the computational power and efficiency of our simulations.

While all of these methods are targeted primarily at the problem of calculating frame potentials to characterize t-designs, all of them are also generally useful for computations regarding quantum circuits.

3.5.1 Optimizing Gate and Circuit Representations

`Circuit` and `Gate` Objects

As discussed in Section 3.2, the fundamental operators we're selecting from generating sets and combining to form circuits are local operators. They act on only one or two qubits, and thus can be represented by 2×2 or 4×4 matrices – much more manageable matrices than the $2^N \times 2^N$ matrices we deal with in general. This suggests a path for improving on the naïve approaches of using full $2^N \times 2^N$ matrices produced by Kronecker products and of storing circuits as products of these matrices.

Instead of these approaches, we generate and store circuits as `Circuit` objects – collections of component `Gate` objects, where each `Gate` stores its relevant matrix and the qubits it acts on. This has several advantages.

First, it significantly reduces the space complexity of a quantum circuit, making it $O(N)$ instead of $O(2^{2N})$. Second, it makes it straightforward and efficient to find the Hermitian conjugates of circuits: the relation $(\hat{U}\hat{V})^\dagger = \hat{V}^\dagger\hat{U}^\dagger$ means we can simply reverse the `Circuit` and take the Hermitian conjugate of each gate individually, without performing any operations on a gigantic $2^N \times 2^N$ matrix. Third, it lends itself to further optimizations like those described in Section 3.5.1.

Tensor Representations

A further improvement enabled by storing circuits as `Circuit` objects is that when we apply a `Gate` to a state or compose two `Gate` objects, we need not do

a full matrix multiplication. Instead, we can perform an operation on only the relevant qubit sites.

To accomplish this, we eschew `numpy` matrices in favour of the package's more general tensor objects (see Section 2.1 for more background on index notation). Let μ_i and ν_i be tensor indices that attain values in $\{0, 1\}$, corresponding to the two dimensions of our qubit Hilbert spaces. Then, a single-qubit state $|\psi\rangle$ can be represented by the tensor ψ^μ, a tensor product of pure qubit states $|\psi_0\rangle \otimes |\psi_1\rangle \otimes \cdots \otimes |\psi_N\rangle$ can be represented by the tensor $\psi_0^{\mu_0}\psi_1^{\mu_1}\ldots\psi_N^{\mu_N}$, and an entangled N-qubit state $|\Psi\rangle$ can be represented by the tensor $\Psi^{\mu_0\mu_1\cdots\mu_N}$. Similarly, an operator $\hat{U}$ acting on one qubit can be represented by $U^\mu{}_\nu$ and an operator operating on N qubits can be represented by $U^{\mu_0}{}_{\nu_0}{}^{\mu_1}{}_{\nu_1}\ldots{}^{\mu_N}{}_{\nu_N}$.

With this framework in mind, when applying local operators to states we need only sum over one or two indices, depending on how many qubits an operator acts on. This summing process can be accomplished using `numpy.tensordot`. Similarly, when combining multiple operators, as we must do to calculate the frame potential, we need only sum over operator indices corresponding to the qubits those operators act on. Of course, as we combine operators, they will tend to act on increasing numbers of qubits as the various component Hilbert spaces become entangled with one another. As a result, the tensor structure speedup in this case is diminished for long circuits.

This latter point suggests that we can combine single-qubit operators at a low computational cost, and that combining operators operating on distinct sets of qubits is also straightforward. Thus, it is advantageous to find methods which perform as many computations as possible while the component operators are relatively unentangled (with components operating on a small number of qubits).

In this framework, the trace of an operator becomes a large tensor contraction. If $\hat{U}$ is represented as $U^{\mu_0}{}_{\nu_0}{}^{\mu_1}{}_{\nu_1}\ldots{}^{\mu_N}{}_{\nu_N}$

$$\mathrm{Tr}\Big[\hat{U}\Big] = \hat{U}^{\mu_0}{}_{\mu_0}{}^{\mu_1}{}_{\mu_1}\ldots{}^{\mu_N}{}_{\mu_N}$$

Furthermore, we can describe the process of multiplying two operators and taking the resulting trace as one expression where all the operators' shared indices are contracted and all the remaining indices are contracted pairwise.[5]

A Related Dead End: Sparse Matrices

A natural path for us to explore was the use of sparse matrices instead of normal (dense) matrices or tensors. The distinction between sparse and dense matrices

[5] This has not yet been implemented, but deserves testing in the future as it may be more efficient than the processes being conducted separately.

arises because there are two natural ways to store the information in a matrix: keeping the value of every component (dense), or keeping a list of nonzero components and their locations (sparse). The former is advantageous when a matrix has mostly non-zero elements and the latter is advantageous when it has few non-zero elements.

The sparse matrix approach is generally helpful when the focus is on states, since one can apply a single operator at a time – since these operators are local, they truly will be sparse. However, if the object of interest is the circuit itself, sparse matrices are less helpful. Our experience with initial tests was that as more and more local operators were combined, the resulting circuit became increasingly dense until sparse matrix methods became a liability. Furthermore, we don't expect that sparse operators acting on states should be significantly more efficient than local tensor operators, which already avoid carrying unneeded information. Thus, we concluded that sparse matrices were not a helpful method for this project.

3.5.2 More Efficient and Robust Computations

More Raw Power

Regardless of any clever optimizations we choose to make, it's always better to run our computations on a more powerful machine. A relatively easy way to do this was to arrange access to a Pomona College virtual machine (VM). Virtual machines are essentially large slabs of processing and computing power partitioned into individual "computers" by a VM management software. As a result, it is straightforward to allocate further computing resources to an existing VM.

Over the course of this thesis, we were allocated a Pomona College VM which, over time, grew to have 48 cores and 500 GB of RAM, the maximum for that setup. This has helped support our other optimizations, especially the parallel processing described below.

In the future, it could be beneficial to migrate to a larger VM or a supercomputing cluster, both of which Pomona has access to.

Parallel Processing

The Pomona College VMs have large numbers of relatively weak processor cores. Thus, it's advantageous to simultaneously assign these cores independent tasks so that together they can make up for their individual weakness. This is the central idea behind parallel computing.

In particular, each of our K circuits can be converted to a single operator without impacting the other circuits. Similarly, each of the K^2 terms in the unitary frame potential sum can be computed independently. Let C be the number of processor cores our VM has. Since we can do the K circuit conversions asynchronously and then calculate the K^2 terms asynchronously, we can use $\max(C, K)$ cores for the first step and then $\max(C, K^2)$ cores for the second. Thus, with 48 cores in our VM, we'll certainly be able to make use of all of them for the second step, and often will for the first step as well (when $K \geq 48$).

To effect this in Python, we use the `Pool` class in the `multiprocessing` library. This allows us to make a `Pool` of workers – usually, one for each processor – and assign them to work on a list of tasks asynchronously.

A Temporary Dead End: `Numba`

`Numba` is a Python package designed for optimizing code. It re-compiles certain types of Python operations so that they run more efficiently. This type of tool would be very helpful for our purposes, and could even naturally implement the parallel processing described above.

However, there is a trade-off: among other things, `Numba` requires that every variable have a rigid and unchanging *type*.[6] Unfortunately, this includes fixing the shape of all `numpy` tensor objects. This makes `numba` incompatible with our tensor implementation of gates, because tensor products and contractions change the tensors' shapes.

Nevertheless, `Numba` is still a relatively young service and is in active development. There's no reason that it or a similar package couldn't perform the same tasks while remaining compatible with shape-changing tensors – thus, more efficient Python compilers are a worthwhile tool to explore further in the future.

[6] Like `Integer` or `String`. This is unlike Python in general, which is generally very liberal about types.

Chapter 4

Results and Discussion

Our primary aim was to determine the density of `T` gates necessary for circuits composed primarily of Clifford gates to compose a unitary 4-design. Unfortunately, we did not achieve this task. In fact, we can demonstrate that the question is not tractable using our original method – calculating the frame potentials defined in Sections 2.3.2 and 2.3.3. However, we have identified and begun to explore a new approach which has the potential to be successful.

The intractability of the problem was not apparent for some time, primarily because an error in our simulations led to reasonable-seeming results. Therefore, we directed our efforts towards speeding up our simulations. In the process, we built a robust engine for efficiently generating quantum circuits of arbitrary architectures. This engine has been and will continue to be useful for exploring the emergence of irreversibility. In general, there seems to be dual information stored both in states and in circuits, exemplified by Theorem 2.3.8, which connects the two frame potentials defined in Equations 2.3.3 and 2.3.6 respectively. Since states are generally easier to compute than circuits, previous research efforts (like in [4] and [34]) have primarily leveraged information in evolved states to compare Clifford and random universal circuits; this engine enables us to analyze circuits directly.

In this chapter, we begin in Section 4.1 by discussing the t-design transition, explaining why it cannot be identified computationally using our original method, exploring the misleading results we got along the way, and proposing a new method which circumvents the flaws in our original method. Second, in Section 4.2, we discuss and benchmark the major computational steps we took to improve our circuit-generating engine. Third, in Section 4.3, we demonstrate the utility of our circuit-building engine by using it to explore patterns in quantum circuits that may mark the transition from Clifford circuits to random universal circuits.

4.1 Analyzing the t-Design Transition

4.1.1 The Intractability of our Original Frame Potential Approach

We noticed the intractability of our original frame potential approach in two steps. First, we found that making useful conclusions using the projective frame potential was unfeasible. As a result, we turned to the unitary frame potential instead. Second, we found that making useful conclusions using the unitary frame potential was also unfeasible.

Since the two frame potentials are unhelpful for very similar reasons, the arguments for each are made side by side. Unfortunately, there was a period of some five months between these realizations. Hence, after the first realization, we focused our attention on using the unitary frame potential. Although we knew that simulating operators would be far more difficult than simulating states, we hoped that this would be made up for by the seemingly more attainable bound ($t!$ instead of $\frac{1}{W_t}$). Unfortunately, as the proofs below demonstrate, this was not the case.

Recall that the two frame potentials were defined as follows.

Definition 2.3.3. *The tth frame potential of a set of K normalized vectors $\{\psi_i\} \subset \mathbb{C}^d$ is defined as:*

$$\Phi_t(\{\psi_i\}) := \frac{1}{K^2} \sum_{i,j} |\langle \psi_i | \psi_j \rangle|^{2t}$$

Definition 2.3.6. *The tth frame potential of a set of K unitary operators $\{U_i\} \subset U(d)$ is defined as:*

$$\Phi_t(\{U_i\}) := \frac{1}{K^2} \sum_{i,j} \left| \mathrm{Tr}\left[U_i^\dagger U_j\right] \right|^{2t}$$

In each case, a larger K means a larger set of states or operators, and therefore a better chance of approximating all of the complex unit sphere or $U(d)$. However, a larger K also means a more difficult computation: the cost of calculating each frame potential should be roughly proportional to K^2. Unfortunately, we can prove that the values of K necessary for the frame potentials to saturate their lower bounds[1] are prohibitively large. In particular, the required value of K grows exponential with N, the number of qubits in our system.

We prove this in Theorems 4.1.1 and 4.1.2. For both, the key issue lies in the $i = j$ terms of the frame potential sums. However well-distributed the states or

[1] Recall that saturating the Welch bound or its unitary group equivalent is equivalent to being a t-design by Theorems 2.3.4 and 2.3.7 respectively.

operators are (reducing the various inner products), the $i = j$ terms of the sums always reduce to the norms of single elements, which remain relatively large.

Theorem 4.1.1. *Establishing the condition in equation 2.3.4 requires K to be $O(2^N)$.*

Proof First, note that $|\langle\psi_i|\psi_j\rangle|^{2t} = 1$ whenever $i = j$, which happens K times in our sum for the tth frame potential. Furthermore, note that $|\langle\psi_i|\psi_j\rangle|^{2t}$ is always positive for any i, j. Thus, it follows that

$$\Phi_t(\{|\psi_i\rangle\}) = \frac{1}{K^2}\sum_{i,j}|\langle\psi_i|\psi_j\rangle|^{2t} \tag{4.1}$$

$$= \frac{1}{K^2}\sum_{i\neq j}|\langle\psi_i|\psi_j\rangle|^{2t} + \frac{1}{K^2}\sum_{i}|\langle\psi_i|\psi_i\rangle|^{2t} \tag{4.2}$$

$$= \frac{1}{K^2}\sum_{i\neq j}|\langle\psi_i|\psi_j\rangle|^{2t} + \frac{1}{K} \tag{4.3}$$

$$\geq \frac{1}{K}. \tag{4.4}$$

Next, consider

$$W_t := \binom{d+t-1}{t} = \frac{(d+t-1)!}{(d-1)!t!} = \frac{1}{t!}\prod_{k=0}^{t-1}(d+k)$$

Since k in the product above is always greater than or equal to zero,

$$W_t = \frac{1}{t!}\prod_{k=0}^{t-1}(d+k) \geq \frac{1}{t!}d^t = \frac{1}{t!}2^{tN}$$

Consequently, if we wish to show that a set of states comprises a projective t-design, we need K to satisfy

$$K \geq \frac{1}{\Phi_t(\{|\psi_i\rangle\})} = W_t \geq \frac{1}{t!}2^{tN} \tag{4.5}$$

As a result, for fixed t, we need K to be $O(2^N)$ in order to show that a set of states comprises a projective t-design. □

Theorem 4.1.2. *Establishing the condition in equation 2.3.7 requires K to be $O(2^N)$.*

Proof Let d be the dimension of the unitary group we're considering. Since each $U_i \in U(d)$, it follows that $U_i^\dagger U_i = \mathbb{1}_d$, the $d \times d$ identity matrix. Consequently, $\mathrm{Tr}\left[U_i^\dagger U_j\right] = d$ when $i = j$. This occurs K times in our sum for the tth

frame potential. Also, each term in the sum is a positive real number due to the absolute value taken. Thus,

$$\Phi_t(\{U_i\}) = \frac{1}{K^2}\sum_{i,j}\left|\mathrm{Tr}\left[U_i^\dagger U_j\right]\right|^{2t} \tag{4.6}$$

$$= \frac{1}{K^2}\sum_{i\neq j}\left|\mathrm{Tr}\left[U_i^\dagger U_j\right]\right|^{2t} + \frac{1}{K^2}\sum_{i}\left|\mathrm{Tr}\left[U_i^\dagger U_i\right]\right|^{2t} \tag{4.7}$$

$$= \frac{1}{K^2}\sum_{i\neq j}\left|\mathrm{Tr}\left[U_i^\dagger U_j\right]\right|^{2t} + \frac{d^{2t}}{K} \tag{4.8}$$

$$\geq \frac{d^{2t}}{K} \tag{4.9}$$

Since $\gamma(t,d) = t!$ in all the cases we're concerned with, showing that a set of operators comprises a unitary t-design requires that K satisfies

$$K \geq \frac{1}{t!}d^{2t}$$

Since $d = 2^N$, we again need K to be $O(2^N)$ in order to show that a set of states comprises a projective t-design for some fixed t. □

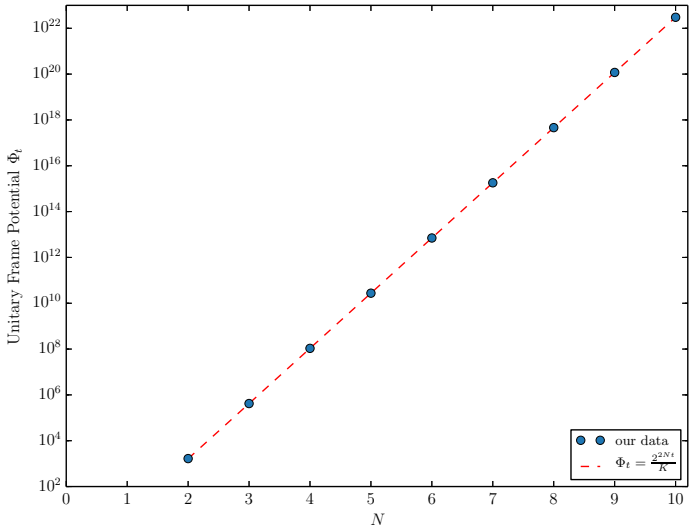


Figure 4.1: Φ_4 data for a range of N values are plotted alongside $\frac{2^{2Nt}}{K}$, the diagonal portion of the unitary frame potential sum. The measured data was gathered for Clifford circuits with $K = 40$ and $b = 20$.

As suggested in Section 3.3, we had hoped that even if we couldn't reach high enough K to identify a set as a t-design, we could instead identify a trend in

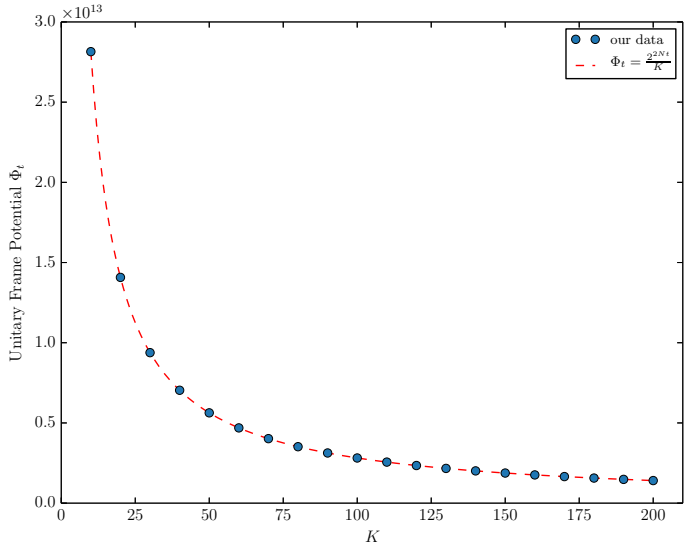


Figure 4.2: Φ_4 data for a range of K values are plotted alongside $\frac{2^{2Nt}}{K}$, the diagonal portion of the unitary frame potential sum. The measured data was gathered for Clifford circuits with $N = 6$ and $b = 20$.

Φ_t and use it to determine whether or not Φ_t ever saturated its lower limit. However, the second term in Equations 4.4 and 4.9 is sufficiently large to dominate the frame potential for the values of K we can test. This is demonstrated for the unitary frame potential by Figures 4.1, 4.2, and 4.3. In each figure, the measured unitary frame potential is compared with the second term of Equation 4.9 (with $d = 2^N$). The growth of this term with N, K, and t matches our data very closely, strongly implying that this term is entirely dominant in the regime we can simulate.

The data in 4.1, 4.2, and 4.3 are composed of just one measurement per point, explaining the absence of error bars. This choice was made since we knew the data would not be scientifically useful and that the adherence to the dominating term of Equation 4.9 was clear.

Unfortunately, in this chapter we never display t-design data from circuits architectures other than entirely-Clifford and random unitary – we don't explore the case of `T` gates judiciously injected into mostly-Clifford circuits. This is intentional. There is no reason to explore the transition from 3-design to 4-design using these circuit architectures unless we can first make sufficiently accurate measurements to distinguish Clifford circuits from 4-designs.

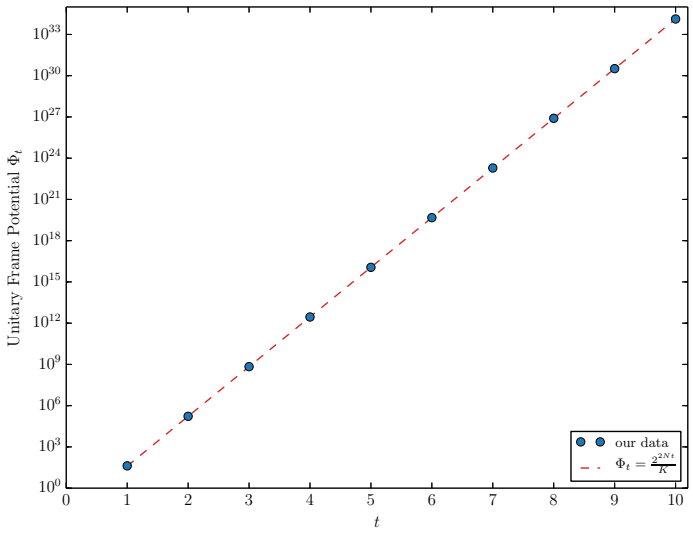


Figure 4.3: Φ_t data for a range of t values are plotted alongside $\frac{2^{2Nt}}{K}$, the diagonal portion of the unitary frame potential sum. The measured data was gathered for Clifford circuits with $N = 6$, $K = 100$ and $b = 20$.

4.1.2 The Source of Our Confusion

As suggested above, the reason we spent so much time optimizing our simulations despite the intractability of our method (illustrated by Section 4.1.1) is that a bug in our code led us to get reasonable-appearing results.

Table 4.1 (for which $t = 4$, $N = 8$, and $b = 1$ or 10 for various measurements) contains some representative results from our flawed code: (1) some measurements are smaller than $t!$, which should be a lower bound, (2) some measurements look spot-on, and (3) some measurements look far too large. In most cases, the bulk of the data fell into category (2), inspiring misplaced hope. Hoping that the unusually large or small results would disappear for sufficiently large K and b, we sought to optimize our computations.

However, the source of these results was actually a flaw in our gate multiplication method. When gates are represented by tensors, they are multiplied using a tensor contraction, as described in Section 2.1. The essential steps are: determining which indices are shared and must be summed over, performing this sum using `numpy`'s `tensordot` method, and transposing the resulting tensor. This transposition is necessary because the `tensordot` method simply leaves indices

	Unitary Frame Potential	
K	*Clifford*	*random unitary*
25	11.1	17.3
25	25.4	24.9
25	$6.87 \cdot 10^6$	649
50	20.6	19.1
50	25.4	27.1
50	$6.73 \cdot 10^3$	$6.47 \cdot 10^3$
100	24.9	25.7
100	$1.73 \cdot 10^3$	$1.71 \cdot 10^3$

Table 4.1: Sample unitary frame potential data for Clifford and random universal circuits over a range of K values (with $t = 4$, $N = 8$, and $b \in \{1, 10\}$. Included in this sample data are various plausible-seeming, too-high, and too-low results for the frame potential ($\Phi_4 = 24$ for a unitary 4-design).

in the order it found them after performing contractions. In other words,

$$\left(A^{\mu_1}{}_{\nu_1}{}^{\mu_2}{}_{\nu_2}{}^{\mu_3}{}_{\nu_3}\right)\left(B^{\nu_3}{}_{\alpha_3}{}^{\nu_1}{}_{\alpha_1}\right) = C^{\mu_1\mu_2}{}_{\nu_2}{}^{\mu_3}{}_{\alpha_3\alpha_1}$$

under a `tensordot` contraction when really we would like

$$\left(A^{\mu_1}{}_{\nu_1}{}^{\mu_2}{}_{\nu_2}{}^{\mu_3}{}_{\nu_3}\right)\left(B^{\nu_3}{}_{\alpha_3}{}^{\nu_1}{}_{\alpha_1}\right) = C^{\mu_1}{}_{\alpha_1}{}^{\mu_2}{}_{\nu_2}{}^{\mu_3}{}_{\alpha_3}$$

Accomplishing the latter thus requires an appropriate transposition. Our multiplication method did use a transposition, but in some cases that transposition was simply incorrect.

We can also provide some motivation for why this error yielded plausible-seeming results. A sufficiently incorrect multiplication process becomes something like a random operator generator, especially after many flawed gate multiplications. Thus, we were essentially sampling random operators instead of, for example, the Clifford architectures we intended to sample. And, since random operators are well-distributed in the space of operators, it's not surprising that we often found sets which were unitary t-designs, with their unitary frame potentials saturating the lower bound of $t!$.

Furthermore, since the error in our transpositions was a small mistake, it is possible that the multiplication method was not flawed for two operators acting on every qubit, as would be the case for operators resulting from long circuits. Thus, although many multiplications along the way would be flawed, it's possible that the multiplications used for actually measuring the frame potential was handled accurately, further explaining our plausible results.

Additionally, this explanation accounts for the oddly small and large unitary frame potentials that sometimes occurred. Since our multiplication was ill-behaved, it's possible that the resulting circuits were not unitary or that the

frame potential calculation was flawed. Either of these could lead to a frame potential below the lower bound. Alternatively, if a calculation turned out luckily accurate, it makes sense that the resulting frame potential would be very large: we just showed in Section 4.1.1 that this is inevitable when the frame potential is measured accurately.

Fortunately, we eventually grew suspicious of our multiplication method and were able to track down the bug. Of course, this just led us into the mess of prohibitively large frame potentials described in Section 4.1.1. However, the large $i = j$ terms in the frame potential which cause this problem led us to ask a new question: what if we could just ignore the terms that make this difficult? Section 4.1.3 provides an answer to this question.

4.1.3 A New Way Forward: Ignoring Diagonal Terms in the $K \to \infty$ Limit

As suggested by Theorems 4.1.1 and 4.1.2 and exemplified by Figures 4.1, 4.2, and 4.3, there is little information to be gained from frame potentials for the values of K we can test.

Our primary trouble is that it's difficult to calculate the frame potential for large sets of states or operators, but large sets are *necessary* to build t-designs for most N. If it is possible to build t-designs using a particular architecture of circuit, this behaviour will be most apparent for large K. Thus, in this case we expect the frame potentials to saturate their lower bounds as K tends to infinity.

Trying to find a trend in Φ_t with K was impossible because Φ_t was dominated by the diagonal term $\frac{2^{2Nt}}{K}$ for the K we could test. However, consider the following *off-diagonal*[2] *frame potentials*.[3]

Definition 4.1.3. *The tth off-diagonal frame potential of a set of K normalized vectors $\{\psi_i\} \subset \mathbb{C}^d$ is defined as:*

$$\Phi'_t(\{\psi_i\}) := \frac{1}{K^2} \sum_{i \neq j} |\langle \psi_i | \psi_j \rangle|^{2t}$$

[2] If the terms of the double sum defining our frame potentials were viewed as a matrix, then the $i = j$ terms that we have removed would sit along the diagonal. Thus, the remaining terms are "off-diagonal".

[3] Interestingly, similar off-diagonal constructions are considered for frame energies in [12] and for frame potentials in [33]. We were not aware of these articles when we defined our own off-diagonal frame potentials, and we have not yet confirmed how the constructions in [12] and [33] compare to the more familiar ones we introduced in Section 2.3.

Definition 4.1.4. *The tth off-diagonal frame potential of a set of K unitary operators $\{U_i\} \subset U(d)$ is defined as:*

$$\Phi'_t(\{U_i\}) := \frac{1}{K^2} \sum_{i \neq j} \left| \mathrm{Tr}\left[U_i^\dagger U_j\right] \right|^{2t}$$

Clearly, the off-diagonal frame potentials do not suffer from the same dominating terms which led to Figures 4.1, 4.2, and 4.3. Thus, it's possible that a meaningful trend might exist in Φ'_t for small K that doesn't exist in Φ_t. However, we have not yet shown that Φ'_t is a suitable replacement for Φ_t. Facially, it seems like it wouldn't be – it's not the frame potential! Nevertheless, it turns out to be just as useful.

The following two theorems are almost identical, but both are provided for completeness.

Theorem 4.1.5. $\Phi_t(\{\psi_i\}) = \Phi'_t(\{\psi_i\})$ *in the* $K \to \infty$ *limit. In other words,*

$$\lim_{K\to\infty} \Phi_t(\{\psi_i\}) = \lim_{K\to\infty} \Phi'_t(\{\psi_i\})$$

Proof According to Equation 4.4,

$$\Phi_t(\{\psi_i\}) = \frac{1}{K^2} \sum_{i \neq j} |\langle \psi_i | \psi_j \rangle|^{2t} + \frac{1}{K} \tag{4.4}$$

Therefore,

$$\begin{aligned} \lim_{K\to\infty} (\Phi_t(\{\psi_i\}) - \Phi'_t(\{\psi_i\})) &= \lim_{K\to\infty} \frac{1}{K} \\ &= 0 \end{aligned}$$

□

Theorem 4.1.6. $\Phi_t(\{U_i\}) = \Phi'_t(\{U_i\})$ *in the* $K \to \infty$ *limit. In other words,*

$$\lim_{K\to\infty} \Phi_t(\{U_i\}) = \lim_{K\to\infty} \Phi'_t(\{U_i\})$$

Proof According to Equation 4.9,

$$\Phi_t(\{U_i\}) = \frac{1}{K^2} \sum_{i \neq j} \left| \mathrm{Tr}\left[U_i^\dagger U_j\right] \right|^{2t} + \frac{d^{2t}}{K} \tag{4.9}$$

Therefore,

$$\begin{aligned} \lim_{K\to\infty} (\Phi_t(\{\psi_i\}) - \Phi'_t(\{\psi_i\})) &= \lim_{K\to\infty} \frac{d^{2t}}{K} \\ &= d^{2t} \left(\lim_{K\to\infty} \frac{1}{K} \right) \\ &= 0 \end{aligned}$$

□

As a consequence of these theorems, the off-diagonal frame potentials are suitable for assessing whether or not a set is a t-design: it is if and only if its off-diagonal frame potential tends to the Welch bound or its unitary equivalent. However, having removed the diagonal terms in the sum, it's no longer guaranteed that the Welch bound and its unitary equivalent function as *lower* bounds.

In a sense, these theorems show that the diagonal terms in the frame potentials – which dominate for small K – are really transient[4] with respect to K. In the $K \to \infty$ limit, whether or not a circuit architecture produces t-designs doesn't depend on these diagonal terms.

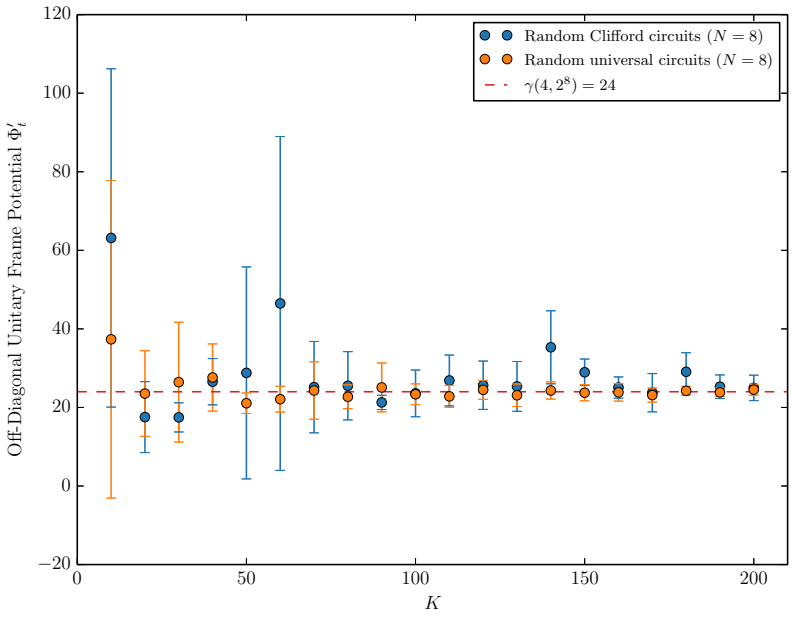


Figure 4.4: This figure is the most faithful to our original goal: it seeks to distinguish Φ'_4 for Clifford circuits from that for random universal circuits to enable subsequent study of intermediate circuit architectures. It depicts Φ'_4 for each circuit architecture over a range of K values (with $N = 8$ and $b = 20$), and also has a reference line at $\gamma(4, 2^8) = 24$. It seems clear from these data that random universal circuits compose unitary 4-designs, as expected, but the data are not sufficient to establish that this is not true of Clifford circuits. Thus, further development of this method is required.

[4]This terminology is meant to recall the decomposition of solutions to damped, driven oscillators into steady and transient components.

We came up with the idea of off-diagonal frame potentials late in this project and well into the writing of this document. Nevertheless, we have conducted an initial study of this new method, the results of which are contained in Figure 4.4.

From these preliminary data, it's clear that distinguishing between random unitary and Clifford circuits remains difficult, even using this new method. However, the large error bars in Figure 4.4 suggest that further trials are required and the fact that the off-diagonal frame potentials for the random universal circuits consistently lie below those for the Clifford circuits is suggestive, if inconclusive.

Although we have convincingly ruled out our original method using frame potentials, this new off-diagonal frame potential approach doesn't suffer the same problems, and thus warrants further study.

4.2 Computational Progress

In the months between our promising preliminary results for the unitary frame potential (described in Section 4.1.2) and our proof of Theorem 4.1.2 (which spelled the end of our initial method), our main aim was to optimize our simulations. In doing so, we hoped to reach circuit lengths and values of K large enough to yield distinguishable results for random unitary and Clifford circuits.

We made computational optimizations in three ways: improving our techniques for representing quantum gates and circuits, increasing our available computational power, and using this power more efficiently. Our representation of gates as tensors instead of matrices and our lazy approach to storing circuits (see Section 3.5.1) both fall in the first category. Our migration from a laptop computer to a Pomona College virtual machine (VM) with, ultimately, 48 cores and 500 GB of RAM falls in the second category. Our implementation of parallel processing for certain key steps of our simulations falls in the third.

Each of these advances has reduced how long it takes to measure the unitary frame potential for a given N, K, and b.[5] Figures 4.5, 4.7, and 4.8 illustrate our progress by showing how the simulation time (ST) – the time it takes to generate a set of circuits and measure the unitary frame potential – varies with N, K, and b. In each figure, the simulation time is given for three different implementations of our simulations. All three implementations use some degree of lazy circuit storage and were benchmarked on the Pomona College VM. We found these optimizations necessary for gathering meaningful benchmark-

[5]Throughout this chapter, N is the number of qubits in the system, K is the number of circuits generated, and b is the coefficient scaling the number of gates in each circuit: bN^2. Although parameters like the circuit architecture and the value of t influence the resulting frame potential, they do not meaningfully impact the time a simulation takes.

ing data. Without lazy circuit storage, it is nearly impossible to measure frame potentials using matrix representations for quantum gates and it is very difficult even using tensor representations if the simulations are run on a laptop. Furthermore, our migration to the Pomona College VM allowed us to run in minutes simulations that would take many hours on a laptop. Therefore, the following three implementations are already significantly more efficient than the naïve approach we took initially.

1. **Unparallelized Matrix (UM) Circuits.** This is the most basic implementation of the three. It uses traditional matrix representations for gates and does not use parallel processing.

2. **Unparallelized Tensor (UT) Circuits.** This is identical to the previous implementation, except that it uses tensor representations for gates instead of matrix representations. Since lazy circuit storage is used in both cases, this doesn't massively impact the amount of memory used. However, it allows for a more efficient gate composition process.

3. **Parallelized Tensor (PT) Circuits.** This is our most advanced implementation, and this is reflected in the benchmarking data below. Again, quantum gates are represented as tensors in this implementation. The biggest change from the previous versions is that parallel processing is implemented: each of our K circuits is converted from a list of gates to a single operator asynchronously, and each term in the frame potential's double sum of circuits is computed asynchronously. To make both of these possible, all of the circuits must be converted to a single operator before any of the frame potential calculations can occur. As a result, we remove one aspect of our lazy circuit storage: previously we had reduced memory usage by only converting two circuits to single operators at a time, but in this implementation we convert all K circuits into operators concurrently. This reduces the amount of time spent multiplying gates together in each circuit (as this is done K times instead of K^2 times) at the cost of using substantially more memory.

There's an important flaw in this benchmarking that we must address. The three different implementations were benchmarked simultaneously on the same machine. Thus, the measured simulation times occurred in a different environment than what would be done in practice, which may have stressed the implementations in different ways.[6] Of course, benchmarking is always sensitive to environmental factors (the machine's processing power, its memory, and other loads on these at the time of benchmarking) but these factors should

[6] In particular, it's possible that different parts of the simulation would become bottlenecks under different circumstances, so that the dominant factors in our benchmarking here do not represent the factors that would dominate in the normal course of business. However, even if this is the case, it's not unusual for multiple simulations to be run concurrently, so these data still may be important.

mostly impact the three different implementations in similar ways, so that they can still be meaningfully compared.

That being said, if any of the implementations was disproportionately impacted by running them all simultaneously, we expect that it would be the PT approach. Unlike the other approaches, it is able to take advantage of all available processing power. Therefore, while the UM and UT implementations wouldn't be significantly impacted by work being done on other processor cores, the PT approach would be.[7] As a result, we conclude that the PT approach is *at least* as good as it seems from the following figures, and quite possibly better, so we feel confident moving forward with this combination of optimizations.

4.2.1 Growth of Simulation Time with N

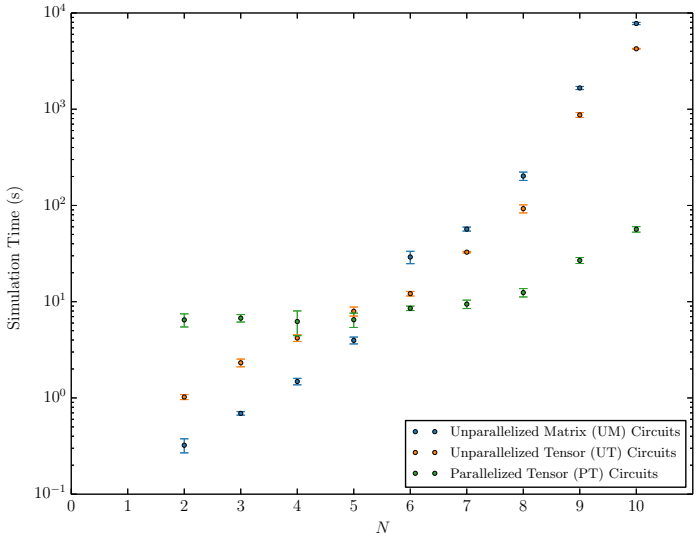


Figure 4.5: The growth of simulation time with N for our three implementations (using Clifford circuits with $t = 3$, $K = 10$, and $b = 1$). Although the implementations perform the opposite of what we would expect for small N, the expected ranking returns for larger N. In the largest system sampled, the PT approach performs approximately 100 times better than the other approaches.

Figure 4.5 shows on a semi-log plot how ST grows with N for the three implementations. For all three, $t = 3$, $K = 10$, $b = 1$ and the circuits are composed of

[7] The second PT data set in Figure 4.8 provides evidence that the PT implementation was, in fact, hampered by being run concurrently with the other implementations.

Clifford gates. We predicted that the PT implementation would perform best, then the UT implementation, and finally the UM implementation. Interestingly, this behaviour is flipped for small N. This is not so surprising. Parallelization adds computational overhead, which can dominate for easy simulations but becomes negligible for difficult simulations. Also, our tensor implementation involves a more complicated (though quicker) operator multiplication process, adding overhead for small N in a similar way.

For larger N, the behaviour of the three implementations is what we expect. In the large-N regime, the UT approach is approximately twice as fast as the UM approach. The PT approach is quicker by a factor which increases with N and, even at $N = 10$, it is already approximately 100 times faster than the other implementations.

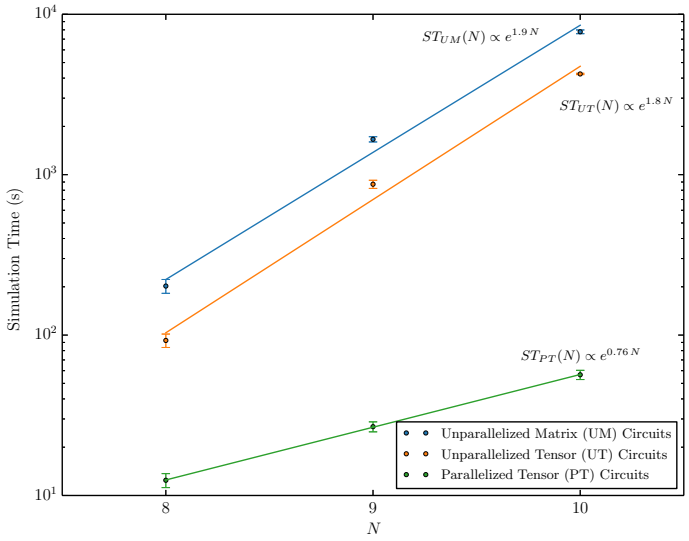


Figure 4.6: This plot contains the three highest-N data points for each implementation from Figure 4.5. Here, roughly exponential behaviour appears to be present in all three implementations, but the PT approach has a significantly slower growth rate than the other two.

The semi-log plot is useful because we expect to find – and, in reality, seem to find – a roughly exponential dependence of ST on N.[8] This is suggested by the linear (on the semi-log plot) behaviour of the UM and UT data sets over all N. In contrast, the PT data set does not seem linear. Nevertheless, it seems to enter a linear regime for larger values of N. In particular, Figure 4.6 fits the three data sets for $N \geq 8$. These fits (which are linear in the semi-log plot, exponential

[8] Recall that exponential behaviour appears linear in semi-log plots.

in reality) suggest that $ST_{UM}(N) \propto e^{1.9\,N}$ and $ST_{UT}(N) \propto e^{1.8\,N}$, indicating much quicker growth than the PT approach, for which $ST_{PT}(N) \propto e^{0.76\,N}$.

4.2.2 Growth of Simulation Time with K

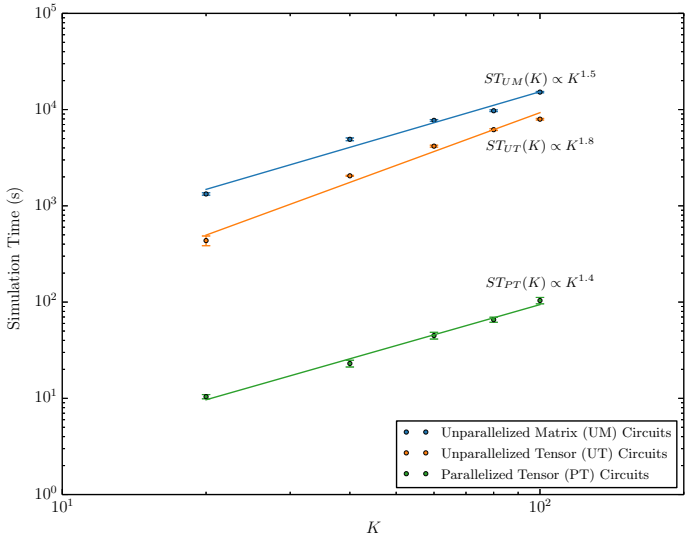


Figure 4.7: The growth of simulation time with K for our three implementations (using Clifford circuits with $t = 3$, $N = 8$, and $b = 1$). All implementations seem to have time complexity that is roughly polynomial in K and, for the largest K sampled, the PT approach performs approximately 100 times better than the other approaches.

Figure 4.7 shows on a log-log plot how ST grows with K for the three implementations. For all three, $t = 3$, $N = 8$, $b = 1$ and the circuits are composed of Clifford gates. Since the number of circuits is linear in K and the number of terms in the unitary frame potential is quadratic in K, it makes sense that ST would be proportional to some power of K for each implementation. The linear appearance of the data in Figure 4.7 supports this.[9]

Here, the performance of the three implementations is ordered as we expect for all K. Again, the UT approach is approximately 2 times faster than the UM approach, and the PT approach is approximately 100 times faster than both of them. The linear fits in the log-log plot imply that $ST_{UM}(K) \propto K^{1.5}$, $ST_{UT}(K) \propto K^{1.8}$, and $ST_{PT}(K) \propto K^{1.4}$.

[9] Recall that power law behaviour appears linear in log-log plots.

The fractional power of K – between 1 and 2 – driving the growth of simulation time for each implementation underscores the difficulty of intuiting time complexities. There are reasons to suggest that the growth of simulation time would be approximately linear or quadratic. Instead, a fractional power law emerges from some combination of these and other, more difficult to identify factors.

4.2.3 Growth of Simulation Time with b

Figure 4.8 shows on a log-log plot how ST grows with b for the three implementations. For all three, $t = 3$, $N = 8$, $K = 10$ and the circuits are composed of Clifford gates. Unlike the previous figures, two data sets are included for the PT approach. This is because the growth of ST with b seemed unusually high for the PT approach in PT1 data set, which was measured concurrently with the UM and UT data sets. The PT2 data set was measured without any other major processes running on the Pomona College VM. In fact, it was the distinction between the PT1 and PT2 data sets here that prompted the realization that running simulations concurrently could produce flawed benchmarking results.

With this in mind, it's not surprising that the PT2 data indicate faster simulations than the PT1 data set – there were more available processor cores. However, it's interesting that the growth of ST with b appears different: when many cores are available, the growth is suggestive of a square root dependence whereas when few are available, the growth is suggestive of a linear dependence.

The three implementations are ordered as we would expect in Figure 4.8: PT1 is about 50 times faster than UT, which is about twice as fast as UM. Also PT2 is about 10 times faster than PT1. The linear fits in the log-log plot imply that $ST_{UM}(b) \propto b^{0.82}$, $ST_{UT}(b) \propto b^{0.62}$, $ST_{PT1}(b) \propto b^{0.97}$, and $ST_{PT2}(b) \propto b^{0.51}$.

In addition to further demonstrating the efficacy of the PT approach, these data demonstrate that in general, and especially for the PT approach, increasing b is not especially impactful on ST. This is reassuring, because increasing b allows us to ensure we are sufficiently sampling the space of architectures we're simulating.

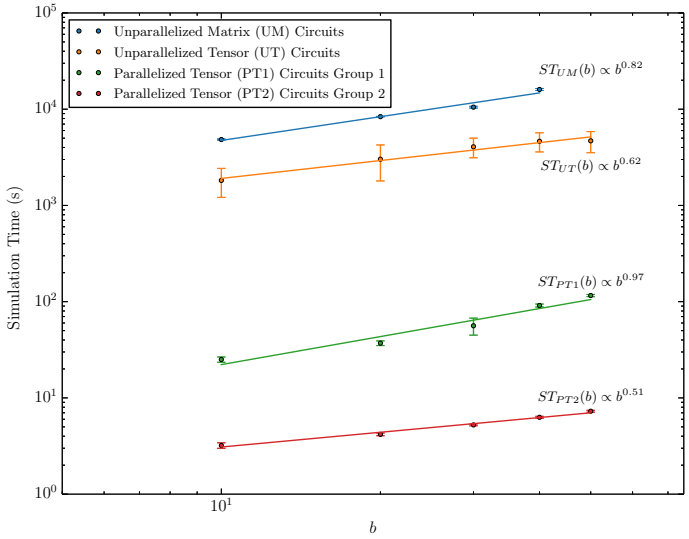


Figure 4.8: The growth of simulation time with b for our three implementations (using Clifford circuits with $t = 3$, $N = 8$, and $K = 10$). Each implementation's simulation time has a relatively forgiving (sub-linear) dependence on b, but the PT approach is a significant factor faster. Interestingly, ST_{PT1} (run concurrently with the other implementations) is approximately linear in b, whereas ST_{PT2} (run alone) scales roughly with $\sqrt{b}$.

4.2.4 Concluding Thoughts

First, it's worth noting that since the UM and UT approaches are *not* parallelized, each uses only one processor core at a time. Thus, separate simulations can be run on different cores without issue, up to the total number of cores. Thus, if one were interested in running 48 iterations of a simulation, then the UM and UT approaches would be 48 times faster, amortized over the various simulations. This makes the UM and UT approaches look somewhat less dismal in comparison to the PT approach.

Nevertheless, the PT approach remains superior for two reasons. First, even accounting for the factor of 48 boost in UM and UT amortized speeds, the PT approach still remains faster in most cases. Second, the ability to run multiple UM and UT simulations in parallel by hand doesn't help make more computationally intense simulations feasible. And that is often what we're interested in. It's difficult to perform even one test for large N (and K and b). The PT approach extends what systems are feasible to simulate, whereas the UM and UT approaches can only simulate already feasible systems in greater numbers.

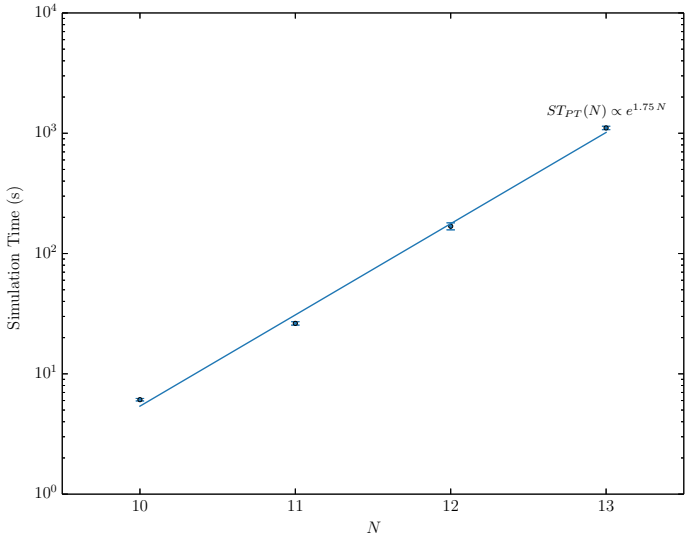


Figure 4.9: The growth of simulation time with N for measurements of the off-diagonal frame potential (using Clifford circuits with $t = 3$, $K = 10$, and $b = 1$). Although this is a PT implementation, the exponential growth with N seems similar to that of the UM and UT implementations in Figure 4.6. This could be a new PT feature for higher N, a more accurate measurement, or indicate some sort of super-exponential growth.

There is still progress to be made in pushing our simulations up to higher N. Figure 4.9 depicts the growth of ST with N for a higher regime than Figures 4.5 and 4.6. Like those figures, $t = 3$, $K = 10$, $b = 1$ and the circuits are composed of Clifford gates. Unlike those figures, only the PT approach is depicted.

The data set in Figure 4.9 differs in two ways from the previous PT data sets in this section. First, it was gathered without other major processes running on the Pomona College VM. Second, it corresponds to measurements of the off-diagonal unitary frame potential described in Section 4.1.3. This also increases the speed of measurements, since this frame potential has fewer terms.[10]

For this implementation, it appears that $ST_{PT}(N) \propto e^{1.75\,N}$, better matching the UM and UP results above. This result is different from that implied by Figure 4.6 – the difference could be due to the larger-N regime or the slightly

[10]This PT implementation not only ignores $i = j$ terms in the sum, but actually only calculates the $i < j$ terms. This is because $\left|\mathrm{Tr}\left[U_i^\dagger U_j\right]\right| = \left|\mathrm{Tr}\left[U_j^\dagger U_i\right]\right|$, so those calculations can be reused for the remaining terms. The same could be done for the normal unitary frame potential, but wasn't when the benchmarking data for this section was taken.

different implementation. Alternatively, the growth of ST with N could be even more severe than the exponential growths we have fitted.

The data in Figure 4.9 go up to $N = 13$. For N larger than this, the memory required in the current implementation exceeds the 500 GB on the Pomona College VM. This is surprising: an individual circuit should only require 2 GB for $N = 14$, leading to 20 GB in total for $K = 10$. Although more memory is required than just storing the circuits, it appears that the implementation is using more memory than it should; fixing this will allow us to probe larger N.

Even if it's not possible to directly reduce the memory used by this PT implementation, there are other ways to circumvent this issue. For example, we could divide our set of K circuits into M partitions and only convert circuit objects to single operators for circuits in two partitions at a time. For each partition, we would loop through the other partitions, generating circuit operators and calculating terms of the frame potential. This allows us to trade processing time for memory: only two partitions worth of circuits are stored in memory at a time, but each circuit needs to be converted to a single operator M times. Therefore, the memory used decreases by a factor of $\frac{M}{2}$ while the processing time increases by a factor of M. This is not ideal, as it would certainly slow our simulations, but it could help make probing larger N possible.

4.3 Our Circuit Engine in Action: Machine Learning Patterns in Quantum Circuits

In this section, we take an approach developed for states in [34] and extend it to entire quantum circuits, demonstrating the utility of our efficient circuit generation engine.

4.3.1 Visualizing States and Circuits

Figure 4.10 depicts an aesthetically and scientifically appealing pattern discussed in [34]. It depicts a state $|\psi\rangle$ in an N-qubit system $\mathcal{H}$ which has been partitioned in two equal-dimensional parts so that $\mathcal{H} = \mathcal{H}_A \otimes \mathcal{H}_B$. Then, for $|\alpha_i\rangle \in \mathcal{H}_A$ and $|\beta_i\rangle \in \mathcal{H}_B$, we can write:

$$|\psi\rangle = \sum_{i,j} \Psi(i,j)(|\alpha_i\rangle \otimes |\beta_j\rangle) \tag{4.10}$$

Here, Ψ is a map from $\{1, \ldots, 2^{N/2}\} \times \{1, \ldots, 2^{N/2}\} \to \mathbb{C}$. This lends itself naturally to a two-dimensional plot where the two axes represent the two distinct regions of our system. In particular, what's plotted in Figure 4.10 is $|\Psi(i,j)|$.

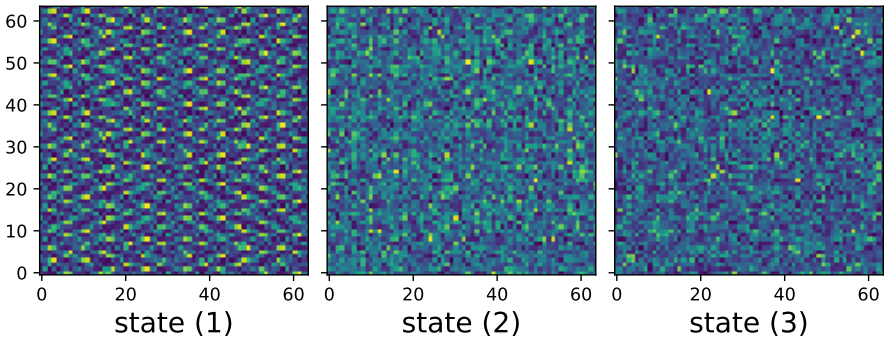


Figure 4.10: This figure, borrowed from [34], is the inspiration for this proof of concept. The three states all have $N = 12$. State (1) was evolved through a Clifford circuit, State (2) through a mostly-Clifford circuit with 12 `T` gates, and State (3) through a random universal circuit.

It's apparent that states generated by Clifford circuits have a nice periodic behaviour in this representation, and that this periodicity is quickly lost as circuits transition to being random unitary. Current research seeks to train a neural network to determine what type of circuit was used to generate a state based on the state's representation as an image like Figure 4.10.

The relationship between projective and unitary t-designs suggests that examining this problem from the perspective of circuits could be fruitful. This sort of analysis is made possible by the efficient circuit techniques we have implemented over the course of this thesis. Indeed, learning about a circuit by looking at the states it generates throws away significant information about the circuit itself. Multiplying an N-qubit operator by the 2^N different basis states yields 2^N independent evolved states, so perhaps each of these slices of the operator could be used in the same way as an individual state above.

Additionally, there is another way to visualize the information in circuits, somewhat more in line with the technique in Equation 4.10. Just like a state $|\psi\rangle \in \mathcal{H}$ can be represented as a sum of basis states $|\psi_i\rangle$:

$$|\psi\rangle = \sum_i \Psi(i) \, |\psi_i\rangle$$

we can also represent an operator $\hat{U}$ as a sum of outer products:

$$\hat{U} = \sum_{i,j} U(i,j) \, |\psi_i\rangle \langle\psi_j|$$

This makes sense since such a sum describes where each basis vector in $\mathcal{H}$ is mapped to by $\hat{U}$.

Consequently, we can write $\hat{U}$ in a bipartitioned system as follows:

$$\hat{U} = \sum_{i,j,k,l} U(i,j,k,l)(|\alpha_i\rangle \otimes |\beta_j\rangle)(\langle\alpha_k| \otimes \langle\beta_l|)$$

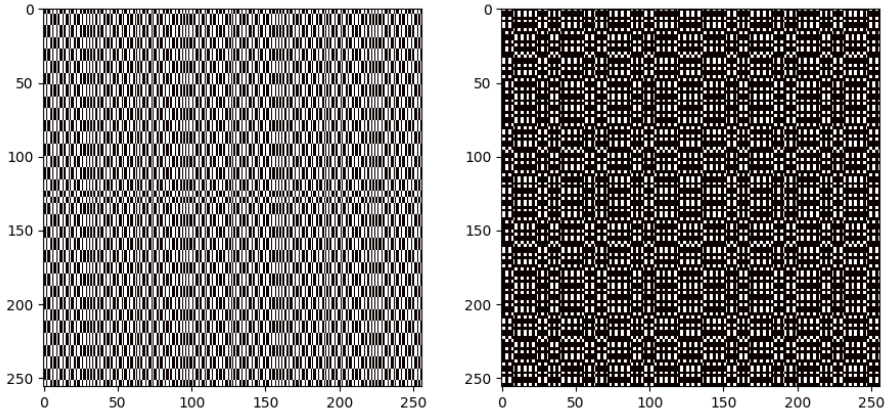
Then, if we define some $f_A, f_B : \{1,\ldots,2^{N/2}\}\times\{1,\ldots,2^{N/2}\} \to \{1,\ldots,2^N\}$, we can describe $\hat{U}$ as a two-dimensional function where each dimension corresponds to a partitioned region of our system:

$$\hat{U} = \sum_{i,j,k,l} U(f_A, f_B)(|\alpha_i\rangle \otimes |\beta_j\rangle\,(\langle\alpha_k| \otimes \langle\beta_l|) \tag{4.11}$$

where f_A stands for $f_A(i,k)$ and f_B stands for $f_B(j,l)$, so that both combine information from only one region of the system. An appropriate definition of these functions is $f_A(i,k) = f_B(i,k) = 2^{N/2}i + k$. This is the definition we use hereafter.

4.3.2 Initial Results

Figure 4.11: The colour map used to convert circuits to pictures. Larger magnitudes are converted to lighter colours and lower magnitudes to darker colours.



(a) **(b)**

Figure 4.12: Two examples of the black and white class of Clifford circuit images.

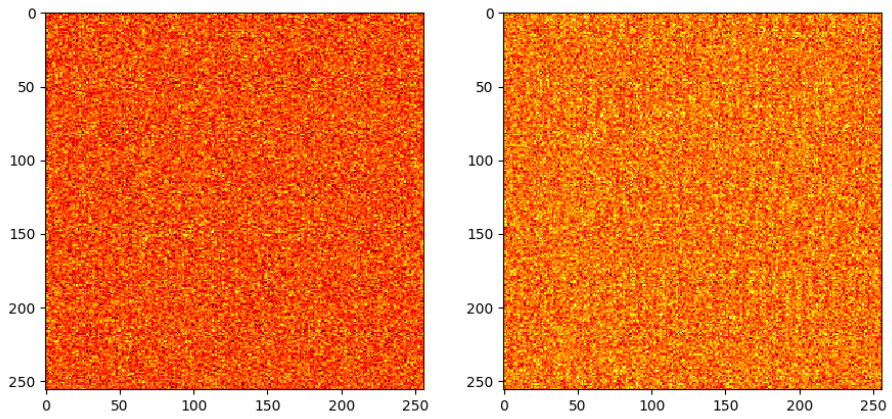


Figure 4.13: Two examples of the light orange and yellow class of Clifford circuit images.

Although these results are not the focus of this thesis, we've generated several circuit images as proofs of concept. So far, we have found that Clifford circuits and random universal circuits are visually distinguishable, but it remains to be seen how many `T` gates are required so that the resulting circuit image looks random unitary. Note that all of these images are generated by assigning a colour as in Figure 4.11 so that operator elements with larger magnitudes are mapped to lighter colours and elements with smaller magnitudes are mapped to darker colours.

Visually, Clifford circuits seem to be divided into two classes: primarily black/white images (see Figure 4.12), and mostly orange/yellow images (see Figure 4.13). In contrast, there only seems to be one class of random universal circuits: all of them seem to appear mainly black and red (see Figure 4.14).

Finally, Figure 4.15 depicts the evolution of an initially Clifford circuit as `T` gates are embedded in the middle of the circuit. Certainly the appearance of the circuit changes as `T` gates are added, but it's not yet clear whether this illustrates a meaningful pattern and if it is representative of other mostly-Clifford circuits.

An important caveat about these initial results is that colours are assigned from Figure 4.11 scaled according to the minimum and maximum values in the matrix. Thus, it's possible that some of the major colour differences between the Clifford images and random unitary images are due to this scaling. In fact, this seems to be the case, at least to a certain extent. We measured the maximum magnitudes in 10 Clifford circuits and 10 random universal circuits

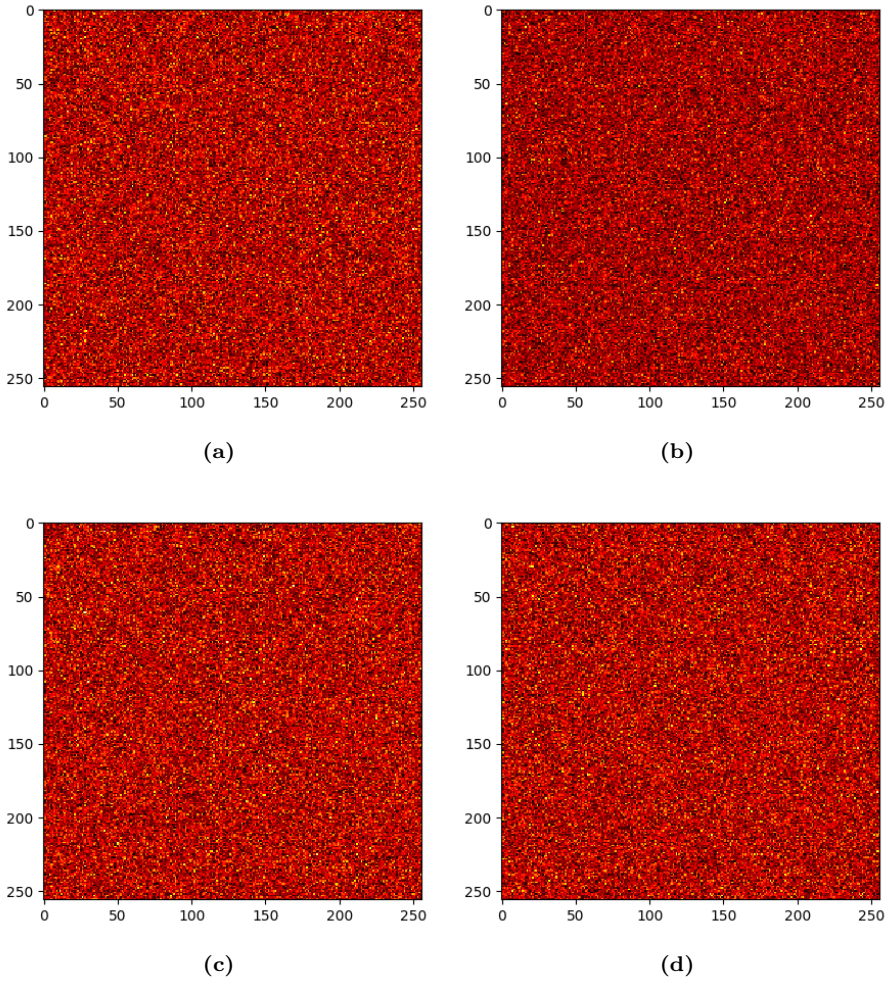
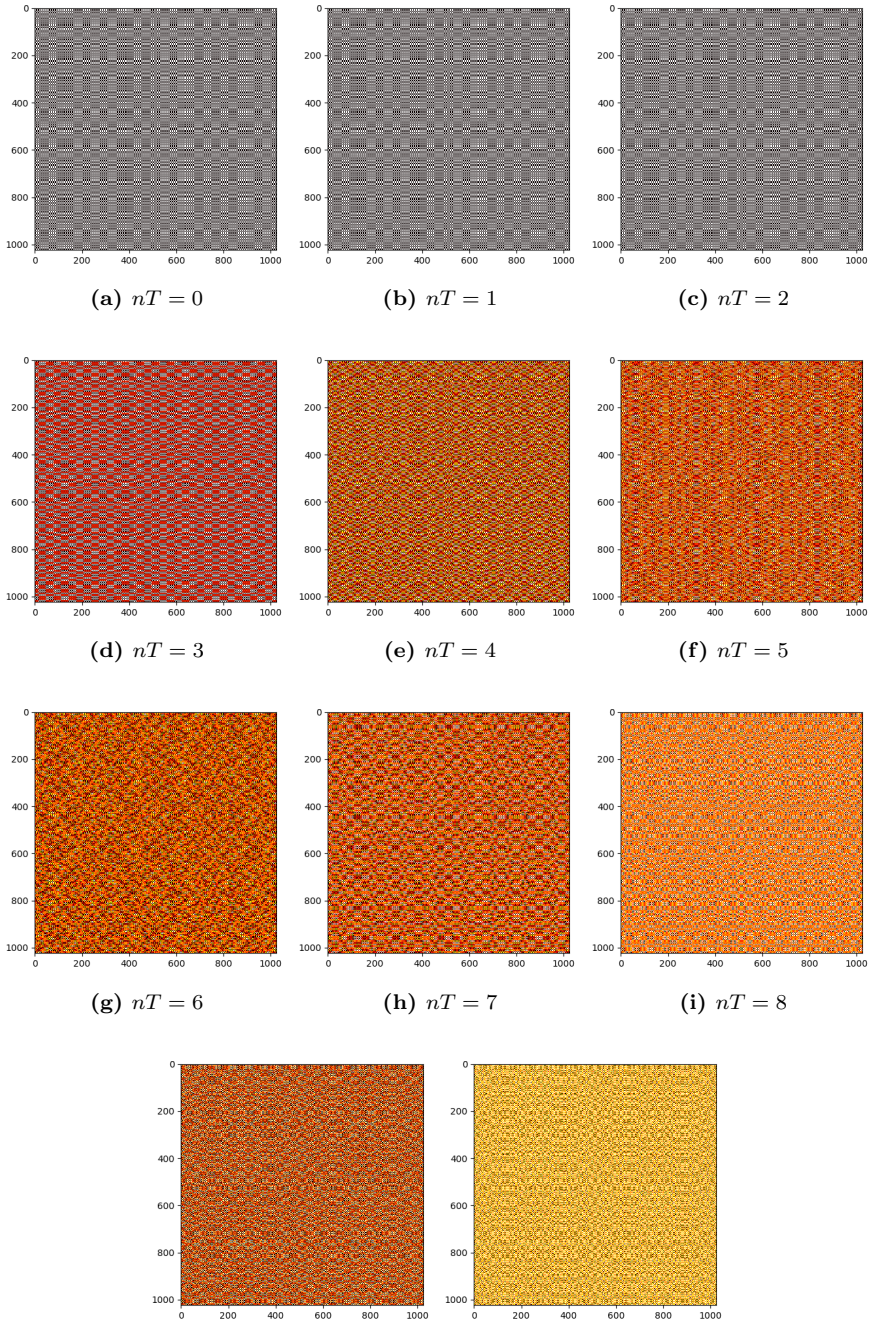


Figure 4.14: Four examples of the darker red appearance of random universal circuit images.

with $N = 8$ and $b = 50$ and found that $max = 0.092 \pm 0.026$ for the Clifford circuits and $max = 0.21 \pm 0.012$ for the random universal circuits. Nevertheless, the fact that the two circuits seem to have distinguishable images is promising – especially if this can be refined with appropriate colour normalization and such – and suggests that machine learning on these circuit images may be possible. Either way, this proof of concept demonstrates that more study is required to understand and make use of these images.



(a) $nT = 0$ **(b)** $nT = 1$ **(c)** $nT = 2$

(d) $nT = 3$ **(e)** $nT = 4$ **(f)** $nT = 5$

(g) $nT = 6$ **(h)** $nT = 7$ **(i)** $nT = 8$

(j) $nT = 9$ **(k)** $nT = 10$

Figure 4.15: $N = 10, b = 50$ mostly-Clifford circuits with n_T `T` gates embedded in the middle. Each successive image adds a new `T` gate to the same circuit.

Chapter 5

Conclusion

5.1 A Convincing Answer: Quantum Homeopathy

At the core of this thesis is the question of how many non-Clifford gates are required for a collection of circuits composed mostly of Clifford gates to constitute a unitary 4-design. An even stronger question is how many non-Clifford gates are required to produce a t-design for arbitrary t.[1] Both questions were resolved by an unaffiliated research group a couple of months before the completion of this thesis [16]. Although it's unfortunate that we could not demonstrate this result with our computational methods, the results achieved in [16] are both impressive and satisfying. Their methods are entirely analytical – leveraging powerful representation-theoretic machinery – and go beyond the question this thesis sought to answer.

In particular, it was found that for a particular architecture of Clifford and non-Clifford gates depicted in Figure 5.1, the number of non-Clifford gates required to transition in an N-qubit system is proportional only to t, not to N. Since a non-constant polynomial-length (in N) sequence of gates from G_C is required to produce an arbitrary Clifford gate, it follows that in the thermodynamic limit the density of non-Clifford gates required to produce a t-design drops to zero. The authors of [16] dub this phenomenon *quantum homeopathy* in analogy with regular homeopathy, "which posits that medication can be effective even if the active ingredient is arbitrarily diluted" – here, the active ingredient is `T` gates.

In Section 1.2.3, we introduced the compelling conjecture made in [34] that the transition from 3-design to 4-design could match that made from Poisson to

[1] Of course, none are required for $t \leq 3$.

Wigner-Dyson ESS. Since the density of `T` gates in either case may be made to drop to zero, the notion of quantum homeopathy settles this conjecture in the affirmative.

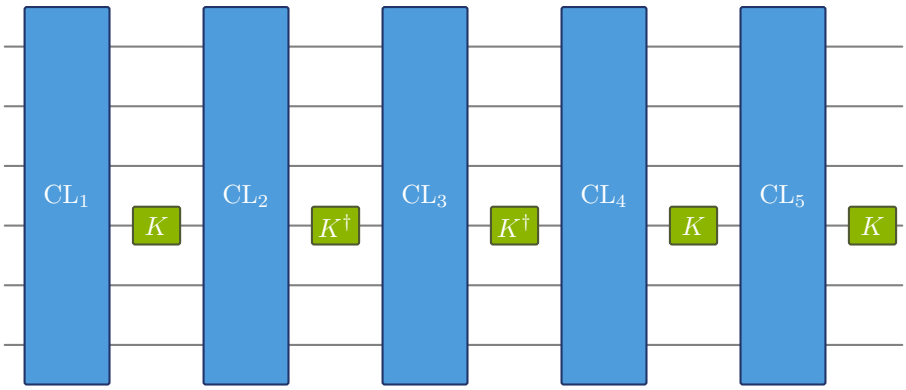


Figure 5.1: A circuit architecture where instances of some non-Clifford gate K (or $K^\dagger$) are interspersed between layers of Clifford gates. Such an architecture can be used to construct unitary t-designs from mostly-Clifford circuits for any t. When this is done, the number of non-Clifford gates depends only on t. This figure is borrowed from [16].

Figure 5.1 depicts the circuit architecture used in [16]. Let K be a non-Clifford unitary gate acting on one qubit (e.g., a `T` gate). In Figure 5.1, random Clifford gates are interleaved with either K or $K^\dagger$. A circuit interleaved in this way has *depth* k if it is composed of k layers of Clifford gates and k layers of K or $K^\dagger$ gates. It was found that a circuit depth of $O(t^4 \log^2(1/\epsilon))$ is required for such circuits to compose an ϵ-*approximate* t-*design*. Although this differs in subtle ways from the t-designs we introduced in this thesis, this result still implies that circuits of depth $O(t^4)$ can compose a set matching the behaviour of a t-design to arbitrary precision.

If we were generating a circuit like that in Figure 5.1, we would use bN^2 circuits selected from $G_C = \{\texttt{H}, \texttt{S}, \texttt{CNOT}\}$ – generating a random Clifford gate – for each blue layer. Thus, the circuit would have length $O(N^2)$ at least. Since the number of non-Clifford gates required for such an architecture to produce a unitary t-design depends only on t, for large N the density of non-Clifford gates required drops to zero, as noted above.

5.2 Our Progress

Although we were not able to answer our central question – and another group did in a deeply convincing way – we still made meaningful progress over the course of this thesis.

We built a robust and efficient engine for generating and manipulating random quantum circuits. As shown in Section 4.2, our most advanced implementation performs multiple orders of magnitude better than some of our earlier implementations, which themselves are much more efficient than the naïve approach. Thus far, the largest systems we have been able to test have 13 qubits, but we are hopeful that we will be able to push this further.

We also demonstrated the intractability of our original frame potential approach. Furthermore, we suggested and began to test a new, related approach that could yield useful results.

Lastly, we applied our random circuit engine to a new task: we extended a method of visualizing states to apply to operators as well, with the eventual aim of using machine learning to predict circuit architectures from these images. We generated initial proof-of-concept data, demonstrating the broader utility of our random circuit engine.

5.3 Future work

Although it was established in [16] that only $O(t^4)$ non-Clifford gates are required to build unitary t-designs out of mostly-Clifford circuits, this only describes the asymptotic growth of this number – it doesn't make clear how many such gates are required. Determining this would be of use in assessing whether the number of non-Clifford gates required is practically feasible. Furthermore, this result was shown for a specific architecture of circuit. A successful computational approach to assessing unitary t-designs would help us explore a wider range of potential architectures.

Therefore, our original goal of computationally determining when mostly-Clifford circuits constitute unitary 4-designs is still a useful one. Our next step is to further explore our new method, gathering more off-diagonal frame potential data. Additionally, with this and other projects in mind, we're interested in further optimizing our computations. In this vein, we can explore the possibility of using a super-computing cluster or finding new ways to increase the efficiency of our code. Additionally, there are some existing quantum information packages that we haven't yet explored very deeply (e.g., [14] and [22]). Trying to determine whether these have applicable tools or methods for us could be productive.

Finally, as demonstrated by our work in Section 4.3, the machinery we have built over the course of this thesis is also applicable to other interesting quantum information problems. Thus, we intend to further develop it and apply it to new problems in the future.

Bibliography

[1] S. J. Blundell and K. M. Blundell. *Concepts in Thermal Physics*. English. 2 edition. Oxford ; New York: Oxford University Press, Nov. 2009. ISBN: 978-0-19-956210-7.

[2] F. G. S. L. Brandao, A. W. Harrow, and M. Horodecki. "Local random quantum circuits are approximate polynomial-designs". In: *Communications in Mathematical Physics* 346.2 (Sept. 2016). arXiv: 1208.0692, pp. 397–434. ISSN: 0010-3616, 1432-0916. DOI: 10.1007/s00220-016-2706-8.

[3] S. Bravyi and A. Kitaev. "Universal Quantum Computation with ideal Clifford gates and noisy ancillas". en. In: (Mar. 2004). DOI: 10.1103/PhysRevA.71.022316.

[4] C. Chamon, A. Hamma, and E. R. Mucciolo. "Emergent irreversibility and entanglement spectrum statistics". In: *Physical Review Letters* 112.24 (June 2014). arXiv: 1310.2702, p. 240501. ISSN: 0031-9007, 1079-7114. DOI: 10.1103/PhysRevLett.112.240501.

[5] R. Chao and B. W. Reichardt. "Fault-tolerant quantum computation with few qubits". en. In: *npj Quantum Information* 4.1 (Sept. 2018). Number: 1 Publisher: Nature Publishing Group, pp. 1–8. ISSN: 2056-6387. DOI: 10.1038/s41534-018-0085-z.

[6] J. B. Conway. *A Course in Functional Analysis*. English. 2nd edition. New York: Springer, Jan. 1994. ISBN: 978-3-540-96042-3.

[7] T. H. Cormen et al. *Introduction to Algorithms, 3rd Edition*. English. 3rd edition. Cambridge, Mass: The MIT Press, July 2009. ISBN: 978-0-262-03384-8.

[8] C. Dankert et al. "Exact and Approximate Unitary 2-Designs: Constructions and Applications". In: *Physical Review A* 80.1 (July 2009). arXiv: quant-ph/0606161, p. 012304. ISSN: 1050-2947, 1094-1622. DOI: 10.1103/PhysRevA.80.012304.

[9] P. Delsarte, J. M. Goethals, and J. J. Seidel. "Spherical codes and designs". en. In: *Geometriae Dedicata* 6.3 (Sept. 1977), pp. 363–388. ISSN: 1572-9168. DOI: 10.1007/BF03187604.

[10] D. P. DiVincenzo and IBM. "The Physical Implementation of Quantum Computation". In: *arXiv:quant-ph/0002077* (Apr. 2000). arXiv: quant-ph/0002077 version: 3. DOI: 10.1002/1521-3978(200009)48:9/11<771::AID-PROP771>3.0.CO;2-E.

[11] R. P. Feynman. "Simulating physics with computers". en. In: *International Journal of Theoretical Physics* 21.6 (June 1982), pp. 467–488. ISSN: 1572-9575. DOI: 10.1007/BF02650179.

[12] A. Glazyrin and J. Park. "Repeated minimizers of p-frame energies". In: *arXiv:1901.06096 [math]* (Aug. 2019). arXiv: 1901.06096.

[13] D. Gottesman. "The Heisenberg Representation of Quantum Computers". In: *arXiv:quant-ph/9807006* (July 1998). arXiv: quant-ph/9807006.

[14] J. Gray. *jcmgray/quimb*. original-date: 2015-12-09T14:02:41Z. Apr. 2020. URL: https://github.com/jcmgray/quimb (visited on 04/30/2020).

[15] D. Gross, K. Audenaert, and J. Eisert. "Evenly distributed unitaries: on the structure of unitary designs". In: *Journal of Mathematical Physics* 48.5 (May 2007). arXiv: quant-ph/0611002, p. 052104. ISSN: 0022-2488, 1089-7658. DOI: 10.1063/1.2716992.

[16] J. Haferkamp et al. "Quantum homeopathy works: Efficient unitary designs with a system-size independent number of non-Clifford gates". en. In: *arXiv:2002.09524 [math-ph, physics:quant-ph]* (Feb. 2020). arXiv: 2002.09524.

[17] T. D. Ladd et al. "Quantum Computing". In: *Nature* 464.7285 (Mar. 2010). arXiv: 1009.2267, pp. 45–53. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/nature08812.

[18] T. A. Moore. *A General Relativity Workbook*. English. Workbook edition. Mill Valley, California: Univ Science Books, Sept. 2012. ISBN: 978-1-891389-82-5.

[19] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. English. 10th Anniversary ed. edition. Cambridge ; New York: Cambridge University Press, Jan. 2011. ISBN: 978-1-107-00217-3.

[20] M. Ozols. *Clifford Group*. July 2008. URL: http://home.lu.lv/~sd20008/papers/essays/Clifford%20group%20[paper].pdf.

[21] A. Paler and S. J. Devitt. "An introduction into fault-tolerant quantum computing". In: *Proceedings of the 52nd Annual Design Automation Conference*. DAC '15. San Francisco, California: Association for Computing Machinery, June 2015, pp. 1–6. ISBN: 978-1-4503-3520-1. DOI: 10.1145/2744769.2747911. URL: https://doi.org/10.1145/2744769.2747911 (visited on 04/24/2020).

[22] *qutip/qutip*. original-date: 2012-10-09T06:20:46Z. Apr. 2020. URL: https://github.com/qutip/qutip (visited on 04/30/2020).

[23] J. M. Renes et al. "Symmetric Informationally Complete Quantum Measurements". In: *Journal of Mathematical Physics* 45.6 (June 2004). arXiv: quant-ph/0310075, pp. 2171–2180. ISSN: 0022-2488, 1089-7658. DOI: 10.1063/1.1737053.

[24] W. Rudin. *Real and Complex Analysis*. English. 3 edition. New York: McGraw-Hill Education, May 1986. ISBN: 978-0-07-054234-1.

[25] D. Shaffer et al. "Irreversibility and Entanglement Spectrum Statistics in Quantum Circuits". In: *Journal of Statistical Mechanics: Theory and Experiment* 2014.12 (Dec. 2014). arXiv: 1407.4419, P12007. ISSN: 1742-5468. DOI: 10.1088/1742-5468/2014/12/P12007.

[26] S. Shahriari. *Algebra in Action: A Course in Groups, Rings, and Fields*. English. 1st edition. Providence, Rhode Island: American Mathematical Society, Aug. 2017. ISBN: 978-1-4704-2849-5.

[27] H. M. Sheffer. "A set of five independent postulates for Boolean algebras, with application to logical constants". en. In: *Transactions of the American Mathematical Society* 14.4 (1913), pp. 481–488. ISSN: 0002-9947, 1088-6850. DOI: 10.1090/S0002-9947-1913-1500960-1.

[28] V. M. Sidelnikov. "Upper bounds on the cardinality of a binary code with a given minimum distance". en. In: *Information and Control* 28.4 (Aug. 1975), pp. 292–303. ISSN: 0019-9958. DOI: 10.1016/S0019-9958(75)90310-1.

[29] D. Stinson. *Combinatorial Designs: Constructions and Analysis*. en. New York: Springer-Verlag, 2004. ISBN: 978-0-387-95487-5. DOI: 10.1007/b97564.

[30] J. Tolar. "On Clifford groups in quantum computing". In: *Journal of Physics: Conference Series* 1071 (Aug. 2018). arXiv: 1810.10259, p. 012022. ISSN: 1742-6588, 1742-6596. DOI: 10.1088/1742-6596/1071/1/012022.

[31] J. S. Townsend. *A Modern Approach to Quantum Mechanics*. English. 2nd Revised ed. edition. Sausalito, Calif: Univ Science Books, Feb. 2012. ISBN: 978-1-891389-78-8.

[32] L. Welch. "Lower bounds on the maximum cross correlation of signals (Corresp.)" In: *IEEE Transactions on Information Theory* 20.3 (May 1974), pp. 397–399. ISSN: 1557-9654. DOI: 10.1109/TIT.1974.1055219.

[33] Z. Xu and Z. Xu. "The minimizers of the p-frame potential". In: *arXiv:1907.10861 [math]* (Aug. 2019). arXiv: 1907.10861.

[34] S. Zhou et al. "Single T gate in a Clifford circuit drives transition to universal entanglement spectrum statistics". In: *arXiv:1906.01079 [cond-mat, physics:quant-ph]* (Apr. 2020). arXiv: 1906.01079.

[35] Y. Zhou and X. Chen. "Spherical t_ϵ-Designs for Approximations on the Sphere". In: *arXiv:1502.03562 [math]* (Feb. 2015). arXiv: 1502.03562.

[36] H. Zhu et al. "The Clifford group fails gracefully to be a unitary 4-design". In: *arXiv:1609.08172 [quant-ph]* (Sept. 2016). arXiv: 1609.08172.