



Compiling single-qubit gates for spin qubits at zero magnetic field

Csongor Hunyady

Physicist-Engineer

TDK thesis

Supervisors

András Pályi

Associate professor

BME, Department of Theoretical Physics

Baksa Kolok

PhD student

BME, Department of Theoretical Physics

Budapest University of Technology and Economics

November 3, 2024

Abstract

Experimental efforts towards quantum computing explore different physical platforms of quantum bits. A promising platform is based on electrically defined quantum dots in semiconductors. In this architecture, each qubit is realised by the spin of a single electron. Traditionally, spin qubits are operated in a static, homogeneous magnetic field, that induce a Larmor frequency of a few GHz, and single-qubit logical operations are performed by resonant driving. Recent experiments, however, have indicated that the qubit lifetime decreases as the magnetic field increases, hence it might be optimal to operate the qubits at zero magnetic field.

In this work, I investigate how to obtain a “target” single-qubit gate for spin qubits at zero magnetic field, as a sequence of two given “resource” single-qubit gates. The two resource gates can be realized by utilizing spin-orbit coupling as the electron is shuttled through two different loops of quantum dots in a two-dimensional quantum dot array. To obtain the target single-qubit gate, we use the Solovay-Kitaev algorithm. In particular, we compute the probability distribution of the sequence length needed for the Hadamard gate H and the phase gate S , assuming that the two resource gates are random. In the short term, our results facilitate Ramsey-type measurements and quantum state tomography experiments to monitor spin qubit decoherence at zero magnetic field. On the long run, the methods we develop might find use in quantum circuit compilers for quantum computing platforms where the qubits operate at zero energy splitting.

Contents

1	Introduction	3
2	Spin-based Quantum Computing	5
3	Holonomic Quantum Computing	5
4	Shuttling-Based Holonomic Quantum Computation	7
5	Bloch Sphere	7
6	Generating Random Unitary Matrices	8
7	Fidelity	9
8	The Brute Force approach	11
8.1	Cumulative density function of the error	11
8.2	Exponential fit of error	12
8.3	Error Outliers	15
8.4	Brute Force complete gate set	15
8.5	Bloch Sphere visualization	16
8.6	"Fixing" bad sets	19
9	The Solovay-Kitaev approach	22
9.1	The Solovay-Kitaev Algorithm	22
9.2	The Solovay-Kitaev theorem applied	23
9.3	Solovay-Kitaev Results	23
10	Solovay-Kitaev and Brute Force comparison	25
11	Conclusion	28
	References	30

1 Introduction

Quantum computing represents a transformative approach to information processing, leveraging the unique principles of quantum mechanics to tackle computational tasks that are infeasible for classical systems. At the core of quantum computing are quantum bits, or *qubits*, which, unlike classical bits that are limited to representing either a 0 or a 1, can exist simultaneously in multiple states through quantum superposition. This fundamental property enables quantum systems to encode and manipulate an exponentially larger amount of information compared to classical bits.

Beyond superposition, quantum computers exploit entanglement phenomenon where qubits are so strongly correlated such that from the measurement outcome of one, one can predict the measurement outcome of the other. Additionally, quantum interference allows the probabilities of qubit states to amplify or cancel out, enabling the precise manipulation of quantum information. These properties are the foundation of quantum algorithms that can outperform their classical counterparts in areas such as cryptography, optimization, and the simulation of quantum systems.

Applications of quantum computing span a wide range of fields where these properties make a real difference. For instance, Shor’s algorithm [1] leverages superposition and entanglement to factorize large integers in polynomial time, which could have significant implications for cryptographic security. Quantum systems are also particularly well-suited for simulating other quantum systems [2], making quantum computing especially promising for areas like condensed matter physics, quantum chemistry, and materials science, where classical simulations struggle due to their complexity. Additionally, there are potential applications in solving optimization problems, improving machine learning algorithms, and solving large linear systems more efficiently.

However, realizing the full potential of quantum computing is extremely challenging. Quantum states are inherently fragile and highly sensitive to their environment, which causes them to lose their quantum properties through a process called decoherence. To overcome these challenges, researchers are focused on developing robust qubit designs, advanced error correction techniques [3], and scalable, fault-tolerant quantum systems [4]. Different types of qubits—such

as superconducting circuits [5], trapped ions [6], and semiconductor quantum dots [7, 8]—are being explored, each with its own trade-offs in terms of coherence times, gate fidelities, and scalability. Recent experiments [9] suggest that spin-qubits are more resilient to decoherence and have longer lifetimes in zero magnetic fields. Furthermore harnessing SOC in zero magnetic field allows us to create geometric gates by utilizing Rashba and Dresselhaus SOC.

In this paper, I focus on the utilization of geometric phases and spin-orbit coupling (SOC) as mechanisms to implement stable and efficient quantum gate operations. By investigating these approaches, this work seeks to contribute to the development of more resilient quantum information processing techniques.

This paper is organized as follows. First, I introduce the foundational concepts and processes essential to this study, focusing on key elements and techniques in quantum computing in Sec. 1. Afterwards I explain the basics of spin-based quantum computing in Sec. 2, next I discuss holonomic quantum computing in Sec. 3 and formulate the question this work aims to address. Next I introduce quantum computing concepts that will be used later on in the paper, like the Bloch sphere in Sec. 5, random unitary generation in Sec. 6 and fidelity in Sec. 7. Following this, I present my contributions, examining the practical feasibility of implementing such systems through sequences of SOC-driven operations, first using brute-force approach in Sec. 8 then using the Solovay-Kitaev algorithm in Sec. 9. Afterwards I compare the results in Sec. 10. Finally, I present my conclusion in Sec. 11 and briefly discuss future research opportunities.

2 Spin-based Quantum Computing

Spin qubits represent one specific type of qubit, where quantum information is represented by the spin state of an electron or hole confined in a semiconductor quantum dot. The spin states—usually “up” ($|\uparrow\rangle$) and “down” ($|\downarrow\rangle$)—provide the two-level system needed for qubit representation. Quantum dots act as nanoscale traps that localize these spins, and their states can be manipulated using magnetic or electric fields.

The concept of spin qubits in quantum dots was introduced by Loss and DiVincenzo in [10], proposing that single spins in neighboring quantum dots could be used for quantum computation through exchange interactions. This theory led to further experimental efforts, initially with materials like silicon and gallium arsenide, and used techniques like electron spin resonance (ESR) [11, 12] and electric dipole spin resonance (EDSR) [13] to control spins.

Recent advances have focused on improving coherence times and gate fidelities by optimizing materials. Germanium is emerging as a promising platform for spin qubits due to its favorable electronic properties, such as strong spin-orbit coupling (SOC) and the absence of low-energy valley states that are problematic in silicon-based qubits [14]. SOC in germanium enables rapid spin manipulation through electric fields, reducing the need for complex magnetic setups and simplifying device architecture [15]. Additionally, recent studies have identified optimal operating points for germanium qubits, balancing coherence with rapid gate performance under SOC [16].

3 Holonomic Quantum Computing

Holonomic quantum computing (HQC) is a model that performs quantum operations using geometric phases. These phases, introduced by Berry [17], are phase shifts that a quantum state acquires as it evolves along a closed loop in parameter space, and they are unique in that they do not depend on the traversal speed. This property is particularly valuable because it provides inherent robustness against certain errors that affect conventional approaches [18].

In HQC, the computational space is encoded in degenerate eigenspaces of a Hamiltonian that is controlled by external parameters. As the system moves through closed loops in param-

eter space, it accumulates non-Abelian holonomies that act as unitary transformations on the encoded states. This mechanism, as described by Wilczek and Zee [19] and further explored by Zanardi and Rasetti [20], enables universal quantum computation through sequences of loops. The non-Abelian geometric phase associated with this evolution is represented by the holonomy:

$$\Gamma_A(\gamma) = P \exp \left(\int_{\gamma} A \right), \quad (1)$$

where P represents path ordering. Here, A is known as the adiabatic connection or gauge potential and serves as a non-Abelian gauge potential associated with the control parameters. It is defined as

$$A = \sum_{\mu} A_{\mu} d\lambda_{\mu}, \quad (2)$$

where each component A_{μ} is given by

$$(A_{\mu})_{\alpha\beta} := \langle \psi_{\alpha}(\lambda) | \frac{\partial}{\partial \lambda_{\mu}} | \psi_{\beta}(\lambda) \rangle, \quad (3)$$

with λ_{μ} denoting the coordinates in the parameter space and $\psi_{\alpha}(\lambda)$ representing the degenerate eigenstates of the Hamiltonian at each point in parameter space. This connection A allows for parallel transport within the eigenspace and is responsible for the holonomic transformations that enable quantum computation.

Operating in the absence of an external magnetic field preserves time-reversal symmetry, which introduces a twofold degeneracy of the spin states, known as a Kramers pair [21]. This degeneracy is beneficial because it allows computation within a protected subspace, which is less susceptible to certain types of noise and decoherence. By manipulating these degenerate states along closed loops in parameter space, robust quantum gate operations can be realized, leveraging the stability of the accumulated geometric phase.

In semiconductor quantum dots, holonomic gates can be implemented using spin-orbit coupling (SOC) effects. Due to confinement, a Rashba or Dresselhaus type spin-orbit interaction emerges, which is particularly strong in germanium devices [22]. By precisely controlling the evolution of quantum states, these SOC interactions enable the implementation of holonomic quantum gates within the HQC framework.

This combination of geometric stability, SOC control, and the inherent robustness of geometric phases makes holonomic quantum computation an intriguing approach for quantum information processing in solid-state systems.

4 Shuttling-Based Holonomic Quantum Computation

In the shuttling-based holonomic quantum computation model, a spin qubit is moved around a closed discrete path involving multiple quantum dots, forming geometric single-qubit gates. The idea is that adiabatic movement of the qubit through loops induces a geometric phase — often called a non-Abelian Berry phase — which acts as a unitary transformation on the state. As described by Kolok and Pályi [23], shuttling through a quantum dot loop in the presence of spin-orbit coupling provides a mechanism for accumulating such geometric phases, which can be leveraged as holonomic quantum gates for quantum computation purposes.

If multiple loops are present on the device, such as in recent experimental setups [24], each loop induces a different gate. These gates depend on various factors, including the confinement potential, the mechanical stress profile of the device, and defects, so they can be considered random. We utilize two such loop-induced random unitaries as "resource" gates in our study. Specifically, we explore how, using sequences composed of these two random unitaries and their inverses, we can approximate any target gate, such as the Hadamard gate H or the phase gate S . In the following sections, we examine the feasibility of constructing these target gates using sequences of the available resource gates.

5 Bloch Sphere

The Bloch sphere is a powerful visualization tool for representing the quantum states of a single qubit. On this sphere, each point corresponds to a unique qubit state, allowing us to describe the state as a vector in three-dimensional space. The north and south poles represent the computational basis states $|0\rangle$ and $|1\rangle$, while any other point on the sphere represents a superposition of these states.

Each point on the Bloch sphere can be parameterized by two angles, θ and ϕ , which

uniquely define a quantum state as follows:

$$|\psi\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + e^{i\phi}\sin\left(\frac{\theta}{2}\right)|1\rangle, \quad (4)$$

where $\theta \in [0, \pi]$ is the polar angle, measuring the position between the north and south poles, and $\phi \in [0, 2\pi)$ is the azimuthal angle, representing the position around the equator of the sphere. This parameterization enables us to visualize any pure single-qubit state as a point on the surface of the Bloch sphere, with θ and ϕ defining the exact location of the state.

A key feature of the Bloch sphere representation is that rotations on the sphere correspond directly to transformations of the qubit state, represented by quantum gates. Specifically, each unitary operation in the $SU(2)$ group can be interpreted as a rotation on the Bloch sphere, acting on the state vector to achieve the desired transformation. This connection is formalized by the expression:

$$U = e^{i\frac{\theta}{2}\vec{n}\cdot\vec{\sigma}}, \quad (5)$$

where $\vec{n}$ is a unit vector indicating the rotation axis, θ is the rotation angle, and $\vec{\sigma} = (\sigma_x, \sigma_y, \sigma_z)$ represents the vector of Pauli matrices. This formulation shows that any $SU(2)$ matrix can be seen as a rotation of the qubit's state vector on the Bloch sphere by an angle θ around the axis $\vec{n}$.

This approach allows us to visualize both the states and the effect of quantum gates as rotations, making the Bloch sphere a valuable tool for understanding and analyzing single-qubit systems.

6 Generating Random Unitary Matrices

To generate a random unitary matrix U in $SU(2)$, we begin by drawing four independent random variables from a standard normal distribution, say a , b , c , and d . These random variables will be used to construct a quaternion, which we then normalize to ensure uniform distribution according to the Haar measure. Here's the expanded process with formulas as described in [25]:

- Draw four independent random variables from a standard normal distribution:

$$a, b, c, d \sim \mathcal{N}(0, 1).$$

- Normalize the vector:

Form a vector $\mathbf{v} = (a, b, c, d)$ and normalize it to unit length:

$$\|\mathbf{v}\| = \sqrt{a^2 + b^2 + c^2 + d^2}. \quad (6)$$

- Normalize the components:

$$a' = \frac{a}{\|\mathbf{v}\|}, \quad b' = \frac{b}{\|\mathbf{v}\|}, \quad c' = \frac{c}{\|\mathbf{v}\|}, \quad d' = \frac{d}{\|\mathbf{v}\|}. \quad (7)$$

- Construct the SU(2) matrix:

Use the normalized variables to construct the unitary matrix $U \in \text{SU}(2)$:

$$U = \begin{pmatrix} a' + ib' & c' + id' \\ -c' + id' & a' - ib' \end{pmatrix}. \quad (8)$$

7 Fidelity

In quantum computing, assessing the accuracy of an operation or state preparation typically involves calculating the fidelity [26] between the desired target and the actual outcome. Fidelity provides a quantitative measure of "closeness" or overlap between quantum states, with values ranging from 0 to 1, where a value of 1 signifies identical states (up to a global phase), and lower values indicate increasing deviation. This metric is advantageous in quantum computing, as it inherently disregards global phase differences, which do not affect computational outcomes.

For two quantum states, represented by density matrices ρ (the target state) and σ (the

actual state), the fidelity $F(\rho, \sigma)$ is defined as [27]:

$$F(\rho, \sigma) = \left(\text{Tr} \sqrt{\sqrt{\rho} \sigma \sqrt{\rho}} \right)^2. \quad (9)$$

When both ρ and σ are pure states, this simplifies to $F(\rho, \sigma) = |\langle \psi | \phi \rangle|^2$, where $|\psi\rangle$ and $|\phi\rangle$ are the state vectors of ρ and σ , respectively.

For quantum operations, fidelity can quantify the overlap between two unitary gates, taking an average over possible input states. Referring to the derivations in the research paper by [26], the average fidelity $F(U, V)$ between two unitary operations represented by matrices U and V is given by:

$$F(U, V) = \frac{1}{d^2} \left| \text{Tr}(U^\dagger V) \right|^2, \quad (10)$$

where U and V are $d \times d$ unitary matrices. For the specific case of 2×2 matrices ($d = 2$), this simplifies to:

$$F(U, V) = \frac{1}{4} \left| \text{Tr}(U^\dagger V) \right|^2. \quad (11)$$

In both cases, a fidelity of 1 indicates that U and V are identical up to a global phase, while values closer to 0 indicate a greater "distance" between the operations. According to [26], the average fidelity is particularly well-suited for assessing the accuracy of a sequence of unitary operations in approximating a target gate, as it captures the phase-independent degree of overlap between the desired and implemented transformations.

Infidelity, defined as $1 - F$, provides a convenient measure of deviation from the target operation, where values closer to 1 represent a significant mismatch. We will use this as our error metric, since it provides a more intuitive way to measure deviation, since the higher the error the higher the deviation as well.

8 The Brute Force approach

Our stated goal was to examine the relationship between the error and sequence length in order to determine how feasible the aforementioned method of quantum computing is. To do this, we generate a set of 2 random unitary matrices, and using these matrices and their inverses we search for each given sequence length, the one which, after multiplying it out best approximates our target, in the following section, this will be the Hadamard gate.

8.1 Cumulative density function of the error

After generating 1000 pairs of random resource pairs and saving the results we begin by examining the following graph (see Fig. 1), which displays the cumulative distribution function (CDF) for four sequence lengths. The figure represents the probability that the error is less than or equal to x , mathematically defined as

$$F(x) = P(\text{Error} \leq x).$$

In this graph, $f(x) = \frac{\text{number of pairs with error} \leq x}{1000}$, showing the fraction of pairs for which the error is less than x . According to the law of large numbers, this function $f(x)$ approaches the distribution function as the number of pairs increases.

I chose to display four different sequence lengths to observe general trends without overcrowding the figure. The graph for the 10-length sequences is of particular interest, as it has the lowest error values. From the "tail" of this graph, we observe a sharp bend from approximately 0.01 to around 0.1, which gives us insight into the distribution of errors for different sequence lengths.

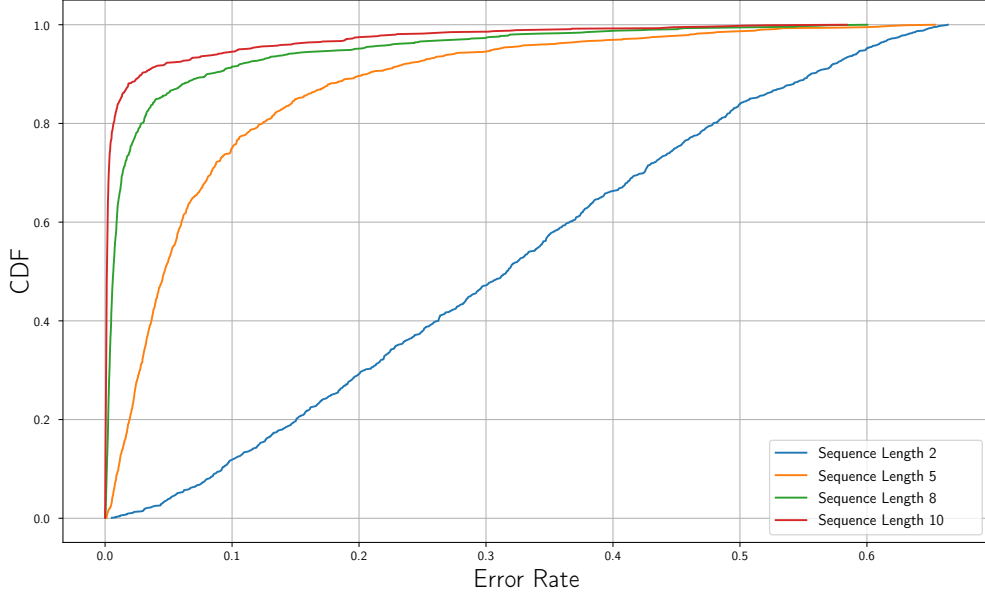


Figure 1: Cumulative distribution function of errors for different sequence lengths

This CDF is crucial as it provides an approximation of the probability distribution of errors, which we set as a goal in the abstract. By analyzing this figure, we gain insights into the distribution of errors across different sequence lengths.

I extracted the number of sets with errors less than 0.01, which we can call "good sets," those between 0.01 and 0.1, which we can call "okay sets," and those above 0.1, which we can call "bad sets." I introduced "okay" sets since they aren't massive statistical outliers like the bad sets but don't have high enough fidelity for precise calculations. The results are as follows: good sets: 846, okay sets: 105, and bad sets: 49. This indicates that the majority of resource gate pairs with 10-length sequences achieve a high fidelity in approximating the target gate. If we exclude the worst 10% (or 15%) of the sets—considering the 90th or 85th percentile of the distribution—we find that we can approximate the target gate with an error below 0.01. Therefore, with approximately 85% probability, we can achieve a fidelity of 98.7% or higher in approximating the target gate. Further investigation of these outliers may be warranted, but it is more beneficial to conduct such an analysis with a complete gate set.

8.2 Exponential fit of error

Having analyzed the cumulative distribution function, we proceed to examine how the error evolves with sequence length by looking at the mean of the lowest error values. There's no

mathematical guarantee that our error will decrease with each step, so at each step, I choose the lowest error so far; I'll expand on this later. By plotting the mean of these lowest error values so far (on the y-axis) against their corresponding sequence lengths (on the x-axis), we can visualize how the error evolves as the sequence length increases (see Fig. 2).

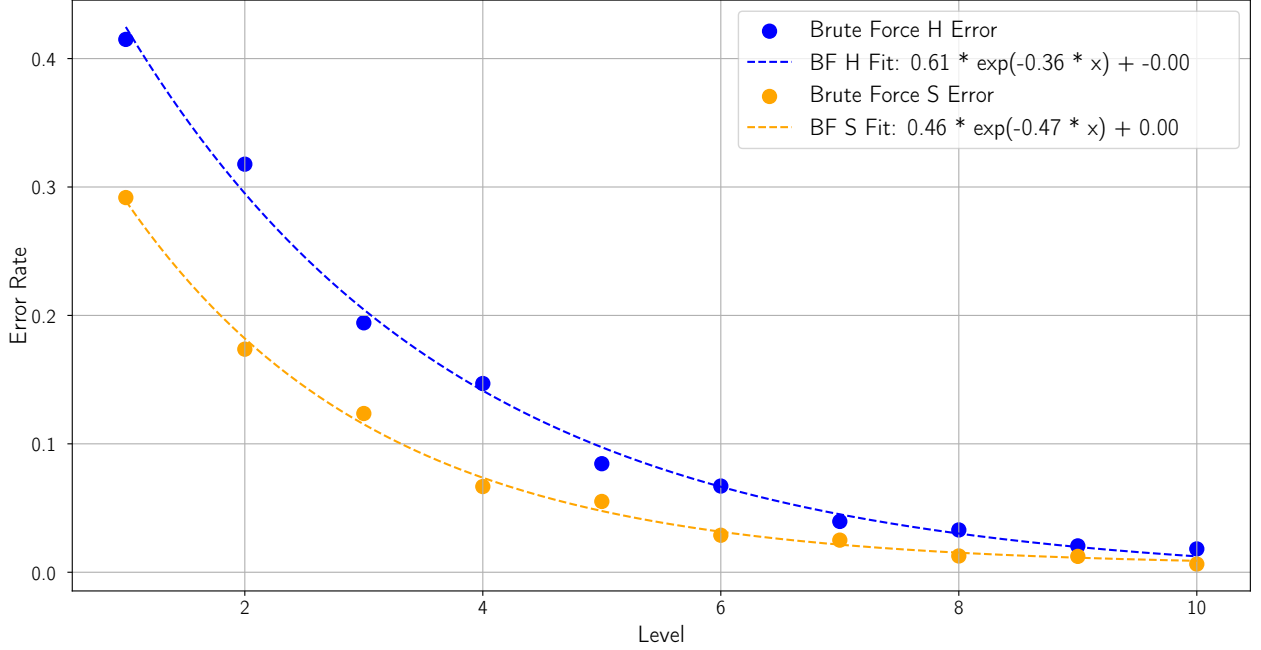


Figure 2: Mean error values plotted against sequence length for H and S gates.

We can clearly see an exponential decline as the sequence length increases. The decline is expected, but the exponential nature of it is noteworthy. We get the mean of the lowest error values in sequences of length 10, which is 0.011734156, but the mean error value of the length 10 sequences we obtain is 0.016856957. This is significantly larger than the mean lowest error for a sequence. Given that the function exponentially decays with sequence length, we would expect a much smaller difference. This is due to the fact that there is a significant number of cases where the lowest error value is not of the length 10 sequence.

As we can see in the following figure, which shows the number of lowest errors achieved by each sequence length plotted against sequence length (see Fig. 3):

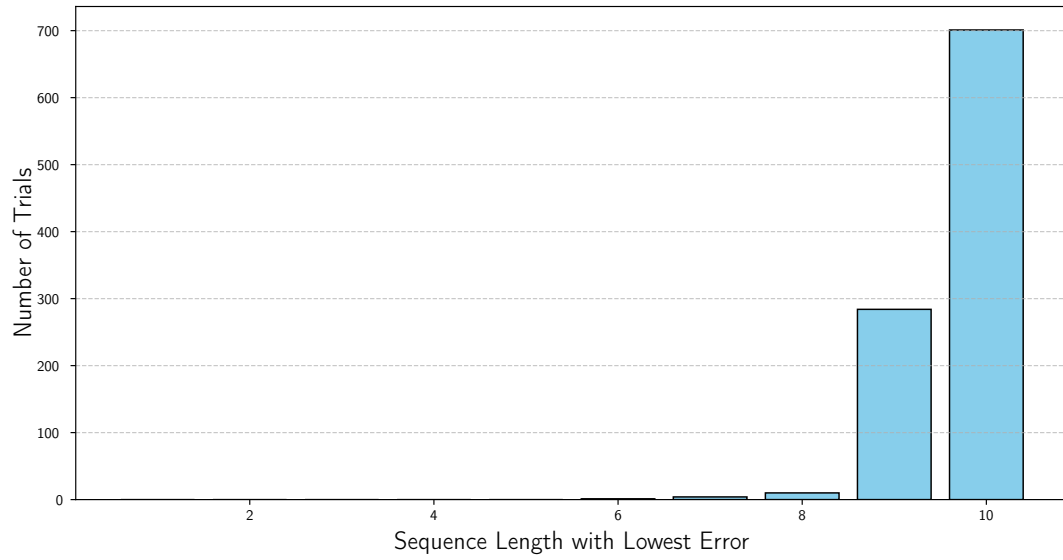


Figure 3: Distribution of sequence lengths achieving the lowest error

We observe that only around 700 out of the 1000 cases resulted in length 10 having the lowest error. Even from this, we may deduce worthwhile conclusions, but another interesting graph to look at is the following (see Fig. 4):

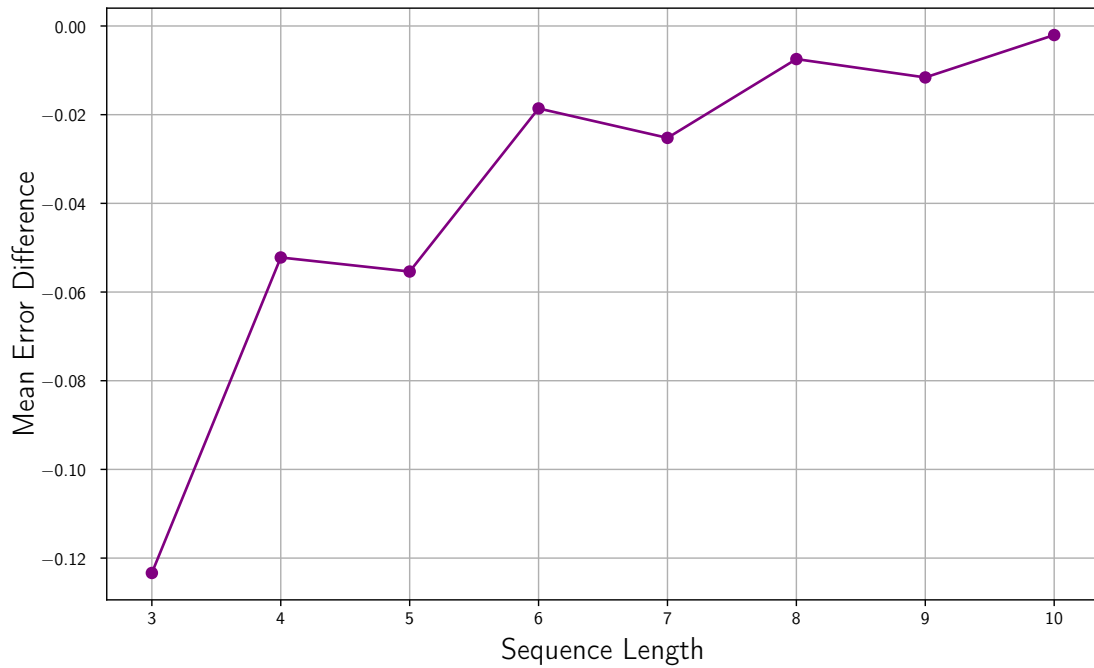


Figure 4: Mean error difference between consecutive sequence lengths

In this figure, we see the mean error difference between consecutive sequence lengths plotted

against the sequence length. The behavior of this graph aligns with our expectations since the mean difference is decreasing, just as in Fig. 2. An interesting observation is that the even steps have lower error reduction compared to odd ones. This is most likely due to the fact that, when we extend a k -length approximation by multiplying it with an additional element on the right (selected from four possible options), there is no guarantee that this added element will bring us closer to the target gate. However, by applying one more gate (extending the sequence by another step), the $k + 2$ -length approximation is at least as good as the k -length one, since the last two gates could act as inverses of each other, effectively neutralizing each other's impact on the sequence. Consequently, the set of $k + 2$ -length sequences inherently includes all k -length sequences, ensuring that adding two elements results in either an improved or at least non-worsening approximation compared to the original k -length sequence. This structure naturally leads to alternating patterns in error reduction, with even-length steps showing more reliable or consistent decreases in mean error compared to odd steps.

8.3 Error Outliers

In the data, we observe an interesting phenomenon that further modifies our analysis. There are clear, visible outliers where the error rate is 2 to 3 orders of magnitude higher than the average. Their distribution is varied, and more data is needed to draw definitive conclusions. These outliers interfere with our averages, so we need to examine them closely and account for their consequences in our statistics.

So far, we have analyzed the Hadamard gate approximation, yielding valuable insights. To gain a complete picture, we must also examine the generation of a universal gate set. In this case, we will use the H and S gates as our gate set.

8.4 Brute Force complete gate set

As we have outlined in the previous paragraph, to get a real assessment on the viability of this method, we must examine the generation of the complete gate set rather than a single gate.

From Fig. 4, we can infer that assessing the feasibility of this method requires more than just approximating a single gate, as the newly implemented S gate (Phase gate) appears much

simpler to approximate with this technique. This disparity likely stems from various factors, but the primary reason is likely the difference in rotation angles: the Hadamard gate applies a 180° rotation, whereas the Phase gate applies only a 90° rotation. Since these resource gates can be seen as rotations on the Bloch Sphere, gates with smaller rotation angles may be less effective at significantly altering the state, making them inherently easier to approximate. This factor, along with the method used to calculate fidelity, likely contributes to the observed ease in approximating the Phase gate. Further investigation could analyze how resource gates approximate unitaries with varying rotation angles, but such an approach would require a large number of gate approximations to control for other influencing factors. Given that approximating even a single gate is computationally intensive, scaling up this analysis would demand resources beyond my current capacity. Additionally, this line of investigation would likely explain the origin of ineffective gate sets rather than provide a practical solution to the issue at hand.

However, another worthwhile aspect to analyze is how a "bad" gate set for the Hadamard gate approximates other gates. From this set, we observe 40 cases where the Hadamard gate approximation is "bad" and 15 where the Phase gate is. Notably, only 6 cases show both gates exceeding the error threshold, suggesting that in most instances, either a 90 or 180 rotation can be reasonably approximated within a manageable sequence length. This is useful information since from this we can examine an approach to remedy this issue. Since there is a clear correlation between magnitudes of rotations and error values, we can explore how small changes localized to a single axis can change the error values.

8.5 Bloch Sphere visualization

But before we test out this hypothesis, we can do a simpler, visual test to notice irregularities and patterns in low or high error matrices. We can do this by plotting each resource gate onto the Bloch sphere, so we can observe the "direction" of its influence. To analyze unitary matrices as rotations on the Bloch sphere, we first normalize each matrix U by dividing it by the square root of its determinant, $U = U/\sqrt{\det(U)}$, ensuring it meets $SU(2)$ representation

requirements. The trace of U provides the rotation angle θ through

$$\theta = 2 \arccos \left(\frac{\text{Tr}(\text{trace}(U))}{2} \right). \quad (12)$$

To determine the rotation axis $\mathbf{n} = (n_x, n_y, n_z)$, we utilize the Pauli matrices σ_x , σ_y , and σ_z , with each component given by

$$n_x = \frac{\text{Tr}(\sigma_x U)}{2 \sin(\theta/2)}, \quad n_y = \frac{\text{Tr}(\sigma_y U)}{2 \sin(\theta/2)}, \quad n_z = \frac{\text{Tr}(\sigma_z U)}{2 \sin(\theta/2)}. \quad (13)$$

This vector $\mathbf{n}$ defines the axis around which the rotation occurs, while θ defines its magnitude.

For a rotation of standard basis vectors $\mathbf{v}_i$ (where $i = x, y, z$), we compute the rotated vectors $\mathbf{v}'_i$ using

$$\mathbf{v}'_i = \cos(\theta) \mathbf{v}_i + \sin(\theta) (\mathbf{n} \times \mathbf{v}_i) + (1 - \cos(\theta)) (\mathbf{n} \cdot \mathbf{v}_i) \mathbf{n}. \quad (14)$$

The following figure is this process applied to a resource gate set with an average error value of 0.0031. The orange and green vectors represent the rotation by U_a and U_b respectively, while the blue vector represents the rotation of the Hadamard gate for reference (see Fig. 5).

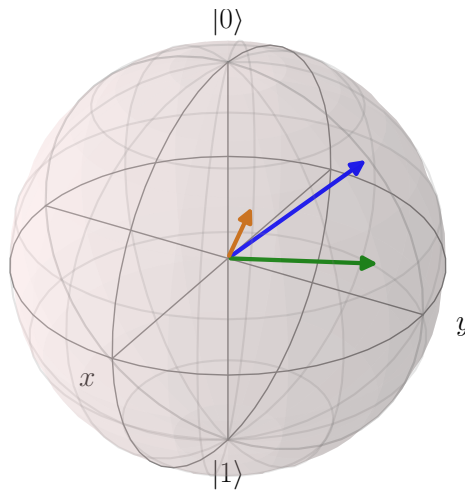


Figure 5: Bloch sphere visualization of rotations by U_a , U_b , and the Hadamard gate.

We aren't particularly interested in this exact figure, as it is not an edge case; it is used

here solely to demonstrate the method of visualization. .

The following four figures display the plots of the lowest error resource gate set, viewed respectively from the side, the x-axis, y-axis, and z-axis. The error is: $1.88623192809434 \times 10^{-6}$ (see 6;

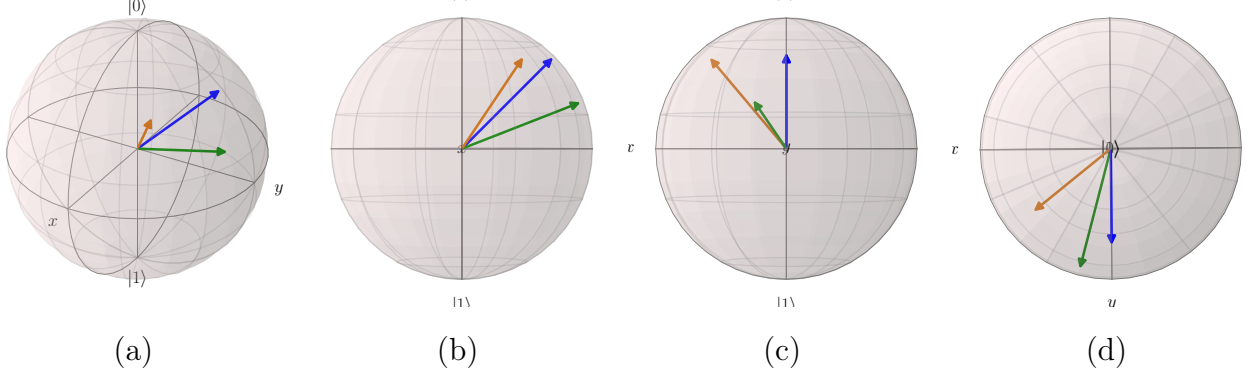


Figure 6: Visualization of the lowest error resource gate set viewed from different angles: (a) Side view, (b) View from x-axis, (c) View from y-axis, and (d) View from z-axis.

As mentioned before, the rotation vectors alone aren't enough to describe a gate operation; they have to be accompanied by a rotation angle, which describes the magnitude of the rotation. For this set, these values are: 79.07° and 113.13° for A and B respectively. For reference, the Hadamard rotates 180° , meaning there is no obvious reason as to why these specific rotations have such low error.

The following figure is laid out just like the one beforehand, but rather than the lowest error, this set created the highest error values see Figs. 11

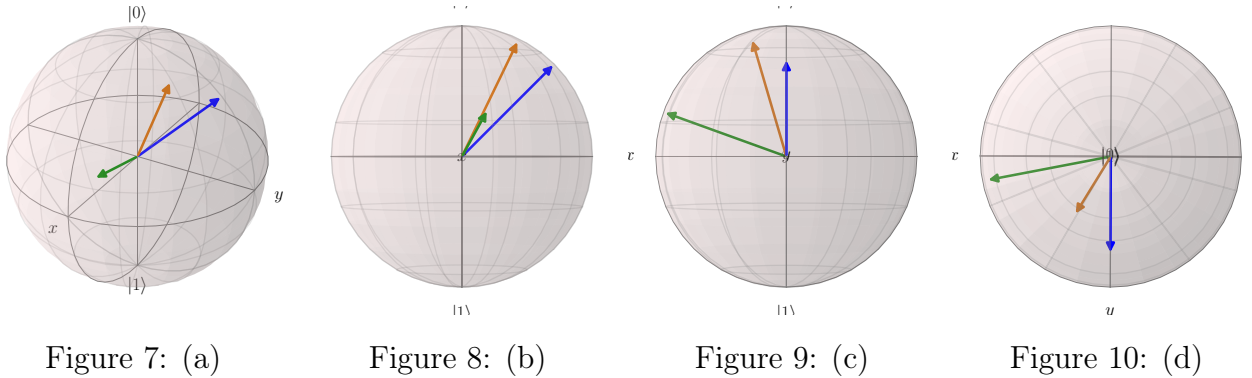


Figure 11: Visualization of the highest error resource gate set viewed from different angles: (a) Side view, (b) View from x-axis, (c) View from y-axis, and (d) View from z-axis.

Unfortunately, once again we see no obvious indication as to why this particular set has such high error values. To try and get a wider assessment, I logged the 30 highest and 30

lowest error matrices with this process and ran some statistical analysis. The box plots on the left (see Fig. 12) display the distribution of the X, Y, and Z components of the rotation axes, comparing the values between the high error and low error sets to show the spread and median of each component in both groups. The histogram on the right illustrates the frequency of rotation angles (in radians) for the high and low error sets, allowing a visual comparison of how often each angle range appears in each error group (see Fig. 12).

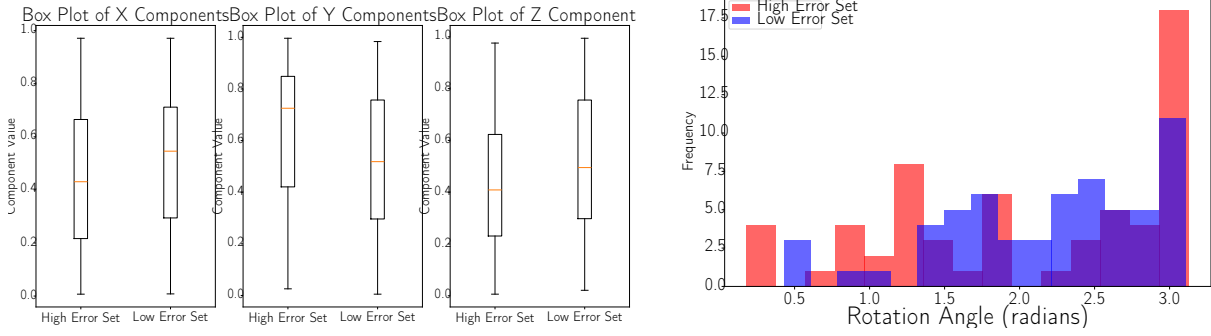


Figure 12: Visual comparison of angle ranges in each error group

As we can see, there is basically no discernible pattern among these values, examine both the rotation vector and the rotation angle we gain no new insight into the cause of these errors. Furthermore since this analysis was only done for the H gate other gates may have completely different distributions. This makes a statistical approach to determine the cause of the errors, impractical.

8.6 "Fixing" bad sets

Since the difference between a good and a bad set is seemingly very little, this gives us a possible solution: a slight "nudge" of this vector could induce a rapid decrease in error, as it may nudge it towards a good configuration. To test this, we will try all possible combinations of nudging each axis both forwards and backwards. Given that for each axis, we can nudge in two directions (forwards and backwards), there are 6 total choices. We can choose to act on 1, 2, or 3 axes, resulting in a total number of unique combinations given by:

$$\sum_{n=1}^3 \binom{3}{n} \times 2^n = 26 \quad (15)$$

Additionally, since we are interested in nature of the error decay, to asses this, we will need more data points. Thus, we will apply 5 nudges and create a fit to assess the trend. Although this approach could be further refined by varying both the direction and magnitude of each nudge, this would be computationally intensive and unlikely to yield additional valuable insights.

TO correctly measure this, we need to be able to make changes in the starting matrices in a way that isolates the change to a specific axis. We can accomplish this by applying a series of controlled rotations along each axis independently. For a given axis A , we define a rotation matrix $R_A(\theta)$ with rotation angle θ as follows:

- For the X -axis:

$$R_X(\theta) = \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & -i \sin\left(\frac{\theta}{2}\right) \\ -i \sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{pmatrix} \quad (16)$$

- For the Y -axis:

$$R_Y(\theta) = \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & -\sin\left(\frac{\theta}{2}\right) \\ \sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{pmatrix} \quad (17)$$

- For the Z -axis:

$$R_Z(\theta) = \begin{pmatrix} e^{-i\frac{\theta}{2}} & 0 \\ 0 & e^{i\frac{\theta}{2}} \end{pmatrix} \quad (18)$$

By applying $R_X(\theta)$, $R_Y(\theta)$, or $R_Z(\theta)$ to an initial matrix U , we can selectively introduce changes along each axis. Specifically, we compute the nudged matrix U' as:

$$U' = RUR^\dagger \quad (19)$$

where A represents the axis of rotation (either X , Y , or Z) and θ is the small nudge angle, in our case $\pi/360$. To study the error decay, we apply these rotations iteratively, observing how the error changes with each incremental nudge. This approach allows us to test whether the error decay follows an exponential trend by measuring the fidelity. Using this method I was able to find the 3 nudge sequences that resulted in the greatest error reduction, and took the parameters of said sequences, and applied 15 nudges to observe the function that arose. The following figure is the error plotted against the nudge sequence length (see Fig. 13).

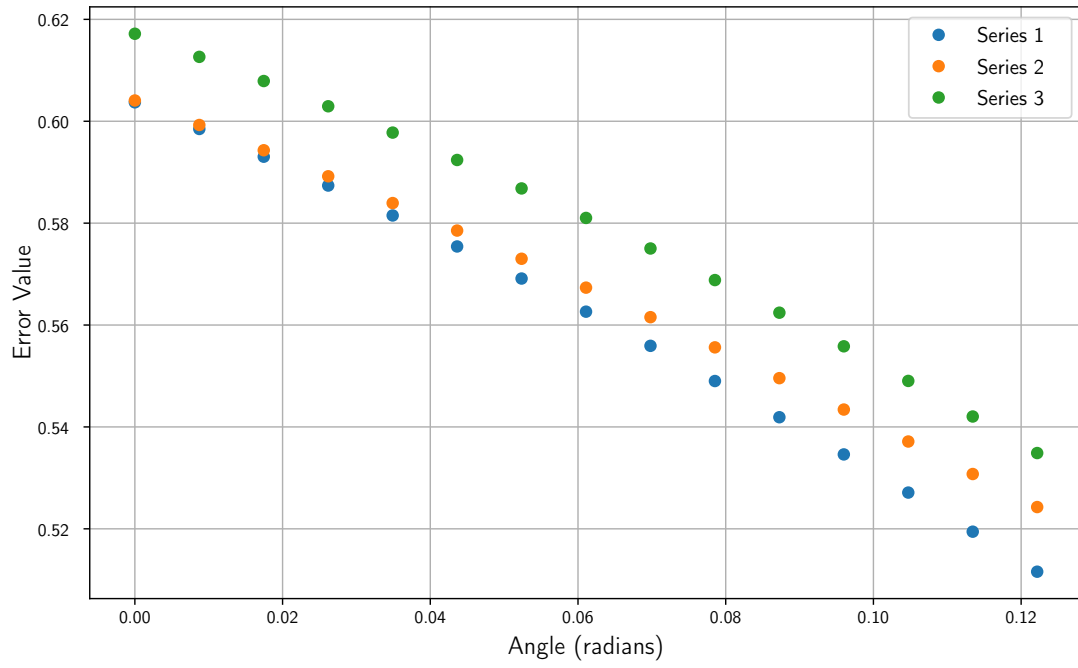


Figure 13: Error plotted against the nudge sequence length

The analysis reveals that the error decay is **predominantly linear** across the nudge sequence length. Further discussion of this result will be in the conclusion section.

9 The Solovay-Kitaev approach

The Solovay-Kitaev theorem asserts that if a set of quantum gates generates a dense subgroup of $SU(2)$, then any desired quantum gate can be approximated by a short sequence of these gates [28]. This theorem is fundamental to the feasibility of this paper, as it guarantees that for any target unitary, we can construct an efficient approximation with a sequence of gates of manageable length.

9.1 The Solovay-Kitaev Algorithm

The Solovay-Kitaev theorem provides a method, known as the Solovay-Kitaev algorithm, which can approximate any given unitary operation using a finite gate set to within any desired accuracy. For an overview of the algorithm [29], The general algorithm proceeds as follows:

- (i) Let U be the target unitary we wish to approximate. Begin by finding a coarse approximation V_0 . This initial approximation can be obtained through brute-force testing of small sequences from the gate set. The accuracy of V_0 at this stage is not crucial, as it will be refined in later steps.
- (ii) The refinement process is recursive, and each iteration refines the current approximation V_0 in three main steps:
 - First, compute the "deviation" or "error" matrix, defined as $\Delta = V_0^\dagger U$, which quantifies the difference between V_0 and U .
 - Next, construct two matrices, A and B , from sequences of gates in the set, such that the commutator $[A, B]$ closely approximates Δ . The commutator $[A, B]$ measures the degree to which the matrices A and B fail to commute, defined as $[A, B] = ABA^{-1}B^{-1}$. If A and B commute, then $[A, B] = I$.
 - To find suitable matrices A and B that produce a commutator approximating Δ , use brute-force search over short sequences from the gate set, or alternatively, use pre-calculated lookup tables.
- (iii) Once a suitable Δ with minimal error is found, update the approximation by setting $V_1 = V_0\Delta$. This recursive process is repeated, progressively refining the approximation

until the desired error threshold or sequence length is achieved.

9.2 The Solovay-Kitaev theorem applied

One major consequence of the theorem is the reduction in the error norm with increasing sequence length. The error norm measures how closely the constructed sequence of gates approximates the target gate. As the Solovay-Kitaev algorithm is applied recursively, the error decreases exponentially, ensuring that even for high-precision requirements, a relatively short sequence of gates can achieve the desired accuracy. Traditionally, the Solovay-Kitaev algorithm uses the operator norm as the error metric, but for our purposes, fidelity is a better fit. While the algorithm still works as intended with fidelity, it may be slightly less efficient since fidelity doesn't measure exactly what the algorithm is optimized for. To find Δ we employ a brute force search of all possible gate combinations of length 1 and 2 from our 4 gates, and choose the best approximation. This limits our sequence length since the commutator construction is made up of discrete sized building blocks. This results in the sequence length starting from 1 and 2 and increasing in increments of 8, so 9/10, 17/18 and so on. The rest of the algorithm applied to our case is standard and follows the process outlined in the previous section

9.3 Solovay-Kitaev Results

Using the previously outlined algorithm, 1000 trials were run alongside the brute-force method (the results of which will be discussed in the next section). The following figure displays the Cumulative Probability Function of the results Fig. 14.

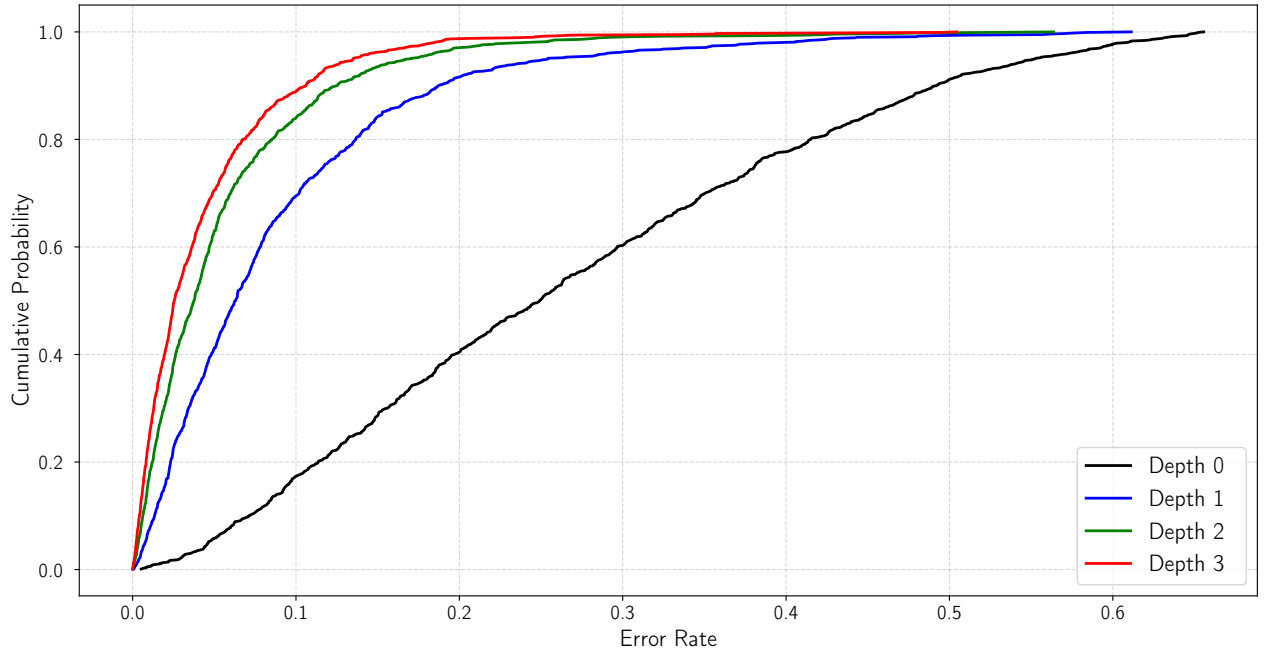


Figure 14: CDF of the Solovay-Kitaev results at each depth

We can see from the figure, that the SK algorithm consistently decreases the error, but keep in mind that each depth of recursion adds 8 more gates to the sequence.

We can also examine the computation time. Not a lot of conclusions can be drawn without any frame of reference (like in the next section), but for any future applications exact measurements may be useful. Here is the average computation time plotted for both the Hadamard and Phase gate (see Fig. 15).

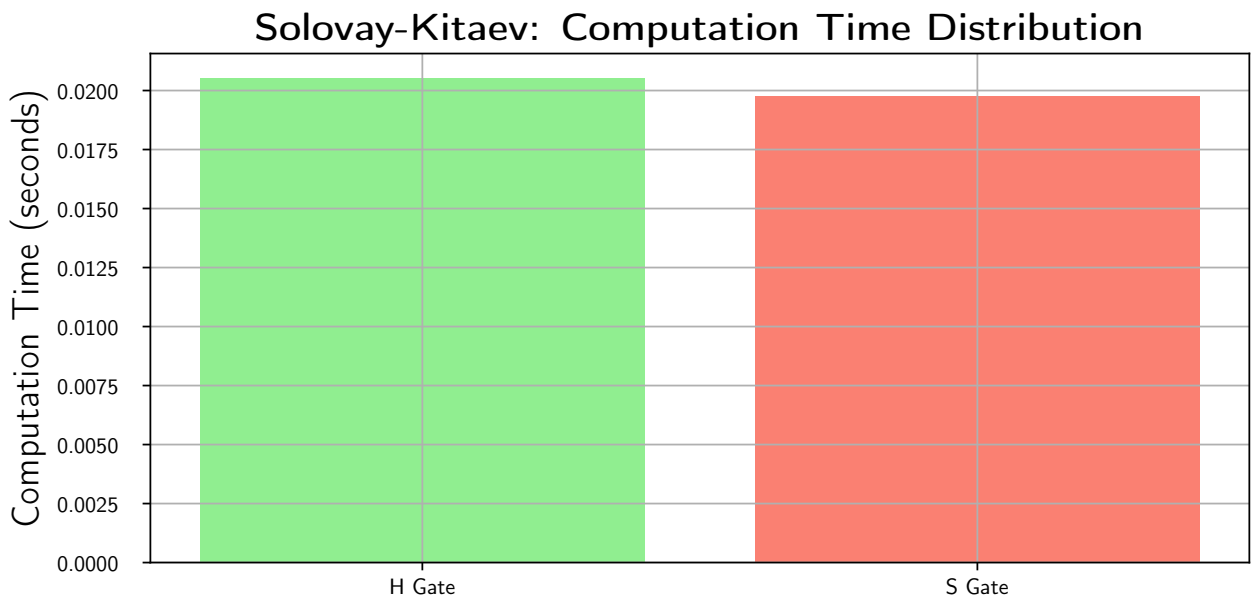


Figure 15: Average computation time for Hadamard and Phase gates

The exact values are 0.02054 seconds for the Hadamard gate and 0.01976 seconds for the Phase gate.

10 Solovay-Kitaev and Brute Force comparison

Using the previously outlined algorithm, we ran 1000 trials using the same resource gate sets as in the brute-force approach, ensuring a fair comparison between the methods. The following figure represents the two most important aspects of this approach: the error decay for approximating the Hadamard and Phase gates.

The figure below shows the mean error value for Hadamard and Phase approximations at each sequence length, plotted against sequence length. For the brute-force approach, an exponential fit is used, while for the Solovay-Kitaev algorithm, the fit follows the scaling dictated by the Solovay-Kitaev theorem. This approach allows for a direct comparison of both methods on the same plot, making it easier to assess the relative efficiency of each (see Fig. 16).

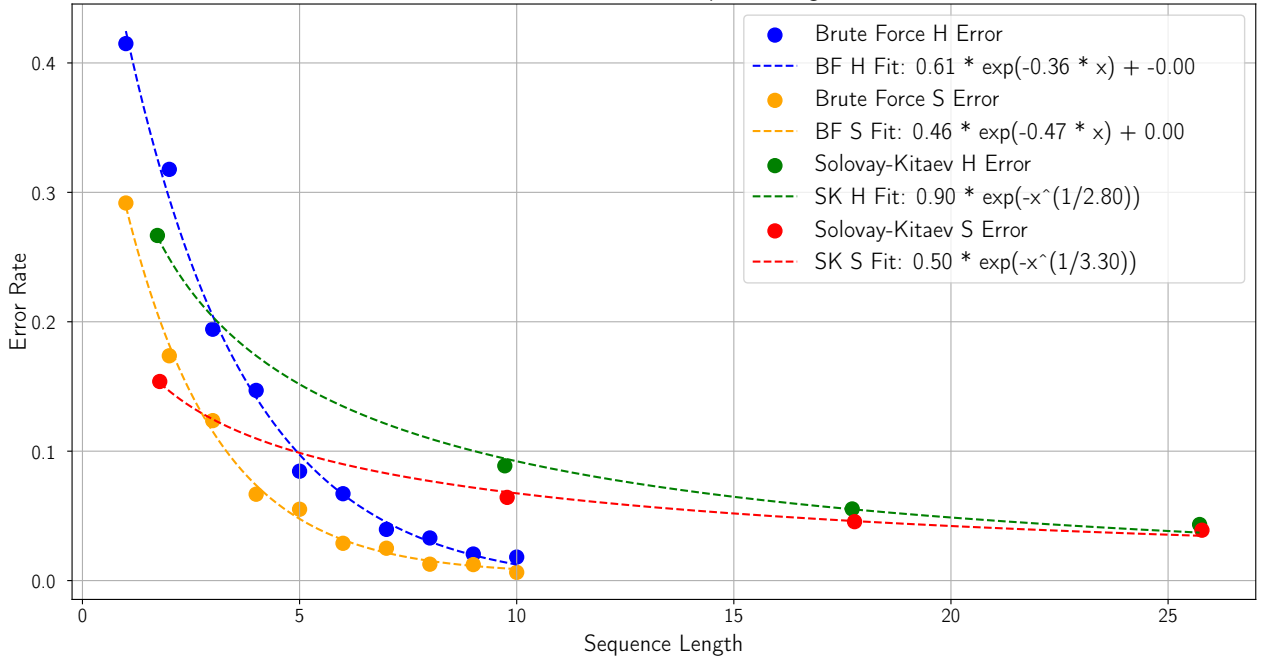


Figure 16: Mean error for Hadamard and Phase approximations at each sequence length, comparing brute-force (exponential fit) and Solovay-Kitaev (theorem-based fit) methods.

Upon examining the figure, one might initially think there could be an issue with the plotting due to the data points of the SK algorithm not aligning precisely with integers on the x-axis. However, this is a result of the SK algorithm's approximation process. During

either the initial rough approximation or the commutator approximation process, cases may arise where a shorter sequence achieves higher fidelity than the longest sequence, causing the algorithm to continue with a shorter sequence. To accommodate this, the x-axis values are represented as averages of sequence lengths at each depth of recursion.

The results align with our expectations, showing a decay in error consistent with the Solovay-Kitaev theorem. Additionally, these results reinforce our hypothesis that the magnitude of rotation influences how well specific gates can be approximated. Notably, the Phase gate appears to be easier to approximate, as evidenced by its error decay rate relative to the other gates.

We observe that the brute-force search provides a more rapid decay of error, as expected since it always finds the lowest error sequence by definition. The Solovay-Kitaev algorithm, while initially offering rapid error decay, levels out more quickly than the brute-force approach, this can also be seen in Fig. 14). as well. This behavior indicates that, to reach an acceptable error level for quantum computation, the sequence lengths for the Solovay-Kitaev method would need to be quite long, typically ranging from 30 to 50, depending on the algorithm parameters and the desired error margin.

This comparative analysis highlights both the efficiency and limitations of the Solovay-Kitaev algorithm relative to brute-force search, particularly when considering practical implementations for quantum gate approximations. However, there is one major advantage of SK compared to the brute-force is not visible on this graph, it is the computation time. The following figure charts the computation time logarithmically on the y-axis (see Fig. 17).

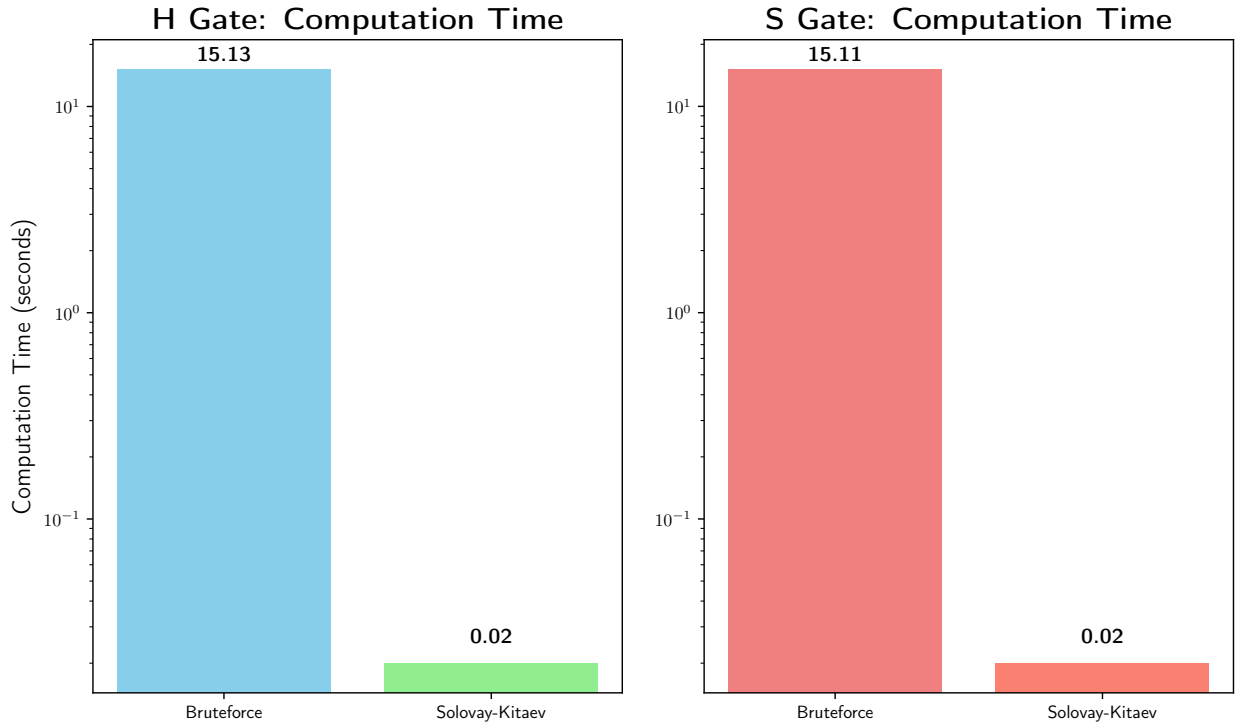


Figure 17: Computation time plotted logarithmically on the y-axis

As we can see, the brute-force computation is orders of magnitude greater than the SK-one. The fact that it's greater is unsurprising, it's the magnitude that is unexpected. Considering this fact we can clearly see the advantages and disadvantages provided by both approaches. If we value accuracy we should use brute force, but if time is of the essence, the SK-algorithm is our best bet. Since both approaches use the same building blocks, namely matrix multiplication and fidelity measurement, we can use these two as rough estimates for the amount of operations each approach requires, so depending on the nature of the physical realization, this could be an influential parameter.

11 Conclusion

In this work, we investigated the feasibility of implementing stable and efficient quantum gate operations using sequences of SOC-driven operations, focusing on shuttling-based holonomic quantum computation in spin qubits. Our primary goal was to determine how effectively random unitary transformations—induced by adiabatically transporting spin qubits through loops in quantum dots—could be combined to approximate specific target gates, such as the Hadamard and Phase gates.

We explored two main approaches: a brute-force method and the Solovay-Kitaev (SK) algorithm. The brute-force approach exhaustively searches all possible sequences of the available resource gates to find the sequence that best approximates the target gate. While this method guarantees the lowest possible error for a given sequence length, it becomes computationally infeasible as the sequence length increases due to the exponential growth in the number of possible sequences.

Our results with the brute-force method showed that the error in approximating the target gate decreases exponentially with the sequence length. Specifically, for sequences up to length 10, we observed that approximately 85% of the randomly generated resource gate sets could approximate the Hadamard gate with an error below 0.01, corresponding to a fidelity of 98.7% or higher. However, there were notable outliers—about 15% of the gate sets—that failed to achieve low error rates even with longer sequences. Attempts to "fix" these bad sets by nudging the rotation axes showed only linear error decay, suggesting limited effectiveness of this corrective approach.

The Solovay-Kitaev algorithm offers a more efficient alternative by recursively constructing approximations of the target gate using commutators of the available resource gates. This method significantly reduces computation time compared to the brute-force approach. Our implementation of the SK algorithm showed that it could achieve acceptable error rates with much shorter computation times, although it generally required longer sequences to reach the same error thresholds as the brute-force method. The error decay observed with the SK algorithm aligned with the theoretical expectations, reinforcing its validity for practical applications.

A key observation from our study is the influence of the target gate's properties on the ease of approximation. Gates involving smaller rotation angles, such as the Phase gate, were consistently easier to approximate than those involving larger rotation angles like the Hadamard gate. This suggests that the inherent geometric characteristics of the target gate play a significant role in the effectiveness of both approximation methods.

Furthermore, our findings support the initial motivation of expecting better qubits in zero external magnetic fields. Operating in such conditions leverages the inherent stability provided by spin-orbit coupling without the need for external magnetic field, which can introduce additional sources of decoherence. The ability to achieve high-fidelity gate operations through both brute-force and Solovay-Kitaev methods indicates that spin qubits in zero-field environments are promising candidates for scalable and robust quantum computing. Near-future research may focus on determining the exact cause of the "bad" gates and how to correct them. An interesting follow-up project we plan to pursue is to propose experiments that reveal the decoherence dynamics of spin qubits in zero magnetic field. At finite magnetic field, this is done using the Ramsey experiment and quantum state tomography, enabled by resonant qubit control. At zero magnetic field, resonant control does not work, but it could be replaced by the sequences of holonomic gates we studied in this paper.

References

- [1] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, October 1997.
- [2] Katherine L. Brown, William J. Munro, and Vivien M. Kendon. Using quantum computers for quantum simulation. *Entropy*, 12(11):2268–2307, November 2010.
- [3] Akshaya Jayashankar and Prabha Mandayam. Quantum error correction: Noise-adapted techniques and applications, 2022.
- [4] C. Ryan-Anderson, J. G. Bohnet, K. Lee, D. Gresh, A. Hankin, J. P. Gaebler, D. Francois, A. Chernoguzov, D. Lucchetti, N. C. Brown, T. M. Gatterman, S. K. Halit, K. Gilmore, J. A. Gerber, B. Neyenhuis, D. Hayes, and R. P. Stutz. Realization of real-time fault-tolerant quantum error correction. *Phys. Rev. X*, 11:041058, Dec 2021.
- [5] Morten Kjaergaard, Mollie E. Schwartz, Jochen Braumüller, Philip Krantz, Joel I.-J. Wang, Simon Gustavsson, and William D. Oliver. Superconducting qubits: Current state of play. *Annual Review of Condensed Matter Physics*, 11(Volume 11, 2020):369–395, 2020.
- [6] Colin D. Bruzewicz, John Chiaverini, Robert McConnell, and Jeremy M. Sage. Trapped-ion quantum computing: Progress and challenges. *Applied Physics Reviews*, 6(2), May 2019.
- [7] Stephan G. J. Philips, Mateusz T. Mądzik, Sergey V. Amitonov, Sander L. de Snoo, Maximilian Russ, Nima Kalhor, Christian Volk, William I. L. Lawrie, Delphine Brousse, Larysa Tryputen, Brian Paquelet Wuetz, Amir Sammak, Menno Veldhorst, Giordano Scappucci, and Lieven M. K. Vandersypen. Universal control of a six-qubit quantum processor in silicon. *Nature*, 609(7929):919–924, 9 2022.
- [8] Nico W. Hendrickx, William I.L. Lawrie, Maximilian Russ, Floor van Riggelen, Sander L. de Snoo, Raymond N. Schouten, Amir Sammak, Giordano Scappucci, and Menno Veldhorst. A four-qubit germanium quantum processor. *Nature*, 591(7851):580–585, 3 2021.
- [9] N. W. Hendrickx, L. Massai, M. Mergenthaler, F. J. Schupp, S. Paredes, S. W. Bedell, G. Salis, and A. Fuhrer. Sweet-spot operation of a germanium hole spin qubit with highly

- anisotropic noise sensitivity. *Nature Materials*, 23(7):920–927, 7 2024.
- [10] Daniel Loss and David P. DiVincenzo. Quantum computation with quantum dots. *Physical Review A*, 57(1):120–126, January 1998.
 - [11] A.J. Sigillito, R.M. Jock, A.M. Tyryshkin, J.W. Beeman, E.E. Haller, K.M. Itoh, and S.A. Lyon. Electron spin coherence of shallow donors in natural and isotopically enriched germanium. *Physical Review Letters*, 115(24), December 2015.
 - [12] F. H. L. Koppens, C. Buizert, K. J. Tielrooij, I. T. Vink, K. C. Nowack, T. Meunier, L. P. Kouwenhoven, and L. M. K. Vandersypen. Driven coherent oscillations of a single electron spin in a quantum dot. *Nature*, 442(7104):766–771, August 2006.
 - [13] Denis V. Bulaev and Daniel Loss. Electric dipole spin resonance for heavy holes in quantum dots. *Physical Review Letters*, 98(9), February 2007.
 - [14] Giordano Scappucci, Christoph Kloeffel, Floris A. Zwanenburg, Daniel Loss, Maksym Myronov, Jian-Jun Zhang, Silvano De Franceschi, Georgios Katsaros, and Menno Veldhorst. The germanium quantum information route. *Nature Reviews Materials*, 6(10):926–943, December 2020.
 - [15] Nico W. Hendrickx, William I. L. Lawrie, Maximilian Russ, Floor van Riggelen, Sander L. de Snoo, Raymond N. Schouten, Amir Sammak, Giordano Scappucci, and Menno Veldhorst. A four-qubit germanium quantum processor. *Nature*, 591(7851):580–585, March 2021.
 - [16] Zhanning Wang, Elizabeth Marcellina, Alex. R. Hamilton, James H. Cullen, Sven Rogge, Joe Salfi, and Dimitrie Culcer. Optimal operation points for ultrafast, highly coherent ge hole spin-orbit qubits. *npj Quantum Information*, 7(1), April 2021.
 - [17] M. Maamache and Yahia Saadi. Quantal phase factors accompanying adiabatic changes in the case of continuous spectra. *Phys. Rev. A*, 78, 11 2008.
 - [18] Erik Sjöqvist. A new phase in quantum computation. *Physics Online Journal*, 1:35, November 2008.

- [19] Frank Wilczek and A. Zee. Appearance of gauge structure in simple dynamical systems. *Phys. Rev. Lett.*, 52:2111–2114, Jun 1984.
- [20] Paolo Zanardi and Mario Rasetti. Holonomic quantum computation. *Physics Letters A*, 264(2):94–99, 1999.
- [21] Hendrik Anthony Kramers. Théorie générale de la rotation paramagnétique dans les cristaux. *Proc. Amsterdam Acad.*, 33:959–972, 1930.
- [22] Vitaly N. Golovach, Massoud Borhani, and Daniel Loss. Holonomic quantum computation with electron spins in quantum dots. *Physical Review A*, 81(2), February 2010.
- [23] Baksa Kolok and András Pályi. Protocols to measure the non-abelian berry phase by pumping a spin qubit through a quantum-dot loop. *Phys. Rev. B*, 109:045438, Jan 2024.
- [24] Chien-An Wang, Valentin John, Hanifa Tidjani, Cécile X. Yu, Alexander S. Ivlev, Corentin Déprez, Floor van Riggelen-Doelman, Benjamin D. Woods, Nico W. Hendrickx, William I. L. Lawrie, Lucas E. A. Stehouwer, Stefan D. Oosterhout, Amir Sammak, Mark Friesen, Giordano Scappucci, Sander L. de Snoo, Maximilian Rimbach-Russ, Francesco Borsoi, and Menno Veldhorst. Operating semiconductor quantum processors with hopping spins. *Science*, 385(6707):447–452, 2024.
- [25] Francesco Mezzadri. How to generate random matrices from the classical compact groups, 2007.
- [26] Line Hjørtshøj Pedersen, Niels Martin Møller, and Klaus Mølmer. Fidelity of quantum operations. *Physics Letters A*, 367(1–2):47–51, July 2007.
- [27] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.
- [28] Christopher M. Dawson and Michael A. Nielsen. The Solovay-Kitaev algorithm. *Quantum Inf. Comput.*, 6:81–95, 2005.
- [29] A Yu Kitaev. Quantum computations: algorithms and error correction. *Russian Mathematical Surveys*, 52(6):1191, dec 1997.