Research in Industrial Projects for Students

Sponsor

NASA Ames Research Center

Final Report

Classical Simulation of Non-Clifford Noise Channels in Stabilizer Formalism

Student Members

Jad Soucar (Project Manager), *University of California, Los Angeles*, jadsoucar@g.ucla.edu

Brandin Farris, *Oregon State Univeristy*

Daley McMahon, *University of Pennsylvania*

Peter Ye, *Yale University*

Academic Mentor

Zhuoyang (John) Ye, yezhuoyang98@g.ucla.edu

Sponsoring Mentors

Namit Anand, Namit.Anand@us.kbr.com

Jeffrey Marshall, jmarshall@usra.edu

Date: August 15, 2024

This project was jointly supported by NASA Ames Research Center and NSF Grant DMS 1925919.

Abstract

We introduce the basics of quantum computing and simulation of quantum systems on classical computers. We then discuss noise in quantum systems and how it is classically modelled, along with the difficulties of simulating quantum noise on classical computers. Our primary question is how to efficiently simulate quantum noise by leveraging existing techniques based upon the Gottesman-Knill theorem, which provides efficient simulation of circuits containing only Clifford gates. We follow this by describing our progress thus far in exploring three primary approaches. First, exploring methods to decompose a noise operator into a sum of cliffords via mixed integer linear programming and using these decompositions to classically simulate noisy quantum circuits within the stabilizer formalism first proposed by Aaronson and Gottesman (2004). We find that the a Clifford decomposition is not guaranteed to have low rank decompositions and that at best the runtime of simulating noise using Clifford decompositions would be $O(2^k)$, where k is the number of Kraus operators applied. Second, we explore the process of dilating our space to convert our noise operators into unitaries in a larger space. Specifically we propose two unitary dilation based algorithms for simulating noise; Sz.-Nagy and Stinespring's dilation algorithms. The two algorithms yield respective run-times of $O(\frac{1}{\delta\Delta^2} s n^3 \eta^{-2} 1.17^t)$ and $O(n^2 \prod_{i=1}^{k} |\xi_i| + n^3 \sum_{i=1}^{k} |\xi_i|^3)$. Third we propose a theoretical framework for generalizing the T-Gadget approach developed by Bravyi and Gosset (2016). Through numerical simulations we show that the generalized framework can be used to produce noise simulation algorithms with efficient runtime and space complexity.

Contents

Chapter 1

Introduction

In the early 1980s, Feynman and Manin among others proposed the concept of a so-called quantum computer that would simulate quantum systems more effectively than classical computers, following the development of quantum computing theory [1; 13, ch. 4]. Since then, quantum computing has been developed, studied, and applied to problems beyond simulating quantum physics. Today, new quantum algorithms like Shor's algorithm theoretically achieve exponential speed up in prime factorization and have the potential to be used in fields such as cryptography, where quantum algorithms can potentially break widely used RSA encryption schemes [13, ch. 4.1]; and in optimization, where quantum annealing can solve optimal trading trajectory problems [**?**].

Unfortunately, real-world quantum computers face significant challenges in producing consistent results due to errors and decoherence brought on by quantum noise; when cutting-edge applications may require circuits comprised of billions of gates, the error in implementing a single gate in order to perform reliable computations must be orders of magnitudes smaller than what is currently accomplishable on state-of-the-art quantum processors. In order to solve this problems, researchers have been developing and testing quantum error correction codes to increase the fault-tolerance of quantum devices, at the expense of increased time or memory requirements [7]. Efficient simulation of noisy circuits on classical computers would therefore allow those without access to quantum hardware to debug or understand the boundaries of potential quantum speeds ups of quantum algorithms [**?**].

However, the classical simulation of quantum systems remains a difficult problem due to exponential computational complexity requirements. Namely, since an n-qubit quantum system is represented by $(\mathbb{C}^2)^{\otimes n}$, the amount of memory required to naïvely store a state vector scales as $\mathcal{O}(2^n)$; moreover, to evolve the system under a unitary transformation $U \in \mathcal{H}((\mathbb{C}^2)^{\otimes n})$ requires $\mathcal{O}(2^{3n})$ time. Functionally, classical simulations have been limited to systems with fewer than approximately 50 qubits [16], where for an example, the Google Quantum AI team experimentally performed a computational task on 53 qubits with quantum hardware, which was later then estimated to take approximately 10,000 years[1] to simulate using classical simulation [15]. One solution to this computational cost was introduced by Gottesman and Knill, where he introduced the stabilizer formalism, a method to simulate a subset of quantum circuits (namely circuits composed of Clifford gates) in polynomial time. This was later improved to universality via Bravyi's method which involves including T gates to the set of Cliffords [5]. Bravyi shows that any unitary can be simulated by decomposing it into C, H, P, and T gates and T gates can be simulated within the stabilizer formalism by adding a mild exponent to the computational cost.

[1] This figure is said to be exaggerated, by [15] and others.

However, some noise operators are not unitary and as a result, simulating this quantum noise can be computationally expensive. It is at this hurdle that our research is situated: In collaboration with NASA Ames Research Center (ARC) we aim to exploit the algebraic structure of quantum noise to develop a computationally cheaper representation of noise to be deployed in classical quantum simulations.

In this report, we will first introduce the necessary background to simulating quantum circuits within the stabilizer formalism, and then we will talk about the different methods used for simulating non-unitary noise channels. The first two methods we propose are dilation methods, which allow us to lift our noise operators to a higher Hilbert space such that the lifted operator is unitary, and then we employ Bravyi's method to apply the unitaries within the stabilizer formalism . The second method we propose is an extension of Bravyi's work which generalizes his T gadget to diagonal and off diagonal operators, which allow us to implement many noise channels, as many are composed of diagonal and off-diagonal operators [6]. We end with methods that held promise but were unfinished, namely decomposing our noise operators into a sum of Cliffords, and applying each Clifford to a copy of our circuit.

Chapter 2

Background

2.1 Gottesman–Knill theorem

Theorem (Gottesman–Knill theorem)**.** *A quantum computation performed with Clifford operations and measurements of observables in the computational basis may be simulated in polynomial time on a classical computer.*

$$C = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}, \quad H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad P = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}$$

The Gottesman-Knill theorem tells us we can simulate circuits composed of Clifford gates in polynomial time. The Clifford gate set is a finite group defined by the generating set of C, H, and P gates (mod $U(1)$). The Clifford group is the normalizer of the Pauli group, meaning that conjugating a Pauli operator by a Clifford unitary results in another Pauli operator. Specifically for $U(n)$ being the n-qubit unitaries and $P(n)$ being the n-qubit Pauli group,

$$C(n) \equiv \{c \in U(n) | \forall p \in P(n), cpc^\dagger \in P(n)\}$$

The process that allows us to simulate these Clifford circuits in polynomial time is based on the stabilizer formalism. Note that the Gottesman-Knill theorem and the stabilizer formalism supports cheap implementation of quantum entanglement, a property that is hard for other frameworks to simulate [21].

2.1.1 Stabilizer formalism

A stabilizer state is any state $|s\rangle$ such that $|s\rangle = C|0\rangle^{\otimes n}$ for some n-qubit Clifford C. All stabilizer states are in *unique* correspondance to a stabilizing group, $\text{Stab}(|s\rangle) = \{P \in P(n) \mid P|s\rangle = |s\rangle$. The stabilizing group is a finite group of order n where $|s\rangle$ is an n-qubit stabilizer state. Notice that when we apply a Clifford operator C to a stabilizer state $|s\rangle$ with a stabilizer group $\text{Stab}(|s\rangle) = \langle g_1, g_2, \ldots g_n \rangle$:

$$C|s\rangle = Cg_i|s\rangle = Cg_iC^\dagger C|s\rangle$$

Then for the generators g_i of $\text{Stab}(|s\rangle)$, $Cg_iC^\dagger$ are the generators of the stabilizing group of $C|s\rangle$. Because these generating sets are in one to one correspondance with a stabilizer

state, we can track the generators of the stabilizing group across the circuit instead of the state vector itself. This initially seems computationally expensive as the process involves matrix multiplication, but there is a computationally efficient way to update our stabilizer group, and that is via the tableau method.

2.1.2 Stabilizer Tableau

The stabilizer tableau is a compact representation of the generating set of the stabilizer group that can be efficiently updated using bitwise operations. The matrix can be represented as a binary matrix:

$$\left[\begin{array}{c|cccc|cccc} s_1 & x_{1,1} & x_{1,2} & x_{1,3} & x_{1,4} & z_{1,1} & z_{1,2} & z_{1,3} & z_{1,4} \\ s_2 & x_{2,1} & x_{2,2} & x_{2,3} & x_{2,4} & z_{2,1} & z_{2,2} & z_{2,3} & z_{2,4} \\ s_3 & x_{3,1} & x_{3,2} & x_{3,3} & x_{3,4} & z_{3,1} & z_{3,2} & z_{3,3} & z_{3,4} \\ s_4 & x_{4,1} & x_{4,2} & x_{4,3} & x_{4,4} & z_{4,1} & z_{4,2} & z_{4,3} & z_{4,4} \end{array}\right]$$

where $\mathbf{s}$ represents the sign(1 for minus and 0 for positive), and $\mathbf{x}$ and $\mathbf{z}$ are binary matrices representing the Pauli X and Z components, respectively. Each row of the tableau corresponds to a generator of the stabilizer group. For example, the first stabilizer is equal to

$$S_i = x_{1,1}z_{1,1} \otimes x_{1,2}z_{1,2} \otimes x_{1,3}z_{1,3} \otimes x_{1,4}z_{1,4},$$

$$x_{i,j}z_{i,j} : 00 \to I, 01 \to Z, 10 \to X, 11 \to Y$$

The *ith* column of the x matrix $\mathbf{x}_i$ and z matrix $\mathbf{z}_i$ together correspond to the *ith* qubit. Notice, however, that this does not imply that the stabilizers of the *ith* column are necessarily the stabilizers of the *ith* qubit in the system, as the stabilizer formalism supports entanglement, so the qubits may be inseparable. As an example, consider the following tableau:

$$\left[\begin{array}{c|cc|cc} 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 \end{array}\right]$$

The generating set for this stabilizing group is $\text{Stab}(|\phi^+\rangle) = \{ZZ, XX\}$. Notice the columns of the first and second qubits do not correspond to the stabilizing group of the individual qubits, as they are an entangled state $|\phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$. Recall that the set $\{C, H, P\}$ are a generating set for all Cliffords, thus when creating and applying Clifford circuits, we apply C, H, and P gates to our tableau. There are rules for how to update the tableau when applying these gates which are outlined in [2]. Their improved tableau also allows for random measurements in $O(n^2)$ time. The Stabilizer Tableau allows an efficient way to store and update the stabilizing group for any n-qubit stabilizer state when applying Cliffords to the system. Recall, however, that the Clifford gate set is not universal in that we cannot apply all unitaries via a composition of C, H, and P gates. As it turns out, including one more gate, canonically the T gate, makes this generating set universal.

2.2 Noise Channels

In this discrete setting, noise channels acting on our system are completely positive, trace preserving maps mapping density matrices to density matrices, $\phi : \rho \mapsto \rho'$. This mapping is called operator sum representation.

Definition. *Operator-sum representation. A map $\Phi : \rho \mapsto \rho'$ has a Kraus operator-sum representation (OSR) [i.e., $\Phi(\rho) = \sum_\alpha K_\alpha \rho K_\alpha^\dagger$ with $\sum_\alpha K_\alpha^\dagger K_\alpha = I$] if and only if it is linear and completely positive trace preserving (CPTP).*

It is important to note here that the Kraus operators K_i may not be Clifford. In fact, many of them are not even unitary. The noise channels we're interested in simulating are noise channels where some or all of the operators are non-Clifford, as such channels are more difficult to simulate. Consider two noise channels, in terms of their constituent Kraus operators of some operator-sum representations: the phase dampening map and the amplitude dampening map. The phase damping map is defined as $\Phi(\rho') = p\rho + (1-p)Z\rho Z$. Then the Kraus operators are $K_0 = \sqrt{p}I$ and $K_1 = \sqrt{1-p}Z$. This map can be understood as:

$$\rho \mapsto \rho' = \begin{cases} \rho & \text{with probability } p \\ Z\rho Z & \text{with probability } 1-p \end{cases}$$

The amplitude damping map is defined as $\Phi(\rho) = K_0 \rho K_0^\dagger + K_1 \rho K_1^\dagger$ where the Kraus operators are $K_0 = |0\rangle\langle 0| + \sqrt{1-\gamma}|1\rangle\langle 1|$ and $K_1 = \sqrt{\gamma}|0\rangle\langle 1|$. This map can be understood as:

$$\begin{aligned} |0\rangle &\to |0\rangle \quad \text{with probability } 1 \\ |1\rangle &\to |0\rangle \quad \text{with probability } p \end{aligned}$$

or if the density matrix is written as $\begin{pmatrix} \rho_{00} & \rho_{01} \\ \rho_{10}^* & 1-\rho_{00} \end{pmatrix}$ then ρ' is given as

$$\rho \mapsto \rho' = \begin{cases} \begin{pmatrix} \rho_{00} & \sqrt{1-\gamma}\rho_{01} \\ \sqrt{1-\gamma}\rho_{01}^* & (1-\gamma)(1-\rho_{00}) \end{pmatrix} & \text{with probability } 1-\gamma \\ \gamma(1-\rho_{00})|0\rangle\langle 0| & \text{with probability } \gamma \end{cases}$$

2.2.1 Quantum trajectories method

Obviously, working with density matrices is much more expensive than working with pure states, so we wish to avoid implementing OSR. Instead of working directly with operator sum representation, we can apply a method that allows us to work only with pure states, and still apply a full channel. Given a set of Kraus channels $\xi_i = \{K_{i,j}\}$ and a pure state $|\psi\rangle$, one can sample the output of applying the channels to a ket by sampling and applying a single Kraus to the ket per channel by the given probability distribution:

$$p_i = \langle \psi K_{i,j}^\dagger | K_{i,j} \psi \rangle$$

More details of the algorithm can be found here[**?**]. This means that the probability of choosing a given Kraus is dependent on the current state of the system. We also note that the Kraus operators need not be Clifford or unitary, meaning they do not fit within the stabilizer formalism and cannot be immediately decomposed into CHPT gates to allow a gadgetized implementation. Thus we introduce various methods for applying these non-unitary operators K to our stabilizer state $|s\rangle$.

Theorem 2.1 (Solovay-Kitaev algorithm [11])**.** *Any unitary matrix can be approximated through the Solovay-Kitaev algorithm within error ϵ, which has been optimized to use $O(\log(1/\epsilon)^{1.45})$ total gates.*

The Solovay-Kitaev algorithm provides a computational efficient way to decomposing unitary matrices into compositions of C,H,P and T gates. However, notice that the Clifford decomposition of the T gate has rank 2. Thus, in order to simulate t number of T-gates, we need 2^t tableus. [5] introduces the idea of a T-gadget, which can substitute the spot of each T-gate.

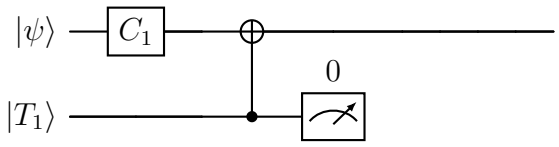


Figure 2.1: T-gadget. We perform a forced measurement on the ancillary qubit to the $|0\rangle$ state.

Rather than relying upon a Clifford decomposition of the T-gate, one can initialize an ancillary qubit in the $|T\rangle$ state and apply a series of Clifford operators and measurements on the extra qubit and current state to achieve the same result as applying a T-gate. Specifically, this $|T\rangle = \begin{pmatrix} 1 \\ e^{i\pi/4} \end{pmatrix}$. Although using T-gadgets removes the cost of applying the T-gate, there is a new cost in terms of storing these ancillary qubits held in the $|T\rangle$-state. Specifically, we must store $|T\rangle^{\otimes t}$. Notice that $|T\rangle$ actually is not a stabilizer state. Thus, the cost comes from the stabilizer decomposition of $|T\rangle^{\otimes t}$.

Notice that $|T\rangle = |0\rangle + e^{i\pi/4}|1\rangle$. Thus, an upper bound on $|T\rangle^{\otimes t} = (|0\rangle + e^{i\pi/4}|1\rangle)^{\otimes t}$ which has rank at most 2^t. However, because $|T\rangle$ is tensored with itself many times, there is reason to believe that one can compress the decomposition of the large tensor product into one with fewer terms. Bravyi and Gosset goes through some details showing that one can find a decomposition of rank $\frac{1.17^t}{\delta}$ which approximates $|H\rangle^{\otimes t}$ within an error bound of δ.

Notice that $|T\rangle = e^{i\pi/8} HP^\dagger |H\rangle$ for $|H\rangle = \begin{pmatrix} \cos(\pi/8) \\ \sin(\pi/8) \end{pmatrix}$. Thus, if one can represent $|H\rangle^{\otimes t}$ with a low rank decomposition, then they can also obtain an efficient decomposition of $|T\rangle^{\otimes t}$ by applying Clifford gates and a phase shift.

We work with $|H\rangle$ instead of $|T\rangle$ because $|H\rangle$ has nicer properties. Namely, that $|H\rangle = \frac{|0\rangle+|+\rangle}{2v}$ where $v := \cos(\pi/8)$. Thus, if we define $|\tilde{0}\rangle := |0\rangle$ and $|\tilde{1}\rangle := |+\rangle$, then we get the expression:

$$|H\rangle^{\otimes t} = (\frac{|0\rangle + |+\rangle}{2v})^{\otimes t} = \frac{1}{(2v)^t} \sum_{x \in \mathbb{F}_2^t} |\tilde{x_1}...\tilde{x_t}\rangle \tag{2.1}$$

Definition 2.1 ($Z(\mathcal{L})$)**.** *The normalization function Z maps subspaces $\mathcal{L}$ of $\mathbb{F}_2^t$ as follows:*

$$Z(\mathcal{L}) = \sum_{x \in \mathcal{L}} 2^{-|x|/2} \tag{2.2}$$

Where $|\cdot|$ is defined as the hamming weight of $\cdot$.

The approximations of $\mathcal{H}^{\otimes t}$ are based on choice of $\mathcal{L}$. For any given k-dimensional subspace $\mathcal{L}$, there is the corresponding approximation which has rank 2^k:

$$|\mathcal{L}\rangle = \frac{1}{\sqrt{2^k Z(\mathcal{L})}} \sum_{x \in \mathcal{L}} |\tilde{x_1}...\tilde{x_t}\rangle \tag{2.3}$$

The above decomposition has rank 2^k because there are a total of 2^k bitstrings in a k-dimensional subspace $\mathcal{L}$, and each bitstring corresponds to a unique stabilizer state.

Definition 2.2 (Approximation Error:)**.** *The error function δ is defined such that any given approximation $|\mathcal{L}\rangle$, the error is given by:*

$$\delta|\mathcal{L}\rangle = 1 - ||\langle H^{\otimes t}|\mathcal{L}\rangle||^2$$

Bravyi and Gosset provides several algebraic steps and shows that this error is precisely equal to [5]:

$$\delta(|\mathcal{L}\rangle) = 1 - \frac{2^k v^{2t}}{Z(\mathcal{L})} \tag{2.4}$$

Thus, to minimize the error, one must minimize $Z(\mathcal{L})$. Bravyi and Gosset's paper shows that if k is chosen to the positive number satisfying $4 \geq 2^k v^{2t}\delta \geq 2$, then one can find a subspace with error less than δ after sampling $O(\frac{1}{\delta})$ k-dimensional subspaces [5]. Moreover, the rank of this subspace will be 2^k which is approximatly $\frac{2*v^{-2t}}{\delta}$ which is in $O(\frac{1.17^t}{\delta})$.

Chapter 3

Clifford and stabilizer decompositions

In Bravyi et al.'s 2019 paper [4], the so-called sum-over-Clifford method is presented to simulate arbitrary unitary circuits by finding δ-approximate decompositions into stabilizer states, and is stated to be comparable in efficiency to gadgetization methods, but at the at the benefit of being more robust.

Proposition 3.1 (Sum-over-Cliffords method, [4, sec. 2.3.2])**.** *Given a unitary circuit U on an n-qubit system, we claim we can approximate within arbitrary error δ the output $U|0^n\rangle$ by a superposition ψ of $k \approx \mathcal{O}(\delta^{-2})$ stabilizer states:*

$$\||U|0^n\rangle - |\psi\rangle\| \leq \delta, \qquad |\psi\rangle := \sum_{\alpha=1}^{k} b_\alpha C_\alpha |0^n\rangle,$$

where C_α are Clifford operators, using some form of random sampling.

The choice of approximating with stabilizer decompositions is not an arbitrary one: the rank of a pure state's stabilizer decomposition is actually a measure of magic for pure states. In other words, stabilizer rank quantifies the difficulty[1] of simulating a circuit with our pure state as an outcome [10].

Definition 3.1 (Stabilizer rank, [6])**.** *The stabilizer rank $\chi(\psi)$ of a pure state ψ on an n-qubit system is the smallest integer for which ψ can be written as a superposition of $\chi(\psi)$ stabilizer states.*

However, as there are finitely many stabilizer states, almost every[2] state vector will attain the maximal stabilizer rank 2^n of the size of a basis for the system, which can be known via Sard's theorem. Consequently, this implies arbitrary unitary circuits are difficult to simulate. It is for this reason we introduce the concept of approximate stabilizer decompositions, which analogously quantifies the approximation of a arbitrary unitary circuit with a circuit that is easier to simulate.

Definition 3.2 (Approximate stabilizer rank, [4, def. 2])**.** *Given $\delta > 0$, the δ-approximate stabilizer rank $\chi_\delta(\psi)$ of a pure state ψ on an n-qubit system is the smallest integer for which ψ can be approximated within δ as a superposition of $\chi_\delta(\psi)$ stabilizer states.*

[1] With respect to the stabilizer formalism, meaning in terms of T-cost.

[2] I.e., with probability 1.

For the purpose of obtaining decompositions of unitary operators themselves, we introduce analogous definitions for Clifford decompositions.

Definition 3.3 (Exact and approximate Clifford rank)**.** *For an operator on a quantum system, we define the exact and approximate Clifford ranks for decompositions into linear combinations of Clifford operators similarly.*

3.1 Restatement of decomposition problem

Problem. *Suppose we are in $\mathbb{C}^n$ and have a large but finite subset of vectors $\mathcal{X} = \{x_1, \ldots, x_\alpha\}$ that spans the space. Given an arbitrary vector in $\mathbb{C}^n$, how can we find a decomposition by elements of $\mathcal{X}$ within error of at most δ that minimizes the rank, or number of terms, of the decomposition?*

In our case, our overfull spanning sets are the stabilizer states and Clifford operators for a state space and its space of operators. We were tasked with investigating Clifford decompositions of operators by NASA Ames for use in modified stabilizer simulations; such has been mentioned as an area of development for other quantum simulators [8, 14].

- Known extensively in signal processing literature as the sparsification problem [20].
- Known problem in quantum information theory in constructing certain types of codes, such as Reed-Muller codes [3].

Finding a δ-approximate decomposition into k elements of $\mathcal{X}$ is equivalent to being within δ distance of the k-dimensional subspace spanned by those elements.

Definition 3.4 (Grassmannian, $\mathrm{Gr}_k(\mathbb{C}^n)$)**.** *We define the Grassmannian $\mathrm{Gr}_k(\mathbb{C}^n)$ to be the collection of all k-dimensional subspaces of $\mathbb{C}^n$. On account of the correspondence between the k-dimensional subspaces and k-rank projection operators, we are endowed with a inner product and metric given by:*

$$\langle W, V\rangle_{\mathrm{Gr}_k(\mathbb{C}^n)} := \langle \mathrm{Proj}_W, \mathrm{Proj}_V\rangle = \mathrm{Tr}(\mathrm{Proj}_W^\dagger \mathrm{Proj}_V),$$
$$\mathrm{dist}_{\mathrm{chord}}(W, V) = \tfrac{1}{\sqrt{2}}\| \mathrm{Proj}_V - \mathrm{Proj}_W \| = \tfrac{1}{\sqrt{2}}\sqrt{\mathrm{Tr}[(\mathrm{Proj}_V - \mathrm{Proj}_W)^\dagger(\mathrm{Proj}_V - \mathrm{Proj}_W)]}$$

Grassmannians have more structure than simply an inner product, and are actually dimension $k(n-k)$ compact smooth manifolds constructed from a quotient of the unitary group $U(\mathbb{C}^n)$. This, however, is not relevant to us, as we will primarily care about only geometric considerations related to the various forms of metrics and inner products that can be induced on $\mathrm{Gr}_k(\mathbb{C}^n)$.

With a metric and inner product established on the Grassmannians relating the geometry of linear subspaces, this problem of finding such k-rank δ-approximations turns into a covering problem $\mathrm{Gr}_k(\mathbb{C}^n)$.

3.2 Computational geometry viewpoint

Covering lemmas are basic tools in computational geometry as intermediary technical result concerning pairing down a cover for a subset of a metric space to form a subcover with desirable properties.

Theorem 3.1 (3r-covering lemma)**.** *Given a cover for a subset of a sufficiently nice metric space, there exists a subcover consisting of pairwise disjoint sets that can be dilated by* 3 *to once again cover the entire subset.*

Lifting a cover of δ-balls on $\mathrm{Gr}_k(\mathbb{C}^n)$ to $\mathbb{C}^n$ via the isometry provides a covering by regions that guarantee a δ-approximate k-decomposition, hence we wish to reduce our search space of subspaces check for decompositions.

Optimal configurations for packings minimize the angles between the centers of these balls, subject to restrictions on the maximal diameter according to a pre-determined error bound.

Theorem 3.2 (Conway-Hardin-Sloan simplex bound, [18, cor. 4])**.** *Given a finite set $S \subset \mathrm{Gr}_k(\mathbb{C}^n)$, the largest inner product α between any two distinct subspaces in S satisfies*

$$\alpha \geq k\frac{k|S| - n}{n|S| - n},$$

where equality occurs if and only if $|S| = n^2$ and and S form a simplex in a hyperplane of $\mathbb{R}^{n^2-1}$.

Recalling our problem, since we are starting with a given overfull basis to pare down to a packing, finding any near-optimal configuration depends on the geometry of the spanning set.

3.2.1 Geometry of stabilizer states

The configuration of stabilizer states of an n-qubit system are fairly well understood, as per Garcìa, Markov, and Cross's 2017 paper, *On the Geometry of Stabilizer States* [9].

- Stabilizer states are distributed uniformly on the unit circle of the state space, and locally are identical in with respect to relations with nearest neighbors.
- The maximal inner product attainable by any two n-qubit stabilizer states is $1/\sqrt{2}$, or equivalently, the minimal distance is $\sqrt{2-\sqrt{2}} \approx 0.7$.
- The probability of randomly sampling nearby stabilizer states approaches zero as the number of qubits increases, which would make it difficult to naïvely construct high resolution packings.

3.2.2 Geometry of Clifford operators

It is known that the collection of Clifford operators for a quantum system form a 3-design [22].

Definition 3.5 (Unitary t-design, [22, def. 1])**.** *A finite collection of unitaries $\mathcal{S}$ of dimension d is said to be a t-design for some integer t if for all linear operators X on $\mathbb{C}^{d\otimes t}$, the following holds:*

$$\sum_{C\in\mathcal{S}} C^{\otimes t} X (C^\dagger)^{\otimes t} = \int_{U(d)} U^{\otimes t} X (U^\dagger)^{\otimes t}\, d\mu_{\mathrm{Haar}}\,.$$

A rough interpretation of this property is that the Cliffords operators are distributed very evenly, enough for 3-twirling to be equivalent to Haar-random unitary twirls.

Oszmaniec, Sawicki, and Horodecki [14] connects unitary t-designs to ε-nets, meaning subsets of unitaries that approximate every unitary operation up error ε: only universal gatesets form ε-nets for arbitrary $\varepsilon > 0$, therefore there is a limit to which we can approximate with Cliffords.

3.3 Discussion on decomposition methods

Naïvely, searching through subspaces of very high-dimensional spaces is hard: in the case of k dimensional-subspaces spanned by Clifford operators, our search space is on the order of $\binom{\mathcal{O}(2^n)}{k}$.

Greedy algorithms for searching through a collection of subspace coverings containing a vector generally exist, but likely would require looking through the literature to deal with the non-orthogonality of potential decompositions; a good place to start would be [20, sec. III.D] and move on to reviewing more current methods in signal processing for the sparse representation problem.

However, since stabilizer states and Clifford operators are distributed relatively uniformly in space, these points may already form a near-optimal packing configuration. Therefore if looking for a computational advantage by reducing our search space of subspaces, we may already be in a worse case scenario. It follows that issues regarding the lower bounds on minimal error present in stabilizer or Clifford decompositions may present issues in implementing these methods computationally.

3.4 Application: Clifford decompositions of small Kraus operators with mixed-integer linear programming

One approach to simulating a Kraus operator K_i within the stabilizer formalism is to decompose the operator into a weighted sum of Cliffords,

$$K_i \approx \sum_{i}^{\chi_\delta} c_i R_i$$

Where χ_δ is the rank required to have δ L_2 error in the decomposition. Given a circuit $|\psi\rangle \to U_1 \circ K_1 \circ U_2 \circ K_2$ and the decompositions of the Kraus operators into Cliffords, we can simulate the circuit by creating $\chi_{\delta,i}$ copies of our stabilizer tableau for Kraus operator K_i, and applying one Clifford R_i from the decomp to one tableau S_i. At the end of our circuit, we can convert each tableau into it's vector representation, scale by the stored scalar c_i, and combine it with all other tableaus to retrieve the final ket of the circuit.

To find such decompositions, we can formulate this as an optimization problem by introducing the following variables:$c_{\text{real}} \in \mathbb{R}^m$, the real part of the coefficients; $c_{\text{imag}} \in \mathbb{R}^m$, the imaginary part of the coefficients;$y \in \{0,1\}^m$, binary variables indicating whether a Clifford operator is used;$e_{\text{real}} \in \mathbb{R}^4$, the real part of the error termsl; $e_{\text{imag}} \in \mathbb{R}^4$, the imaginary part of the error terms. λ is the weight placed on minimizing the rank. The larger lambda is the more weight will be placed on minimizing rank instead of error and

vice versa. The objective is to minimize the sum of the errors and the number of vectors used:

$$\min \sum_{i=1}^{4} e_{\text{real},i} + \sum_{i=1}^{4} e_{\text{imag},i} + \lambda \sum_{j=1}^{m} y_j$$

Subject to the following constraints for the real and imaginary parts:

$$\sum_{j=1}^{\chi_\delta} (R_{\text{real},ij} c_{\text{real},j} - R_{\text{imag},ij} c_{\text{imag},j}) - x_{\text{real},i} \leq e_{\text{real},i} \quad \forall i$$

$$x_{\text{real},i} - \sum_{j=1}^{\chi_\delta} (R_{\text{real},ij} c_{\text{real},j} - R_{\text{imag},ij} c_{\text{imag},j}) \leq e_{\text{real},i} \quad \forall i$$

$$\sum_{j=1}^{\chi_\delta} (R_{\text{real},ij} c_{\text{imag},j} + R_{\text{imag},ij} c_{\text{real},j}) - x_{\text{imag},i} \leq e_{\text{imag},i} \quad \forall i$$

$$x_{\text{imag},i} - \sum_{j=1}^{\chi_\delta} (R_{\text{real},ij} c_{\text{imag},j} + R_{\text{imag},ij} c_{\text{real},j}) \leq e_{\text{imag},i} \quad \forall i$$

Linking constraints to ensure coefficients are zero if not used:

$$c_{\text{real},j} \leq M y_j \quad \forall j$$

$$c_{\text{real},j} \geq -M y_j \quad \forall j$$

$$c_{\text{imag},j} \leq M y_j \quad \forall j$$

$$c_{\text{imag},j} \geq -M y_j \quad \forall j$$

Note that our search space is the finite set of n-qubit Cliffords, which becomes intractible at $n = 3$.

$$|C_n| = \prod_{j=1}^{n} 2(4^j - 1)4^j = 2^{n^2+2n} \prod_{j=1}^{n} (4^j - 1)$$

This tells us that at higher qubit systems, finding a suitable Clifford decomposition using mixed integer linear programming becomes infeasible. Also, the memory scales as $O(\prod_i^k \xi_\delta(K_i))$ because we must create $\xi_\delta(K_i)$ copies of however many tableaus we currently have each time we wish to apply another Kraus. As an example, consider a set of two Kraus operators, each with a rank two decomposition. Then every time we apply either Kraus to our system, our system grows by a factor of two, so the number of tableaus we are storing grows to 2^k where k is the number of Kraus operators being applied. The scaling of this method is unfavorable, which led us to consider other potential solutions.

Chapter 4

Dilation Methods

A *noise channel* $\xi_i = \{K_1, ..., K_{|\xi_i|}\}$ is a collection of distinct Kraus operators that form a completely positive and trace preserving (CPTP) map [13]. Where the *channel* size $|\xi_i|$ is number of Kraus operators contained in the noise channel. Kraus operators are traditionally non-clifford and therefor can not be naively applied within stabilizer formalism.

However through the application of stabilizer tableau's [2] and T-gadgets [6], we can implement circuits of unitary operators using stabilizer formalism in a reasonable runtime. In an effort to exploit these tools Suri & Marshall proposed the use of unitary dilation's to simulate noise using tools originally designed for simulating unitary circuits [19]. Bellow we discuss how unitaries can be simulating using stabilizer formalism, before exploring the potential applications of applying noise to a quantum circuit through Stinespring and Sz.-Nagy dilation's Dilations. We note that for the remainder of this section we assume that any noise channel ξ represents local noise and thus operates on one qubit.

4.1 Simulating Unitaries

A unitary operators can be arbitrarily approximated using the CNOT, Hadamard, Phase and T gates. In other words {CNOT, H, P, T} are considered a universal gate set.

$$\text{CNOT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \text{H} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \text{S} = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}, \text{T} = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\frac{\pi}{4}} \end{bmatrix}$$

While we know that its theoretically possible to form a Clifford + T approximation of any unitary, it is useful to have an algorithm capable of finding these decompositions. The problem of decomposing arbitrary unitaries into, {CNOT,H,P,T} gates is a well studied problem with recent results indicating that the upper bound on the number of {CNOT,H,P,T} gates required to form an accurate decomposition using the Solovay-Kiteav algorihtm is $O(log(\frac{1}{\epsilon})^{1.44042...+\delta})$ [11].

As discussed in section 2 the {CNOT,H,P} are contained in the Clifford group, and can therefore be efficiently simulated using stabilizer formalism. Whereas the T gate can be efficiently simulated by adding one ancillary $|T\rangle$ magic gate for every T gate in the unitary decomposition. At this point its useful to have some measure of magic as it will determine how many $|T\rangle$ ancilia must be added to simulate a unitary.

Definition 4.1 (T-Count). *ϵ T-Count $t_\epsilon(\cdot)$ is the number of T gates needed to approximate an arbitrary unitary in terms of $C = \{CNOT, H, P, T\}$ gates such that $||K_j - \prod_{i=0}^{l} C_i|| \leq \epsilon$*

Using the improved Solovay-Kitaev algorithm [11] along with T-count, we can define a general algorithm for simulating any unitary within stabilizer formalism.

Algorithm 1 Unitary Simulation UAPPLY(U,S)

Input 1: A decomposed Unitary $U = \prod_{i=0}^{\chi} C_i$
Input 1: A set of stabilizer tableau's S
for $j = 1$ to χ **do**
    if C_j contains a T gate **then**
        Use a T-Gadget
    else
        Apply C_j using the associated tableau rule.
    end if
end for

Algorithm 2 Unitary Simulation USIM(U)

Input 1: A Unitary U
Decompose U into $\prod_{j=1}^{\chi} C_j = \hat{U}$
Compute $t = t_\epsilon(U)$
Create a circuit S with t $|T\rangle$ gates and a state $|\psi\rangle$
UAPPLY($\hat{U}$,S)

For example for the unitary $U \in \mathbb{C}^{2\times 2}$, where $U = HPT$, USIM(U) would return the following circuit

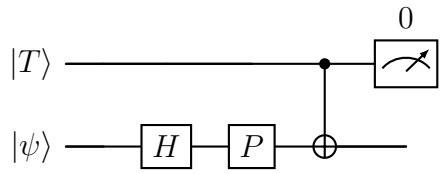


Figure 4.1: Unitary Simulation of $U = HPT$

4.2 Sz.-Nagy Dilation Approach

The first dilation approach will involve dilating each Kraus operator $K_i \in \xi$ separately. To do this we note that all Kraus operators are contractions lemma 4.1, and as a result they can all be dilated using the Sz.-Nagy unitary dilation.

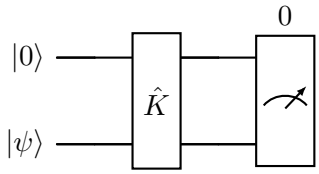
Lemma 4.1 (Kraus Contraction). *For all $K_j \in \xi_i$ K_j is a contraction operator*

Using the fact that a operator sum representation of a noise channel is trace preserving we know that $\sum_{i=0}^{k} KK^\dagger = I$. By extension we know that $\langle vK^\dagger, Kv\rangle = ||v||^2$ which implies that $||Kv||^2 \leq ||v||^2$. Therefore by definition a Kraus operator K is a contraction which proves lemma 4.1.

Theorem 4.1 (Sz.-Nagy Dilation [19]). *For every linear contraction operator A on a complex finite-dimensional Hilbert space $\mathcal{H}$, there exists a unitary dilation operator $U : \mathcal{H} \otimes \mathcal{H} \to \mathcal{H} \otimes \mathcal{H}$ in the following form:*

$$U_A = \begin{pmatrix} A & \sqrt{I - AA^\dagger} \\ \sqrt{I - A^\dagger A} & -A^\dagger \end{pmatrix}.$$

Using lemma 4.1 and theorem 4.1 we can design a quantum circuit that is equivalent to $K|\psi\rangle$, Where $\hat{K}$ is the Sz.-Nagy dilation of K.



The circuit detailed above is equivalent to first computing $\hat{K}(|0\rangle \otimes |\psi\rangle) = |0\rangle \otimes K|\psi\rangle + |1\rangle \otimes \sqrt{1 - K^\dagger K}|\psi\rangle$, and second measuring onto the 0 computational basis to retrieve component of interest, $K|\psi\rangle$. We can now define an algorithm that generalizes this process to stabilizer formalism

Algorithm 3 NAGYSINGLE(K)

1: **Input:** A Kraus operator K
2: Dilate $K \to \hat{K}$
3: Decompose $\hat{K}_{\xi_i,j}$ into $\prod_{j=1}^{\chi} C_j$
4: Compute $t = t_\epsilon(\hat{K})$
5: Create a set of tableaux that represent a circuit S with t $|T\rangle$ gates, one $|0\rangle$ ancilla, and a state $|\psi\rangle$
6: UAPPLY($\hat{K}$,S)
7: Apply a forced measurement onto the 0 computational basis to retrieve $K_1|\psi\rangle$

For example, a possible call of Algorithm 3, NAGYSINGLE(K), could generate and execute a circuit like that drawn bellow.

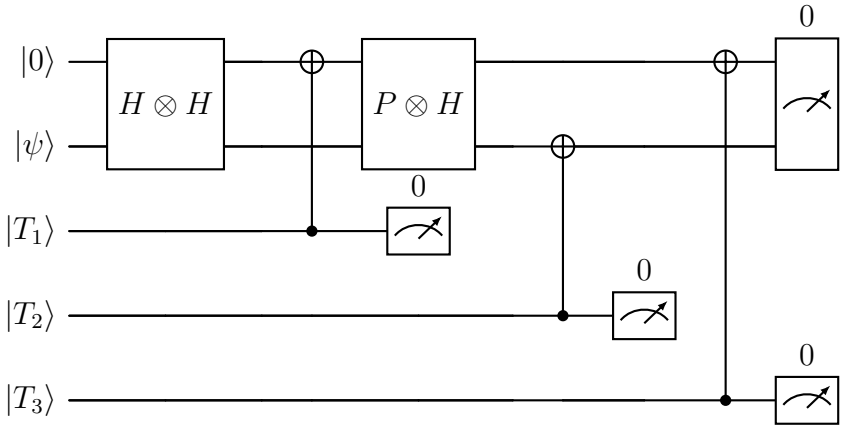


Figure 4.2: Circuit of Sz.-Nagy dilated Kraus $\hat{K} = H \otimes H \cdot T \otimes I \cdot P \otimes H \cdot I \otimes T \cdot T \otimes I$

Equipped with this intuition we can now construct an algorithm that allows us to apply multiple noise channels, instead of an individual Kraus operator. Note that the algorithm defined bellow makes use of the quantum trajectories method since it allows us to approximate the application of k noise channels without requiring the application of all k noise channels in their entirety. In fact, because the Sz.-Nagy dilation is a unitary dilation for one Kraus operator at a time, it is only natural that we make use of the quantum trajectories method which also applies one Kraus operator at a time. The quantum trajectories approach converges to the application of the all k noise channels in their entirety. [**?**].

Algorithm 4 Sz.-Nagy's Algorithm

1: **Input 1:** A list of noise channels $\Xi = \{\xi_1, \xi_2, \ldots, \xi_k\}$
2: **Input 2**: Error for the T-Gadget compression (δ)
3: **Input 3**: Error for the the {CNOT,H,P,T} decomposition (ϵ)
4: **Input 4**: Error for the approximate inner product at line 19 (η)
5: **Input 5**: Error for the the final state (Δ)
6: $t \leftarrow 0$
7: **for** $i = 1$ to k **do**
8: **for** $j = 1$ to $|\xi_i|$ **do**
9: Dilate $K_{\xi_i,j} \rightarrow \hat{K}_{\xi_i,j}$
10: Compute $t_\epsilon(\hat{K}_{\xi_i,j})$
11: **end for**
12: $t \leftarrow t + \max_{\hat{K}_{\xi_i,j} \in \xi_i}\{t_\epsilon(\hat{K}_{\xi_i,j})\}$
13: **end for**
14:
15: **for** $q = 1$ to $\lceil \frac{1}{\Delta^2} \rceil$ **do**
16: Create a circuit S with t $|T\rangle$ gates, one $|0\rangle$ ancilla, and a state $|\psi\rangle$
17: **for** $i = 1$ to k **do**
18: Pick $r \in [0, 1]$
19: **for** $j = 1$ to $|\xi_i|$ **do**
20: Copy $S \rightarrow S_1$
21: UAPPLY($\hat{K}_{\xi_i,j}$,S_1)
22: Retrieve $K_{\xi_i,j}|\psi\rangle$ and compute $p_{i,j} = \|K_{\xi_i,j}|\psi\rangle\|$
23: **if** $p_{i,j} \leq r$ **then**
24: Set $S \leftarrow S_1$
25: **break**
26: **else**
27: $r \leftarrow r - p_{i,j}$
28: Delete S_1
29: **end if**
30: **end for**
31: **end for**
32: Apply a forced measurement onto the 0 computational basis to retrieve a singular quantum trajectory $|Q_q\rangle = K_k \cdots K_2 K_1 |\psi\rangle$
33: **end for**
34: Take the average of all $\lceil \frac{1}{\Delta^2} \rceil$ $|Q_q\rangle$ to retrieve $\xi_k \circ \xi_{k-1} \circ ... \circ \xi_1(|\psi\rangle)$

Sz.-Nagy's Algorithm can be described as follows. We begin with k noise channels $\Xi = \{\xi_1, \xi_2, \ldots, \xi_k\}$ and an initial state $|\psi\rangle$. Next we find an upper bound for the total number of T-gates a single quantum trajectory may require, namely we must compute $t = \prod_{i=0}^{k} T_\epsilon(\xi_i)$, where $T_\epsilon(\xi_i) = \max_{\hat{K}_{i,j} \in \xi_i} \{t_\epsilon(\hat{K}_{i,j})\}$. Lastly we create a circuit S with t T-gadgets and one ancila, which can be represented using a set of $\frac{1.17^t}{\delta}$ tableaus $\{s_i\}_{i=0}^{\frac{1.17^t}{\delta}}$ [5]. At this point we have completed instantiating our circuit, so we can begin computing a quantum trajectory first uniformly generating a number $r \in [0, 1]$. Next we take $\xi_1 \in \Xi$ and pick the first dilated Kraus operator $\hat{K}_{1,1}$. We then make a copy S_1 of the set of tableaus S and apply the Clifford + T decomposition of $\hat{K}_{1,1}$. S_1 functionally serves as a playground to test out a potential $\hat{K}$ compute $p_{1,1} = ||\hat{K}_{1,1} |\psi\rangle||$ up to some error η using fast norm estimation [4]. If $p_{1,1}$ is $\leq r$ keep S_1 and dispose of S. If not dispose of S_1 and keep S and $r = r - p_{1,1}$. It is not guaranteed that the first Kruas operator will be choosen so we repeat the process of computing $p_{i,j}$'s until a Kraus is chosen. Only after a Kraus operator has been chosen for the first channel can we move onto the next channel to repeat the process of selecting a second Kraus operator. We continue until we have chosen a Kraus operator from each noise channel. At the end of this process we retrieve a single quantum trajectory $|Q_q\rangle = K_k \cdots K_2 K_1 |\psi\rangle$ through a forced measurement onto the 0 computational basis. However in order to achieve a final state $|\psi'\rangle$ that approximates a traditional application of the k noise channels within some error Δ, we must compute $\lceil \frac{1}{\Delta^2} \rceil$ quantum trajectories [**?**].

Theorem 4.2 (Sz.-Nagy Runtime Complexity)**.**

$$O(\frac{1}{\delta \Delta^2} s n^3 \eta^{-2} 1.17^t)$$

Where $s = \prod_{i=0}^{k} |\xi_i|$, $t = \prod_{i=0}^{k} T_\epsilon(\xi_i)$, n *is the dimension of the input* $|\psi\rangle$, η *is the acceptable error of each inner product calculation,* δ *is the acceptable error of the* T*-gadget compression, and* Δ *is the acceptable error of the final solution. Where error is defined as* $1 - \langle\psi|\psi'\rangle^2$

The runtime complexity is a result of computing at most s inner products per trajectory using fast norm estimation each of which grows at a rate of $O(\frac{s n^3 \eta^{-2} 1.17^t}{\delta})$. The cost of computing the Sz.-Nagy Dilation is $O(sn^3)$ which is dominated by the leading term contained in the runtime complexity. [4]

Theorem 4.3 (Sz.-Nagy Space Complexity)**.** *The space complexity of the simulating all* $\lceil \frac{1}{\delta^2} \rceil$ *trajectories grows as*

$$O(2\frac{1.17^t (t+2)^2}{8\delta})$$

The space complexity is a result of storing $\frac{1.17^t}{\delta}$ tableau's each of which can be stored as a binary matrix with an associated cost of $\frac{(t+2)}{8}$ bytes. Additionally at any given point we store two copies of the set of tableau's which doubles the space required.

4.3 Stinespring's Dilation Approach

The second dilation approach involves embedding the entire noise channel ξ into a unitary operator using Stinespring's Dilation.

Theorem 4.4 (Stinespring's Dilation)**.** *Given a quantum error channel* $\xi = \{K_1, K_2, ...K_m\}$ *we can lift the channel to a unitary of dimension* $2^m x 2^m$ *by the following construction, where we fill in the remaining entries using a Gram-Schmidt process such that the* K_{ξ_1} *is unitary.*

$$K_{\xi_1} = \begin{bmatrix} K_1 & \cdots & \cdots \\ K_2 & \cdots & \vdots \\ \vdots & \ddots & \vdots \\ K_m & \cdots & \cdots \end{bmatrix}$$

Using theorem 4.4 we can design a process that is equivalent to $\xi_k \circ \xi_{k-1} \circ ... \circ \xi_1(|\psi\rangle \langle\psi|)$ but operates within the stabilizer formalism. Take for example, the application of one noise channels ξ_1 which has size $|\xi_1| = 2$ and has been dilated to K_{ξ_1}. We find that $K_{\xi_1}(|0\rangle \otimes |\psi\rangle) = \begin{pmatrix} K_{\xi_1,1} |\psi\rangle \\ K_{\xi_1,2} |\psi\rangle \end{pmatrix} = |\psi'\rangle$. We can then compute $\text{tr}(\langle\psi'|\psi'\rangle)$ which is equivalent to $\sum_i^2 K_i |\psi\rangle \langle\psi| K_i^\dagger \rho'$. We can now generalize this one noise channel example to k noise channels using the following algorithm.

Algorithm 5 Stinespring's Algorithm

Input 1: A list of noise channels $\Xi = \{\xi_1, \xi_2, \ldots, \xi_k\}$
$t \leftarrow 0$
for $i = 1$ to k **do**
    Dilate $K_{\xi_i} \rightarrow \hat{K}_{\xi_i}$
    Compute $t_\epsilon(\hat{K}_{\xi_i})$
    $t \leftarrow t + \max_{\hat{K}_{\xi_i} \in \xi_i} \{t_\epsilon(\hat{K}_{\xi_i})\}$
end for

Create a set of tableaux that represent a circuit S with t $|T\rangle$ gates, $log_2(\sum_{i=1}^2 |\xi_i|)$ $|0\rangle$ ancilla, and a state $|\psi\rangle$

for $i = 1$ to k **do**
    UAPPLY($\mathbb{I}^{\otimes i-1} \otimes K_{\xi_i}$,S)
end for
Extract the final state $|\psi'\rangle$ and compute $\text{tr}(\langle\psi'|\psi'\rangle) = \rho'$

For example the associated circuit for applying two noise channels ξ_1 and ξ_2 where $t_\epsilon(K_{\xi_1}) = 1$ and $t_\epsilon(K_{\xi_2}) = 2$, would be the following.

The first step of Stinespring's Algorithms, like Sz.-Nagy's algorithm, is to compute the number of T gates that will be used throughout the simulation. The first major deviation is that unlike Nagy's algorithm where only one ancillary qubit is required, for Stinespring's algorithm 1 ancillary qubit is added for every noise channel. The circuit S will contain $log_2(\sum_{i=1}^2 |\xi_i|)$ ancillary $|0\rangle$ qubits and t T-gadgets which can again be stored using $\frac{1.17^t}{\delta}$ stabilizer tableau's. Afterwords we simply apply each dilated noise channel, before finally taking the outer product of the final state $|\psi'\rangle$. We note that taking the outer product of the final state $|\psi'\rangle$ is a post-processing step that can only be computing outside of stabilizer formalism. However we can reduce the computation burden of this step by only summing the outer product of all $2^n \times 2^n$ blocks of the output vector.

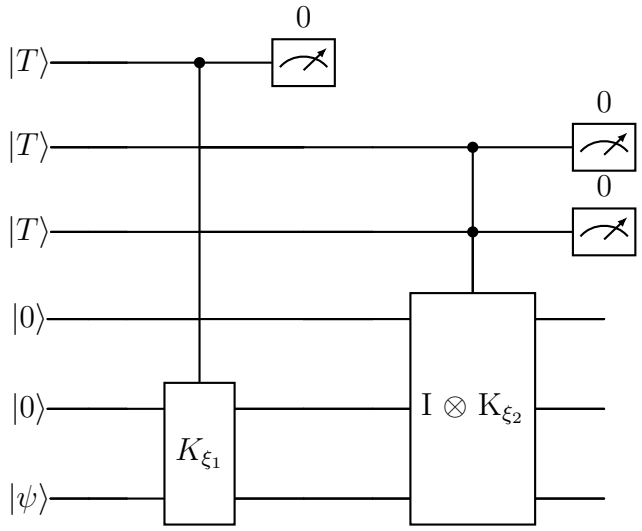


Figure 4.3: The application of two arbitrary noise channel onto a circuit using Stinespring's dilation.

Theorem 4.5 (Stinespring's Algorithm Runtime Complexity)**.** $O(n^2 \prod_{i=1}^{k} |\xi_i| + n^3 \sum_{i=1}^{k} |\xi_i|^3)$, *where n is the qubit size of the initial state $|\psi\rangle$*

The runtime complexity is a result of the $prod_{i=1}^{k}|\xi_i|$ outer products one must take each of which contributes a cost of n^2. The final outer product computation is the most expensive portion of the algorithm, however we do note that computing the stein springs dilation's requires a Gram-Schmit which contributes a cost of $n^3 \sum_{i=1}^{k} |\xi_i|^3$

Theorem 4.6 (Stinespring's Algorithm Space Complexity)**.** $O(\frac{(1+log_2(s)+t)^2 1.17^t}{8\delta})$, *where $s = \prod_{i=1}^{k} |\xi_i|$, n is the qubit size of the initial state $|\psi\rangle$, $t = \sum_{i=1}^{k} t_\epsilon(K_{x_i})$, and δ is the acceptable error of the T-gadget compression.*

The space complexity is similar to that of Sz.-Nagy's algorithm, with the only change being that each tableau has dimension $(1 + log_2(s) + t)$ instead of $(2 + t)$.

4.4 Examples

Next we investigate how many noise channels can be reasonably simulated using both dilation approaches. We take amplitude dampening channel as a case study since its one of the few frequently used non-Clifford 1-qubit noise channels.

Definition 4.2 (Amplitude Dampening Channel)**.** $\xi_{AD} = \left\{ K_1 = \begin{bmatrix} 1 & 0 \\ 0 & \sqrt{1-p} \end{bmatrix}, K_2 = \begin{bmatrix} 0 & \sqrt{p} \\ 0 & 0 \end{bmatrix} \right\}$

From the space complexity of both Nagy and Stinespring's algorithms we know that the T-count is the driving factor in memory consumption. As a result our first step is minimizing the t-count in order to best estimate how many ξ_{AD} channels a user could apply. To do this we first note that K_2 is a projector so an equivalent process to applying K_2 is first applying an X gate then applying a forced projective measurement of 0. In other words $K|\psi\rangle = P_{|0\rangle} X |\psi\rangle$ is a T gate free operation. Therefore we can focus our efforts on quantifying the T-Count of K_1. The Sz.-Nagy dilated form of K_1 can be recast as the following circuit.

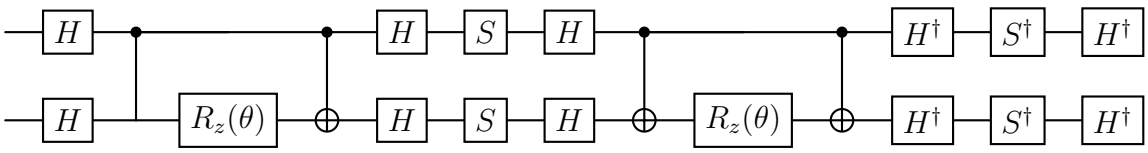


Figure 4.5: Stinespring Dilated Amplitude Dampening Circuit Equivalence

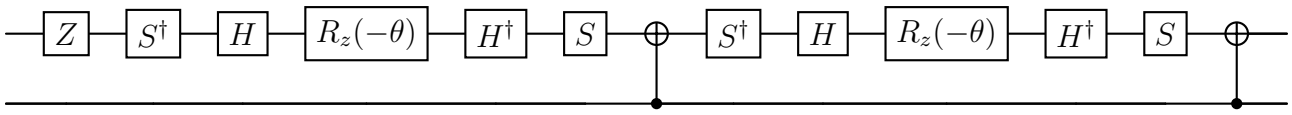


Figure 4.4: Sz.-Nagy Dilated Amplitude Dampening Circuit Equivalence

The Stinespring dilation of ξ_{AD} can also be recast as the following circuit through a few manipulations. Both circuit derivations are detailed in Appendix A.

We can observe that most of our T-Cost comes from the 2 R_z gates in both the Sz.-Nagy and Stinespring dilated unitaries. Luckily we know that the approximate t-cost is a function of how precise we want our p to be. For example a $p = .01$ would require 3 T gates per R_z gate, while a $p = .1$ would require 2 T gates per R_z gate [12]. Using these T count estimates for both the Sz.-Nagy and Stinespring algorithms we find that a user could reasonably apply **10-15** noise channels on a 16GB of memory.

Chapter 5

K-gadgets and Compression

The dilation methods are versatile in that they can simulate all combinations of Kraus operators. However, some quantum systems may experience the same type of noise repeatedly. In systems like these, rather than converting Kraus operators into unitary matrices and decomposing them into sequences of C,H,P and T gates, perhaps one can apply a K-gadget similar to what was done by Bravyi and Gosset [5]. The previous Chapter noted how the T-cost of even simple amplitude dampening channels could reach up to four per Kraus operator. The main benefit of using a K-gadget method is that instead of requiring four T-gadgets, we would only need a single K-gadget per Kraus operator.

Lemma 5.1. *Operating under the same gadget construction as the T-gadget, one can compress diagonal matrices* $K_1 = \begin{pmatrix} a & 0 \\ 0 & b \end{pmatrix}$ *and off-diagonal matrices* $K_2 = \begin{pmatrix} 0 & c \\ d & 0 \end{pmatrix}$ *with corresponding magic states* $m_1 = \begin{pmatrix} a \\ b \end{pmatrix}$ *and* $m_2 = \begin{pmatrix} c \\ d \end{pmatrix}$*. The corresponding gadgets are noted in the appendix, and are easy to verify.*

Moreover, both of these magic states can be written as a linear combination of two stabilizer states as the stabilizer states form a basis over $\mathbb{C}^2$ and $\mathbb{C}^2$ is dimension 2 over $\mathbb{C}$. Thus, any Kraus operator which is either diagonal or off-diagonal has a corresponding magic state $|M\rangle = \frac{|\tilde{0}\rangle + |\tilde{1}\rangle}{c}$ where $|\tilde{0}\rangle = |f\rangle$ and $|\tilde{1}\rangle = \alpha\,|g\rangle$, for any linearly independent pair of stabilizers $|f\rangle$ and $|g\rangle$ and some normalization constant c. For example, the Kraus channel $\begin{pmatrix} 1 & 0 \\ 0 & \sqrt{3}2 \end{pmatrix}$, corresponding to the amplitude dampening channel with parameter $p = 0.5$, has the corresponding magic state $|M\rangle = \begin{pmatrix} 1 \\ \frac{\sqrt{3}}{2} \end{pmatrix}$. This magic state can be written as a linear combination of $|0\rangle$ and $|+\rangle$ where $|M\rangle = (1 - \frac{\sqrt{3}}{2})\,|0\rangle + (\frac{\sqrt{6}}{2})\,|+\rangle = \frac{|+\rangle + \frac{2-\sqrt{3}}{\sqrt{6}}|0\rangle}{\frac{2\sqrt{6}}{6}}$, where $\alpha = \frac{(2-\sqrt{3})}{\sqrt{6}} \approx 0.1$. Notice that this parameterization of $|M\rangle$ in terms of α and two stabilizer states is really just a generalization of the $|H\rangle$-state, with the $|H\rangle$-state being the result for $\alpha = 1$ and two specific stabilizer states. Like Bravyi and Gosset's method, we aim to find an efficient decomposition of:

$$|\mathcal{L}\rangle^{\otimes t} = \frac{1}{\sqrt{K(\mathcal{L})}} \sum_{x \in \mathcal{L}} |\tilde{x}_1 ... \tilde{x}_t\rangle = \frac{1}{\sqrt{K(\mathcal{L})}} \sum_{x \in \mathcal{L}} \alpha^{|x|}\,|x_1 x_2 ... x_t\rangle$$

Where $K(\mathcal{L}) = \sum_{x,y\in\mathcal{L}} \nu^{|x+y|}\alpha^{|x|+|y|}$, with $\nu = \langle f|g\rangle$. We similarly define an approximation corresponding to a subspace $\mathcal{L}$ with:

$$|\mathcal{L}\rangle = \frac{1}{\sqrt{K(\mathcal{L})}} \sum_{x\in\mathcal{L}} |\tilde{x}_1 ... \tilde{x}_t\rangle = \frac{1}{\sqrt{K(\mathcal{L})}} \sum_{x\in\mathcal{L}} \alpha^{|x|} |x_1 x_2 ... x_t\rangle$$

We use the same error function, and we can see that:

$$\langle M^{\otimes t}|\mathcal{L}\rangle = \frac{1}{\sqrt{K(\mathbb{F}_2^1)^t K(\mathcal{L})}} \sum_{x\in\mathbb{F}_2^t, y\in\mathcal{L}} \alpha^{|x|+|y|}\nu^{|x+y|} \tag{5.1}$$

Rather than summing over all $x \in \mathbb{F}_2^t$, we can sum over all possible values of $|x+y|$, ranging from 0 to t. For any given y, the number of terms where $|x+y| = i$ is equal to $\sum_{j=0}^{i} \binom{|y|}{j}\binom{t-|y|}{j-i}$ where j is the number of one-bits that are flipped. Thus, $\binom{|y|}{j}\binom{t-|y|}{j-i}$ counts the number of x terms which are j 1-bit-flips and $i-j$ 0-bit-flips away from y. Thus, using some combinatorics, we can simplify Equation 5.1 to the following expression:

$$\langle M^{\otimes t}|\mathcal{L}\rangle = \frac{1}{\sqrt{K(\mathbb{F}_2^1)^t K(\mathcal{L})}} \sum_{y\in\mathcal{L}} \sum_{i=0}^{t} \alpha^{2|y|}\nu^i \sum_{j=0}^{i} \frac{1}{\alpha^{2j-i}} \binom{|y|}{j}\binom{t-|y|}{j-i} \tag{5.2}$$

The reason why we care about this error function is because it is significantly easier to calculate, as rather than iterating over all bitstrings in $\mathbb{F}_2^t$, one only needs to iterate over the terms in $\mathcal{L}$, which is a subspace significantly smaller than $\mathbb{F}_2^t$. Using this inner product formula, we ran several simulations to test how much we could compress $|M\rangle^{\otimes t}$, as well has what the error would be relative to each compression rate.

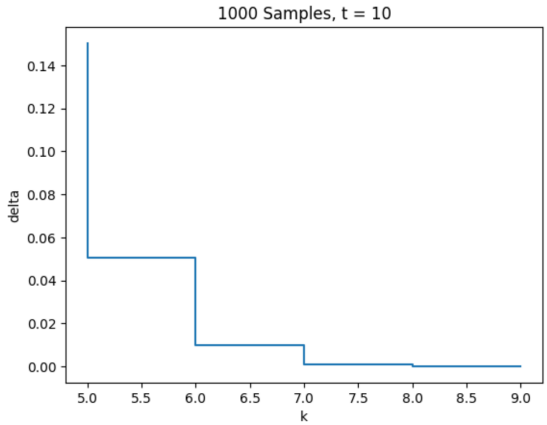


Figure 5.1: Caption: Minimum error across 1000 samples of subspaces across different dimensions. $|M\rangle$ has $\alpha = 0.5$.

In figure 5.1 we sampled 1000 subspaces of $\mathbb{F}_2^t$ with various for each dimension ranging from 5 through 9 and we plotted the minimum estimation error across the 1000 samples for each dimension. Notice using a 7-dimensional subspace, one can expect to achieve error below 0.01 across 1000 samples using a dimension seven subspace. For comparison, the method used in Bravyi and Gosset's paper only suggests a dimension nine decomposition.

The figure indicates that such magic states can be indeed be compressed, and that the true compression rate is lower than what is indicated in [5] for small t values.

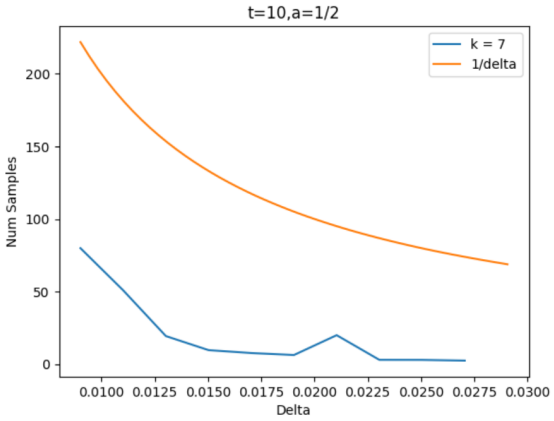


Figure 5.2: Number of samples required to achieve subspace with specific error values

Moreover, the number of samples actually required to find a seven dimensional subspace with corresponding approximation error less than 0.01 is actually quite small as well. Figure 5.2 shows the number of samples required to achieve a dimension 7 subspace with error less than a specific delta when approximating $|M\rangle^{\otimes t}$. This is shown in comparison to the expected number of samples needed to approximate $|H\rangle^{\otimes t}$ using the T-state compression method. Figure 5.2 show that the expected number of samples to achieve any given error is actually fewer in the K-gadget case when $\alpha = 0.5$ than the T-gadget case where α is effectively equal to 1.

Figures 5.1 and 5.2 indicate that magic states with $\alpha = 0.5$ can also be compressed more efficiently than the bounds found in [5], and that sampling a sufficiently accurate approximation would require fewer samples than expected in the $\alpha = 1$ case.

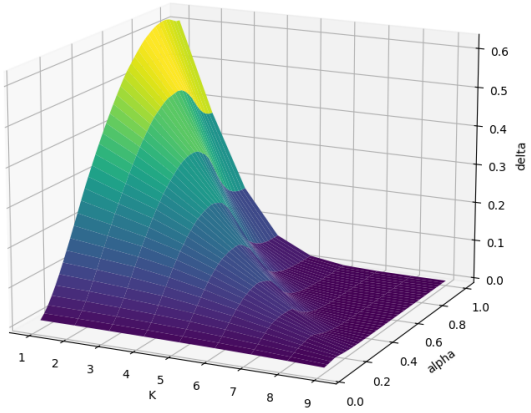


Figure 5.3: Minimum error across 250 samples for various α values and subspace dimensions.

Figure 5.3 shows the minimum error in approximating $|M\rangle^{\otimes 10}$ across 250 samples, while varying the dimensions sampled and the α values that make up $|M\rangle$. Indeed, the error monotonically decreases as the dimension of the approximation increases. However, the error does not monotonically increase as α increases.

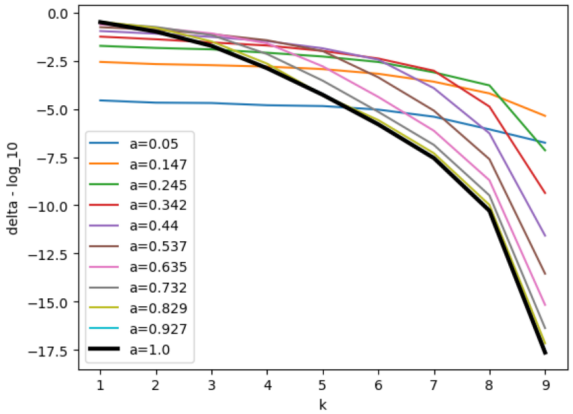


Figure 5.4: Approximations error across different k and α-values.

Figure 5.4 is the result of figure 5.3 when viewed from a particular angle. The figure shows thats that the gradient of the error with respect to the dimension differs for different values of α. In fact, the magnitude of the gradient increases as α increases, indicating that larger values for α benefit more from having higher rank than smaller values of α. The plot also indicates that magic states written with higher values of α can achieve overall lower levels of error than magic states with lower values of α.

The research on K-gadgets is still ongoing and there are many open questions that we have yet to answer. Our goal with K-gadgets is to find a similar bound to the bound provided in the T-gadget compression method, which would inform what subspaces to look over for lower rank compositions, as well as an the order of the expecterd number of samples required to find a subspace with sufficiently low error. We are also interested in how changing the two stabilizer states which compose $|M\rangle$ would affect the compression rate or the sampling process.

Chapter 6

Conclusion and Discussion

We end with a brief comparison and discussion of the four methods proposed throughout this report. Note that the Clifford decomposition approach is intentionally omitted since the $O(2^k)$ space complexity allowed us to rule out this approach early on in our research. the complexity analysis of the K-gadget approach is a rough estimate based on the numerical simulations listed in section 5.

Name	**Runtime Complexity**	**Space Complexity**	**# of Noise Channels**
Sz.-Nagy	$O(\frac{1}{\delta\Delta^2} s n^3 \eta^{-2} 1.17^t)$	$O(2^{\frac{1.17^t (t+2)^2}{8\delta}})$	10-15
Stinespring	$O(n^2 \prod_{i=1}^{k} \lvert\xi_i\rvert + n^3 \sum_{i=1}^{k} \lvert\xi_i\rvert^3)$	$O(\frac{(1+log_2(s)+t)^2 1.17^t}{8\delta})$	10-15
K-Gadget	$O(\lvert\Xi\rvert \cdot \lvert\xi\rvert)$	$O(\frac{(k+1)^2 (1+0.2\alpha)^k}{8\delta})$	57-11,312

Table 6.1: Comparison of Algorithms. Number of noise channels is computed given a 16GB limit in memory. For Sz.-Nagy and Stinespring the range is a result of varying precision of the noise and for the K-gadget approach the range is a result of varying α from 1 to 0

Simulating Quantum circuits is a computationally difficult problem, and the stabilizer formalism provides one of many possible routes to classical simulation. The stabilizer formalism is quite restrictive, only allowing a small set of unitary operators to be simulated efficiently. Furthermore, simulating non-unitary noise channels within the stabilizer formalism is a difficult problem as it does not natively support them. We have provided two main methods for implementing them.

The two dilation methods allow us to lift our Kraus operator or noise channel to a higher Hilbert space such that the lifted operator is unitary, and apply that unitary matrix to our stabilizer formalism using Clifford + T decomposition and applying Bravyi and Gosset's work. The Stinespring's approach allows us to simulate the whole channel with the high upfront computational cost of using the Gram Schmidt process to create our unitary matrices. It remains an open question whether there are more computationally friendly approaches to this process. The post-processing cost of converting the output to OSR is also a hefty computation, requiring $\prod_{i=1}^{k} \lvert\xi_i\rvert$ outer products. Comparatively, the Sz.- Nagy approach incurs more cost during the runtime of the circuit itself, requiring the use of quantum trajectories. Namely, the requirement of applying the Kraus operator K to the current state to find it's respective probability is the largest contributer to the algorithm's runtime

It should be noted that the K-gadgets method allows us to simulate diagonal and off-diagonal matrices with reasonable runtime. This means that we are not restricted to operators that satisfy the noise channel requirement $\sum KK^\dagger = I$, but this method can be applied to any diagonal and off-diagonal matrix. As is explained in Appendix A, this can also work

for arbitrary 2×2 operators, but with an exponential cost of 2^k where is the number of operators applied. It remains an open question what other methods can be employed to more efficiently simulate these arbitrary operators.

Appendix A

Circuit Equivalences

A.1 Circuit Derivations For Section 3.4

First and foremost we'd like to thank Namit Anand for proposing the application of Matchgates to solve this problem for Sz.-Nagy dilation and for solving this problem for the Stinespring dilation case.

A.1.1 Amplitude Dampening Circuit for Sz.Nagy Dilation

The goal will be to get approximate minimal T-Counts of the amplitude dampening channel using Sz. Nagy Dilation and Matchgates. Where the amplitude dampening channel is defined as follows

$$\xi_{AD} = \left\{ K_1 = \begin{bmatrix} 1 & 0 \\ 0 & \sqrt{1-p} \end{bmatrix}, K_2 = \begin{bmatrix} 0 & \sqrt{p} \\ 0 & 0 \end{bmatrix} \right\}$$

We will focus our attention on K_1, whose Sz.-Nagy dilation is the following, when $p = \sin^2(\frac{\theta}{2})$

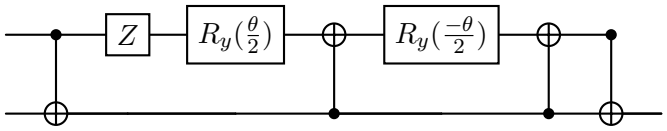
$$\hat{K}_\theta = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\frac{\theta}{2}) & 0 & \sin(\frac{\theta}{2}) \\ 0 & 0 & -1 & 0 \\ 0 & \sin(\frac{\theta}{2}) & 0 & -\cos(\frac{\theta}{2}) \end{bmatrix}$$

We can show that CNOT $\hat{K}_\theta$ CNOT recovers a Givens rotation

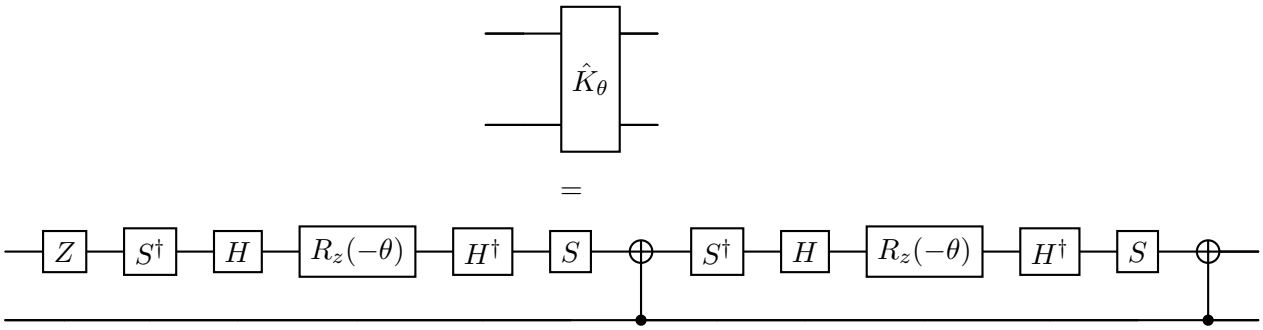
$$\text{CNOT}\,\hat{K}_\theta\,\text{CNOT} = G_{AB}(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\frac{\theta}{2}) & \sin(\frac{\theta}{2}) & 0 \\ 0 & \sin(\frac{\theta}{2}) & -\cos(\frac{\theta}{2}) & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$$

In other words these two circuits are identical

Ramelow established some useful circuit identities for Givens rotations which we use to rewrite G_{AB} as the following circuit [17]



Using the identity that $R_y(\theta)$ can be rewritten as $R_y(\theta) = S^\dagger H Rz(-\theta) H^\dagger S$, we can again rewrite the following circuit equality



A.1.2 Amplitude Dampening Circuit for Stinespring's Dilation

The Stinespring dilation for ξ_{AD} gives us the following unitary matrix if we parameterize $p = \sin^2(\theta/2)$

$$G(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta/2) & -\sin(\theta/2) & 0 \\ 0 & \sin(\theta/2) & \cos(\theta/2) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

This is known as a Givens rotation. It is easy to check that:

$$G(\theta) = \exp\left[-i\theta/2(Y \otimes X - X \otimes Y)\right]$$

To find an optimal Clifford+T decomposition of this gate, we use the following observations. First, recall that the iSWAP gate is a Clifford unitary defined as:

$$\text{iSWAP} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & i & 0 \\ 0 & i & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

We can parameterize this gate as the following equality and can recover the above gate for $\theta = \pi$.

$$\text{iSWAP}(\theta) = \exp\left[i\theta/2(X \otimes X + Y \otimes Y)\right]$$

The first claim is that:

$$G(\theta) = (T^{-1} \otimes T) \cdot \text{iSWAP}(\theta) \cdot (T \otimes T^{-1})$$

This tells us how to generate the Givens rotation using T-gates and iSWAP gates. Now, using the fact that $[X \otimes X, Y \otimes Y] = 0$, we can simplify the iSWAP gate as:

$$\text{iSWAP}(\theta) = \exp\left[i\theta/2(X \otimes X)\right] \cdot \exp\left[i\theta/2(Y \otimes Y)\right]$$

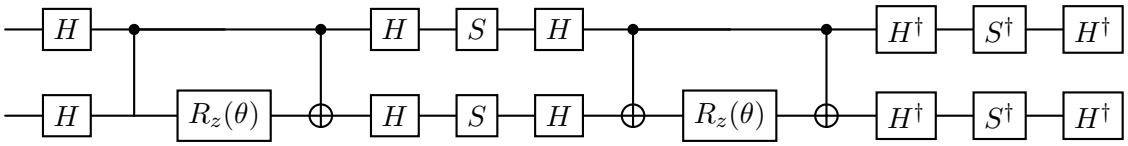
Moreover, using the following two identities:

$$HZH^\dagger = X \quad \text{and} \quad (HSH)Z(HSH)^\dagger = -Y$$

$$\exp\left[i\theta/2(Z \otimes Z)\right] = \text{CNOT} \cdot (I \otimes R_z(\theta)) \cdot \text{CNOT}$$

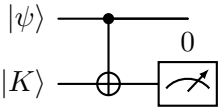
we can write the iSWAP gate as the following circuit

$$\text{iSWAP}(\theta) = H^{\otimes 2} \cdot \text{CNOT} \cdot (I \otimes R_z(\theta)) \cdot \text{CNOT} \cdot H^{\dagger \otimes 2} \cdot (HSH)^{\otimes 2} \cdot \text{CNOT} \cdot (I \otimes R_z(\theta)) \cdot \text{CNOT} \cdot (HSH)^{\dagger \otimes 2}$$

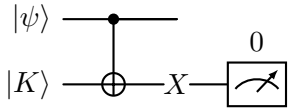


A.2 K-gadgets

Utilizing the fact that we can force a measurement in the Z basis to be $|0\rangle$, we create a gadget that can apply a Kraus operator to a state $|\psi\rangle$. Let $K = \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}$. Then the gadgetized K is $|K\rangle = \begin{bmatrix} a \\ b \end{bmatrix}$. Then to apply the diagonal Kraus operator, we apply the following circuit:



Which produces $K|\psi\rangle$. A similar circuit works for the diagonal matrix $K = \begin{bmatrix} 0 & a \\ b & 0 \end{bmatrix}$. Let $|K\rangle = \begin{bmatrix} b \\ a \end{bmatrix}$. Then



gives you $K|\psi\rangle$.

Note that you can also simulate any Kraus operator $K = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$ by decomposing it into the sum of the diagonal components and the off-diagonal components. However, it has scaling similar to Clifford decomp of rank 2, that is, it scales as 2^k where k is the number of applied Kraus operators.

Bibliography

[1] *40 years of quantum computing*, Nature Reviews Physics, 4 (2022), p. 1.

[2] S. Aaronson and D. Gottesman, *Improved simulation of stabilizer circuits*, Physical Review A, 70 (2004).

[3] A. Ashikhmin and A. R. Calderbank, *Grassmannian Packings From Operator Reed–Muller Codes*, IEEE Transactions on Information Theory, 56 (2010), pp. 5689–5714.

[4] S. Bravyi, D. Browne, P. Calpin, E. Campbell, D. Gosset, and M. Howard, *Simulation of quantum circuits by low-rank stabilizer decompositions*, Quantum, 3 (2019).

[5] S. Bravyi and D. Gosset, *Improved Classical Simulation of Quantum Circuits Dominated by Clifford Gates*, Physical Review Letters, 116 (2016).

[6] S. Bravyi, G. Smith, and J. A. Smolin, *Trading Classical and Quantum Computational Resources*, Physical Review X, 6 (2016).

[7] G. et al., *Suppressing quantum errors by scaling a surface code logical qubit*, Nature, 614 (2023).

[8] H. J. Garcia and I. L. Markov, *Simulation of Quantum Circuits via Stabilizer Frames*, IEEE Transactions on Computers, 64 (2015), pp. 2323–2336.

[9] H. J. García, I. L. Markov, and A. W. Cross, *On the Geometry of Stabilizer States*, Nov. 2017. arXiv:1711.07848 [quant-ph].

[10] M. Howard and E. Campbell, *Application of a Resource Theory for Magic States to Fault-Tolerant Quantum Computing*, Physical Review Letters, 118 (2017), p. 090501.

[11] G. Kuperberg, *Breaking the cubic barrier in the Solovay-Kitaev algorithm*, 2023. Version Number: 1.

[12] G. J. Mooney, C. D. Hill, and L. C. L. Hollenberg, *Cost-optimal single-qubit gate synthesis in the clifford hierarchy*, Quantum, 5 (2021), p. 396.

[13] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information: 10th Anniversary Edition*, Cambridge University Press, 10th anniversary ed., June 2012.

[14] M. Oszmaniec, A. Sawicki, and M. Horodecki, *Epsilon-nets, unitary designs and random quantum circuits*, 2020. Version Number: 3.

[15] F. Pan and P. Zhang, *Simulating the Sycamore quantum supremacy circuits*, Mar. 2021. arXiv:2103.03074 [physics, physics:quant-ph].

[16] E. Pednault, J. A. Gunnels, G. Nannicini, L. Haoresh, T. Magerlein, E. Solomonik, E. W. Draeger, E. T. Holland, and R. Wisnieff, *Breaking the 49-Qubit Barrier in the Simulation of Quantum Circuits*, Tech. Rep. LLNL-JRNL-747743, Lawrence Livermore National Laboratory (LLNL), Livermore, CA (United States), Feb. 2018.

[17] S. Ramelow, A. Fedrizzi, A. M. Steinberg, and A. G. White, *Matchgate quantum computing and non-local process analysis*, New J. Phys., 12 (2010), p. 083027.

[18] A. Roy, *Bounds for codes and designs in complex subspaces*, Journal of Algebraic Combinatorics, 31 (2010), pp. 1–32.

[19] N. Suri, J. Barreto, S. Hadfield, N. Wiebe, F. Wudarski, and J. Marshall, *Two-Unitary Decomposition Algorithm and Open Quantum System Simulation*, Quantum, 7 (2023).

[20] J. Tropp, *Greed is Good: Algorithmic Results for Sparse Approximation*, IEEE Transactions on Information Theory, 50 (2004), pp. 2231–2242.

[21] G. Vidal, *Efficient Classical Simulation of Slightly Entangled Quantum Computations*, Physical Review Letters, 91 (2003).

[22] Z. Webb, *The Clifford group forms a unitary 3-design*, 2015. Publisher: arXiv Version Number: 3.