

# Approximating quantum many-body systems using quantum circuits

Luca Ion, Nicholas Synesi  
School of Physics and Astronomy  
University of Nottingham

May 2023

### Abstract

Many-body physics benefits from the power of quantum computing. We are interested in building quantum circuits that can approximate the ground states of several Hamiltonian models, some of which cannot be solved exactly. We devise an optimization algorithm that will compute the necessary gates that make up the quantum circuit that will approximate the ground state. Once we have the approximated states, we compute different correlation functions on both the exact ground state and the quantum circuit approximation in order to learn more about the quantum circuits. All the computations in this paper are carried out using Python and NumPy. In addition, the use of Nottingham's HPC was used for the more computationally demanding tasks. We find that measuring a variety of correlation functions on our approximated state gives us a good approximation of the state we are trying to approximate. Moreover, with the help of these measures, we get more insights into the quantum circuit structure.

## Contents

<b>1</b>	<b>Introduction</b>	<b>4</b>
<b>2</b>	<b>The Quantum Ising model and Exact Diagonalization</b>	<b>6</b>
2.1	The Ising Model . . . . .	6
2.1.1	Ising Model properties . . . . .	6
2.1.2	Non-integrable Ising model . . . . .	7
2.2	Exact Diagonalization . . . . .	7
2.3	Limitations of Exact Diagonalization . . . . .	8
<b>3</b>	<b>Introduction to quantum computing</b>	<b>9</b>
3.1	Qubits . . . . .	9
3.2	Multi particle systems . . . . .	9
3.3	Quantum gates . . . . .	10
3.4	Quantum circuits . . . . .	11
3.5	Tensor representation . . . . .	12
3.6	The target state . . . . .	13
<b>4</b>	<b>Brick wall circuit</b>	<b>14</b>
4.1	Gate initialization . . . . .	14
<b>5</b>	<b>Optimization process</b>	<b>16</b>
5.1	Polar optimization . . . . .	17
5.2	Gradient descent optimization . . . . .	19
5.3	Polar Vs gradient descent optimization . . . . .	20
5.4	Final results . . . . .	20
<b>6</b>	<b>State Properties</b>	<b>23</b>
6.1	Energy . . . . .	23
6.2	Quantum Entropy . . . . .	23
6.2.1	von Neumann entropy . . . . .	23
6.2.2	Rényi entropy . . . . .	25
6.3	Mutual Information . . . . .	26

<b>7 Discussion</b>	<b>30</b>
<b>A <math>\eta</math> tuning schemes</b>	<b>32</b>
<b>B Increasing overlap</b>	<b>32</b>

# 1 Introduction

Quantum mechanics has revolutionized science and opened the door to quantum information and computing. The study of many-body physics aims to predict the behaviour of interacting quantum particles, which is challenging due to the difficulty of describing vast systems with many quantum sites and interactions. Quantum computers provide a promising solution and are expected to surpass the classical digital computers we have today [1]. Current Noisy Intermediate Scale Quantum (NISQ) devices contain 50-100 qubits, enabling access to a greater number of quantum gate operations [1]. NISQ presents many new opportunities in quantum physics as well as quantum cryptography [2] and quantum deep learning [3]. In order to develop new quantum technologies, one must first have a realization of a quantum computer. Many proposals using different physical systems have been provided: qubits via superconducting materials [4], [5], quantum computation using Rydberg atoms [6], [7], photonic quantum computing [8], [9], and silicon based quantum computers [10], [11].

The complexity of quantum many-body problems lies within the fact that the dimension of the Hilbert space grows exponentially as  $d^N$ , where  $d$  is the number of degrees of freedom of each individual particle, and  $N$  is the total number of particles. Throughout this paper, we will consider two-level systems so  $d = 2$ .

One such attempt to solve quantum many-body problems in equilibrium is through a numerical method called the quantum Monte Carlo (QMC) method [12]. In this approach, the partition function of the interacting system is cast into an integral over each of the configurations in a chosen basis, and through sampling it could then cover each of the sites [13]. Then, QMC can approximate the many-body wave function and calculate various properties of the system, such as the energy and correlation functions. In practice, the effectiveness of QMC methods is mitigated by the infamous "sign problem" [14]. In the classical case, the Hamiltonian is always diagonal, leading to positive weightings. In QMC however, the Hamiltonians have complex off-diagonal matrix elements, which can potentially give negative weightings [15]. This is an issue due to the fact that this requires a quasi-probability distribution to be solved, as opposed to a non-negative probability distribution. The sign problem is particularly severe for strongly correlated systems, where the sign of the wave function frequently changes, leading to a cancellation of positive and negative contributions in the Monte Carlo sampling. This makes it exponentially difficult to obtain accurate results from large-scale quantum systems. Other classical methods used to study quantum many-body problems are Matrix Product States (MPS) [16] and Density-Matrix Renormalization Groups (DMRG) [17]. An MPS allows the quantum state to be decomposed into a series of matrices, and so truncating the matrices means one can simulate larger systems. However, truncation limits the amount of entanglement that can be present in the system. This means MPS are good for systems with low entanglement, such as ground states in one dimension, however they become far less reliable when there are larger amounts of entanglement present. DMRG is one of the most powerful algorithms for studying the ground states of a one-dimensional quantum lattice [18]. The success of DMRG comes from the fact that ground states themselves are not very entangled and can therefore be efficiently written in terms of MPS [19]. This then allows us to reduce the number of parameters from exponential to polynomial. This means systems with many particles become accessible for numerical approximations. DMRG was also extended to simulate real-time evolution, however, the entanglement of pure bipartite states increases linearly with time, leading to a fast exponential growth of the computing costs [19].

In an effort to overcome the problems with classical methods, quantum methods for solv-

ing quantum many-body problems were introduced. On a quantum computer, a variational quantum eigensolver (VQE) can be used to approximate the ground state of a many-body system[20]. In the classical approach, the amount of entanglement present is directly related to the difficulty of representing the state. However, for a quantum circuit, a state with a high amount of entanglement can have a low complexity, where complexity is defined as the number of resources needed to perform the computation[21]. The VQE was used by IBM to find the ground state of molecules in quantum chemistry[22], and was also implemented on a photonic quantum processing unit to solve problems involving small molecules[23]. Another algorithm based on quantum computing is the quantum approximation optimization algorithm (QAOA). This was used to approximate solutions to combinatorial optimization problems[24], which was found to outperform other methods[25]. Recently, quantum tensor networks (TN) were proposed, where TNs such as MPSs and multi-scale entanglement renormalization ansatz (MERA) were promoted to their quantum circuit counterparts[26]. It was found that the quantum circuit tensor networks were more expressive than their classical counterparts for certain physical models[27]. The utility of quantum circuit tensor networks has also been demonstrated by their implementation on NISQ machines for quantum simulations achieved through parallelism[28]. The translation of tensor networks into the quantum realm poses the question of whether all classical simulations of quantum systems can be translated in this way, and what the advantages of doing this would be. The quantum methods described tend to give a speed up in performances compared to the classical methods of solving the same problem. In the case of many-body quantum systems, this is partly due to the fact that quantum circuits that run on quantum computers deal with a lot of entanglement fairly easily, which is beneficial for solving such systems numerically. In the classical approach, a high amount of entanglement means these methods are less effective. However, not much is yet known about the limitations of quantum algorithms via quantum circuits. Our aim is to perform as many measurements on an approximate state generated from a quantum circuit in an effort to find any such limitations.

For our approach, we present the use of quantum circuits using Python, and the package NumPy to create a quantum circuit simulator to approximate the ground states of a many-body system. We will consider different models; the Ising model with a transverse field (integrable), and with both a transverse and longitudinal field (non-integrable).

The structure of this paper is as follows. In Chapter 2, we introduce a classical method for finding the ground states of the quantum Ising model called Exact Diagonalization. We will evaluate this method and assess why its use to find the ground states of larger systems is not possible. In Chapter 3, we then introduce quantum computing and how quantum circuits can be used to instead approximate the ground state of the quantum Ising model, and we will detail our algorithm for doing so in Chapter 4. Chapter 5 is the topic of an investigation of how effective our approximation is compared to the exact ground state by taking the measure of the so-called 'overlap' will be done. After coming to a conclusion on whether we can approximate the state to a high level of accuracy, we will look at other state properties in Chapter 6. These include entanglement entropy, mutual information, and the energy of the state. Using these state properties is another way to test how our approximation measures these quantum properties. Chapter 7, our final section, will serve to summarize our results to assess whether quantum circuits can indeed accurately approximate the ground state of the Ising model.

## 2 The Quantum Ising model and Exact Diagonalization

There are methods to find ground states which we use today on our typical devices, which we refer to as 'classical' devices. These methods have proved accurate but tend to be limited in various ways when it comes to many-body systems. In this paper, we look at the quantum Ising model. Our aim is to approximate the ground states in this model and compare them to exact ground states generated using a classical approach. The ground state is the state of the lowest possible energy. It is important to examine the ground states as they represent a reference point for looking at higher energy states in the system. The method we will be using is exact diagonalization, and it will become our reference point when comparing the approximation of ground states between classical and quantum devices.

### 2.1 The Ising Model

The Ising model is widely studied in the field of statistical mechanics and has been used to model phenomena such as ferromagnetism. In this case, we will be looking at the quantum version of the model, known as the transverse field Ising model. This model is comprised of a 2-level quantum system corresponding to either spin up or spin down, arranged on a 1-dimensional lattice. The only interaction between sites is the nearest-neighbour interaction, so a single spin can only influence its two neighbours. The Hamiltonian is as follows

$$H = J \sum_{i=1}^N X_i X_{i+1} + h \sum_{i=1}^N Z_i, \quad (2.1)$$

where  $N$  is the total number of sites,  $J$ ,  $h$ , and  $g$  are tuning parameters, and  $X$ ,  $Z$  correspond to the Pauli matrices

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

#### 2.1.1 Ising Model properties

One of the reasons that the Ising model was chosen to be the Hamiltonian in this project is due to its simplicity. However, it also has the ability to capture the essence of phase transitions, which makes it a useful tool for studying many physical phenomena. A unique feature of the Ising model is that it is integrable, meaning the ground states can be solved for exactly [29]. In equation (2.1), the tuning parameter  $h$  plays a key role in when a phase transition happens. The parameter  $h$  controls the strength of the transverse field. When  $|h/J|$  is greater than 1, or  $|h|$  is greater than  $|J|$ , the system is in the state of a paramagnet. This means the spins are aligned in the direction of the applied magnetic field. For  $h > 0$ , this is the  $-z$  direction. When  $|h|$  is less than  $|J|$  the system behaves like a ferromagnet, where the spins are all aligned in the  $x$  direction, where the spins are either all spin up, or spin down, giving rise to a degenerate ground state. It is when  $h$  is equal to  $J$ , that the system undergoes a quantum phase transition. When we test our approximation of the ground state, we will examine how well we can approximate this state at the critical point at  $h = J$ , in comparison to a region where  $h$  is greater than 1.

### 2.1.2 Non-integrable Ising model

As well as studying the transverse field Ising model, we can also study an Ising model with both a transverse and longitudinal field, which has a Hamiltonian of the form

$$H = J \sum_{i=1}^N X_i X_{i+1} + h \sum_{i=1}^N Z_i + g \sum_{i=1}^N X_i. \quad (2.2)$$

We can see that for  $g = 0$ , it matches equation (2.1). Notably, when  $g \neq 0$ , the model is no longer integrable. This means the ground states can not be solved for exactly. They can only be approximated numerically or through methods such as perturbation theory. When we look to approximate the ground states, we will aim to build an approximation for both the integrable and non-integrable model to test both the effectiveness of our approximations, and whether the system being integrable or not leads to a better or worse approximation of the state. For future reference, we will label the three models. Model 1 will be the transverse field Ising model at the critical point, or when  $J, h = 1$  and  $g = 0$ . Model 2 will be  $J = 1, h = 1.5, g = 0$ . Finally, Model 3 represents the Ising model with both a transverse and longitudinal field, or when  $J, h, g = 1$ . These models are visualized in the following table

	Model 1	Model 2	Model 3
$J$	1	1	1
$h$	1	1.5	1
$g$	0	0	1

The correlation length is another concept in statistical mechanics that helps differentiate each of our 3 models generated from equation (2.2). It is the length scale for the spatial decay of correlations, and we denote it as  $\xi$ . As we approach the critical point, as in model 1, the correlation length diverges according to the power law

$$\xi = \left| \frac{h}{J} \right|^{-\nu}, \quad (2.3)$$

where  $h$  is our tuning parameter in the Ising model, and  $\nu$  is a critical exponent. This means that in model 1, fluctuations in the system are correlated over an infinitely long length scale.

For any non-critical point in the Ising model, the correlation length does not diverge. Instead, it is finite. This means that the correlation between sites decays exponentially with the length scales set by the parameter  $\xi$  with distance. Models 2 and 3 correspond to these non-critical points in the Ising model and in Chapter 6, we see how this affects the different correlation functions we measure.

## 2.2 Exact Diagonalization

Exact diagonalization (ED) is a powerful numerical method that enables us to study the ground state of quantum many-body systems by diagonalizing the Hamiltonian matrix. The eigenvalues and eigenvectors can then be calculated from this matrix. The eigenvalues represent the possible energies that the system can have, while the eigenvectors represent the corresponding states of the system with those energies. We can find the ground state by taking our diagonalized matrix produced using ED and finding the lowest eigenvalue, and the corresponding eigenvector will be the ground state of the system.

### 2.3 Limitations of Exact Diagonalization

Although exact diagonalization represents one of the most powerful classical computational methods for calculating ground states, it is key to note where its limit lies. The dimension of the Hilbert space scales exponentially with added sites. Consider the example of the Ising model, where each site can either be in spin up or spin down state. In a real ferromagnet, the number of sites would be extremely large, however as mentioned in section 2, ED requires the Hamiltonian matrix in order to diagonalize and find the ground state. A system with  $N$  sites has dimension  $2^N$ , and the Hamiltonian has size  $2^N \times 2^N$ . This means that as more sites are added, the memory requirements scale exponentially. The limit using this approach for our devices is roughly 20 qubits, with even the most powerful supercomputers being unable to handle the memory requirements for roughly 50 sites. To understand why this is the case, consider a system made up of 10 sites. In this case, the Hilbert space has dimension  $2^{10} = 1024$ , and the Hamiltonian has size  $1024 \times 1024$ , which is manageable. However, when we double the number of sites to 20, the Hamiltonian now has size  $2^{20} \times 2^{20}$ , which is a huge jump in memory requirements. There are ways to help mitigate this memory usage. Using a sparse matrix to represent the Hamiltonian is one such way, as these Hamiltonians contain mainly zeroes. However, this method does not eliminate the issue of memory requirements for more and more sites, it just helps to mitigate it. It is clear we must look for other solutions, and this is where we can introduce quantum computers as a new method to approximate ground states.



### 3 Introduction to quantum computing

Now we have looked at the classical approach to finding the ground state of a Hamiltonian, it is clear we must take another path in order to perform these calculations. The main limiting factor as to why we cannot simply find the ground states using ED was the exponentially large amount of computational power needed to construct such a matrix. We thus turn towards a different approach; quantum computers. Quantum computers have different behaviours to classical computing devices, and the following section covers all relevant information needed in order to understand how instead of directly calculating the ground state of a system, we approximate the state using quantum circuits.

#### 3.1 Qubits

The idea at the core of quantum computing is the quantum bit or *qubit*. The qubit is the quantum analogue of the classical bit, which can take a discrete value of 0 or 1. We can map a classical bit into a qubit as follows

$$0 \rightarrow |0\rangle, \quad 1 \rightarrow |1\rangle, \quad (3.1)$$

where the 0 and 1 in the ket brackets represent the qubits.

The key difference between a classical bit and a qubit is that a qubit is not limited by being just a 0 or a 1. It is a *continuous* variable, unlike its discrete, classical counterpart. The general form of a qubit is

$$\alpha |0\rangle + \beta |1\rangle, \quad (3.2)$$

where  $\alpha$  and  $\beta$  are complex numbers which satisfy

$$|\alpha| + |\beta| = 1. \quad (3.3)$$

We can also think of the qubit states in vector form. The states  $|0\rangle$  and  $|1\rangle$  have vector representation

$$|0\rangle \equiv \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad (3.4)$$

$$|1\rangle \equiv \begin{bmatrix} 0 \\ 1 \end{bmatrix}. \quad (3.5)$$

It follows that our general form of the qubit in (3.2) can be written as

$$|\psi\rangle \equiv \alpha |0\rangle + \beta |1\rangle \equiv \begin{bmatrix} \alpha \\ \beta \end{bmatrix}. \quad (3.6)$$

We call the states  $|0\rangle$  and  $|1\rangle$  our basis state.

#### 3.2 Multi particle systems

In this subsection we will present a brief overview of how to represent qubits for multiple particle systems; this is required since the models we will look at in section 2.1 are multi-particle systems.

$|0\rangle$  and  $|1\rangle$  are the basis states of a single particle, 2 dimensional Hilbert space  $\mathcal{H}$ . Now if there are  $N$  particles then the composite system is the tensor product of the individual single-particle Hilbert spaces, therefore we have

$$\mathcal{H} = \mathcal{H}_1 \otimes \mathcal{H}_2 \otimes \dots \otimes \mathcal{H}_N \quad (3.7)$$

It is a well-known fact from linear algebra that the basis set of such a composite system contains tensor product states comprised of the individual basis states of the single particle systems. Written as an equation the basis set is

$$\{|i_1\rangle \otimes |i_2\rangle \dots \otimes |i_N\rangle : i_1, i_2, \dots, i_N \in \{0, 1\}\} \quad (3.8)$$

It is now clear that the dimension of  $\mathcal{H}$  is  $2^N$ . We very often drop the  $\otimes$  symbol whenever working with multi-particle systems, therefore the following basis state (for a 2-particle system) is usually written as

$$|0\rangle \otimes |0\rangle \equiv |0\rangle |0\rangle \equiv |00\rangle \quad (3.9)$$

In the same way as a single particle qubit can be written in vector notation, a multi-particle state can also be written in vector form, for example, equation (3.9) can be written as

$$|00\rangle = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad (3.10)$$

This is useful since one can use matrix multiplication and vector arithmetic to manipulate the abstract notions, which in some instances can be easier to use than the abstract notations.

### 3.3 Quantum gates

We want to manipulate the state of our quantum system, so it is necessary to look at quantum gates, or *unitary operators*. To understand what they are, we can once again look at the classical analogue, a logic gate. One such example is the one-bit NOT gate, which has the following effect on the basis states:

Input	Output
0	1
1	0

We can see here that this gate simply flips the value of the input bit. Something important to note is that this is an example of a *reversible* gate; we can simply apply the NOT gate once more in order to recover the initial state.

The idea of a reversible gate is important when we consider *quantum gates*, or *unitary operators*, denoted by  $U$ . The quantum state evolving through unitary operators naturally implies reversibility as a unitary operator always has an inverse,  $U^\dagger$ . We then have unitary operators that satisfy

$$U^\dagger U = U U^\dagger = \mathbb{1}. \quad (3.11)$$

This property of reversibility is important as the quantum circuits we use for our algorithm in section 5.1 require this property in order to successfully approximate a state.

### 3.4 Quantum circuits

We arrange the quantum gates mentioned in the previous section in quantum circuits. We represent the action of a gate on a qubit as

$$|\psi\rangle \text{ --- } \boxed{U} \text{ ---}$$

Here,  $U$  is a single qubit unitary operator. A simple example of such a quantum gate is the Hadamard gate, which has the matrix representation

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}. \quad (3.12)$$

Applying the Hadamard gate onto one of our basis states creates a superposition of states

$$H|0\rangle = \frac{(|0\rangle + |1\rangle)}{\sqrt{2}}. \quad (3.13)$$

We can represent this action of the Hadamard gate on the 0 qubit as a quantum circuit

$$|0\rangle \text{ --- } \boxed{H} \text{ ---}$$

This is how we construct a quantum circuit. We can think of the qubit states as vectors and the gates as matrices. These are then multiplied together to produce the end result. It is key to note that quantum circuits are just a visual tool to show what happens without using mathematical notation; we can of course think of the circuits using abstract notation instead. Consider the following two-qubit circuit

$$\begin{array}{c} |0\rangle \text{ --- } \boxed{H} \text{ ---} \\ |1\rangle \text{ --- } \text{---} \end{array} \quad (3.14)$$

Here we can see a Hadamard gate being applied to the first qubit, in state  $|0\rangle$ , while no gate is applied to the second qubit, in state  $|1\rangle$ . We can write this as

$$H|0\rangle \otimes \mathbb{1}|1\rangle. \quad (3.15)$$

Here,  $\mathbb{1}$  is the identity matrix. The identity represents the idea that no gate was applied to the 1 qubit, meaning it does not change. This is exactly the effect of applying the identity matrix to the qubit. We can go a step further and write the entire circuit in terms of a matrix equation. The key idea is to think about the circuits in time order, where the time goes from left to right. Studying circuit 3.14, we first write the initial states of qubits as a vector. We know from equations (3.4) and (3.5) how we write the basis states as a vector, so taking the tensor product between our initial qubits states is written as

$$\begin{bmatrix} 1 \\ 0 \end{bmatrix} \otimes \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}. \quad (3.16)$$

Then, we can do the same for the quantum gates, now taking the tensor product of the Hadamard gate and identity matrix

$$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}. \quad (3.17)$$

Now we can simply multiply (3.17) and (3.16), and we get the final qubit state in vector form as

$$\frac{1}{\sqrt{2}} \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}. \quad (3.18)$$

Equation (3.18) is thus the vector representation of the final state of the qubits. This is the idea behind what a quantum circuit represents, and how we can apply gates to evolve an initial quantum state.

### 3.5 Tensor representation

There is a useful relationship between the circuit diagrams and tensors. To understand this, let us choose a  $N = 3$  system as an example. We can write any state in this system as

$$|\psi\rangle = \sum_{i,j,k} \psi_{i,j,k} |ijk\rangle, \quad (3.19)$$

where this is a sum over all basis states and the scalars  $\psi_{i,j,k}$  are the coefficients. These coefficients can be regarded as tensor objects where each index can take 2 values, therefore it is of shape  $(2, 2, 2)$ . Now, the diagram representation for this state is



The diagram shows a vertical rectangle labeled  $|\psi\rangle$  in the center. Three horizontal lines extend from the right side of the rectangle, labeled  $i$ ,  $j$ , and  $k$  from top to bottom respectively. This represents a tensor with three indices.

$$(3.20)$$

Here the dangling wires represent the individual indices of  $\psi_{i,j,k}$ . Each index can take 2 values (either 0 or 1) therefore each tensor leg in the diagram is of shape 2, forming the  $(2,2,2)$  shaped object.

We can use this to now discuss how quantum gates act on states. Let us take a 2 qubit gate  $U$ ; in our circuits we will only use 2 qubit gates but the discussion below generalizes to gates of any number of qubits. We know one can also represent  $U$  in basis form as follows

$$U = \sum_{l,m,n,p} U_{l,m,n,p} |lm\rangle \langle np|. \quad (3.21)$$

The object  $U_{l,m,n,p}$  is a tensor of shape  $(2, 2, 2, 2)$ ; this in diagrams can be represented as



The diagram shows a vertical rectangle labeled  $U$  in the center. Four horizontal lines extend from the rectangle: two on the left and two on the right. The left lines are labeled  $n$  (top) and  $p$  (bottom). The right lines are labeled  $l$  (top) and  $m$  (bottom). This represents a tensor with four indices.

$$(3.22)$$

where as before the legs indicate the 4 indices, each of shape 2. Upon algebraic manipulation one can show that when we act on qubits 1 and 2 with  $U$  we get the following new state

$$(U \otimes \mathbb{1}) |\psi\rangle = \sum_{i,j,k} \psi'_{i,j,k} |i, j, k\rangle, \quad (3.23)$$

where  $\psi'_{i,j,k}$  is

$$\psi'_{i,j,k} = \sum_{l,m} U_{i,j,l,m} \psi_{l,m,k}. \quad (3.24)$$

So the tensor of the resultant state is given by contracting the 3rd and 4th indices of  $U$  with the 1st and 2nd indices of  $|\psi\rangle$  respectively. If we number the indices as shown in circuits 3.20 and 3.22 (we use a 0 index numbering) then the contraction can be described by combining the legs. Putting this all together, equation (3.23) in diagrams is



$$(3.25)$$

The same procedure can be carried out for a two-qubit gate acting on other neighbouring qubits, for example in our case on qubits 2 and 3; the only difference is that at the end we need to reshuffle the tensor legs since they come out in the wrong order (in the case presented above this was not necessary).

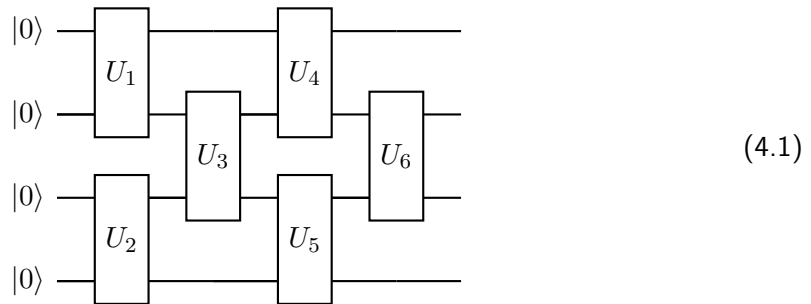
The above method for computing the action of gates is the method that we actually use when building our quantum circuits and for every computation involving the quantum circuits (like inner products). This method is equivalent to the Kronecker product method discussed in the previous subsection. However, when implementing it in Python it is a lot faster and more memory efficient; this becomes very significant when considering a large number of qubits  $N$ , where vector sizes scale exponentially with  $N$  while the number of indices in tensors scale linearly. Moreover, this method is useful when we carry out the optimization process where we will remove quantum gates and replace them with new ones.

### 3.6 The target state

Now we have seen how quantum circuits work, we can finally look to approximate the ground state. It is of course important to define the actual state we aim to approximate. In section 2.1, we looked at the Ising model with both a transverse and longitudinal field. We denote the target state as  $|\psi_T\rangle$ . By choosing values for  $J$ ,  $h$ , and  $g$ , we can then use exact diagonalization to generate our target state. Note that in this work, we are investigating the properties of the quantum circuit so we use the exact states for reference. In practice, we would use VQE or QAOA to find the ground state for larger systems. In the following section, we will outline the algorithm used to approximate the target state.

## 4 Brick wall circuit

We can now start thinking of how to build a quantum circuit that will approximate the ground state. We are going to use a brick wall structure circuit [30][31]. An example for  $N = 4$  and  $M = 4$  layers is shown in circuit 4.1. This structure is chosen as it is the densest circuit structure possible for a set of two-qubit gates. Also, intuitively, we can see from the figure that the quantum gates 'cover all qubits' in the sense that all qubits are connected to their nearest neighbours via a quantum gate; therefore we expect this structure to give us a generic state. Different circuit structures might have been used such as the sequential ansatz [32].



As mentioned in section 3.5, each gate in the brick wall circuit is a two-qubit gate acting on neighbouring qubits; so formally for a gate acting on qubits 2 and 3 in the above figure we have

$$V_3 := \mathbb{1} \otimes U_3 \otimes \mathbb{1}, \quad (4.2)$$

where  $V_3$  acts on the whole system. Therefore, we can write down the result of our quantum circuit in a compact way.

$$|\psi_c\rangle = V_6 V_5 V_4 V_3 V_2 V_1 |0000\rangle. \quad (4.3)$$

One can see that the resulting state is clearly dependent on the quantum gates that it is made out of. As a result changing a gate will change the output state. The ultimate goal of this chapter and the next is to find a set of gates that would produce a state that approximates the exact ground state. More about this will be explained in the next chapter. For now, we still have to explain how we initialize the circuit and what are our initial gates.

### 4.1 Gate initialization

At first, it seemed that how we pick the initial gates did not matter. However, after doing the optimization procedure we realized that the initialization of gates has an effect on optimization. In this section, we will describe the two methods we considered for initialization and in the following chapter we will describe how these methods impacted the optimization procedure, and which of the two was favoured in order to calculate our final approximated states.

The first method we looked at comprises of populating the circuit with random unitary 4x4 matrices for the initial quantum gates. We have used the `scipy.stats` library to produce the random unitary matrices.

```
from scipy.stats import unitary_group as U
gate = U.rvs(4)
```

The second method we looked at was a more systematic way of choosing the random matrices to be *close to the identity*. We first start with a random matrix  $R$  which we find

by using `numpy.random.rand(4,4)` which returns a random 4 by 4 real matrix with entries between 0 and 1 drawn from a uniform distribution. A fully random complex matrix can also be used instead. We then perturb the identity matrix slightly by this random matrix

$$M = \mathbb{1} + \epsilon R, \quad (4.4)$$

Where  $\epsilon$  is a small parameter that we chose to have a fixed value of 0.01. We can then calculate a Hermitian matrix

$$H = \frac{1}{2}(M + M^\dagger). \quad (4.5)$$

Finally, we can come up with a unitary by exponentiating the Hermitian matrix from equation (4.5) [33].

$$U = e^{iH}. \quad (4.6)$$

We chose to look at this alternative method because the ground state obtained through exact diagonalization is generally a low entangled state [34] and thus the ground states populate a small portion of the total Hilbert space. On the contrary, our brick wall circuit generates a state with a lot of entanglement when we initialize the gates as fully random unitaries, as outlined in our first approach, thus placing us far away from the portion of the Hilbert space where our target state lives. As a result, this makes optimization a harder task when choosing a random set of unitary matrices. A circuit with all the gates initialized as the identity would give us the lowest entangled state (in fact it would actually give us the ground state for  $J = 0, h = 1, g = 0$  parameters). Therefore what we do is allow some small random deviation from the identity to start us off which in turn puts us closer to where the ground state is and makes optimization easier. With this in mind, from now on we will be using this method.

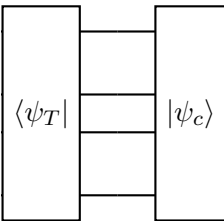
## 5 Optimization process

Now that we have introduced the brick wall circuit and discussed the initialization process, we are ready to discuss the optimization procedure. Recall that the goal is to find a state  $|\psi_c\rangle$  produced by our circuit that approximates the exact ground state  $|\psi_T\rangle$ . There are different metrics we can use to gauge how close we get to  $|\psi_T\rangle$ . We will be focusing on the inner product. To explain how this works we can think of our circuit 4.1 as a machinery that produces  $|\psi_c\rangle$ , this is shown in circuit 5.1



$$(5.1)$$

We can now find the inner product between the states  $|\psi_c\rangle$  and  $|\psi_T\rangle$  by contracting all tensor indices as follows



$$= \langle \psi_T | \psi_c \rangle, \quad (5.2)$$

where we note that we need to take the conjugate when we contract the indices. The above inner product gives a complex number dependent on the gates that the brick wall contains. We now need to construct an algorithm that will change the gates in order to maximize this inner product and get it as close to 1 as possible (recall that our states are normalized so in particular  $\langle \psi_T | \psi_T \rangle = 1$ ).

We tested 2 different optimization methods which we call *polar optimization* 5.1 and *gradient descent optimization* 5.2 respectively. In both methods the idea is that we sweep through the brick wall circuit gate by gate and replace the gates with *better* ones in the sense that overall we will be moving in the direction of increasing overlap. We call 1 sweep whenever we finished one round of going through each gate in the circuit. We do this for many sweeps in an attempt to reach the global maxima of the overlap. In the below plots and explanations I will look at how the infidelity  $1 - |\langle \psi_T | \psi_c \rangle|$  changes as a function of number of sweeps, thus the aim is to minimize this function as much as possible (get it as close to 0 as possible) and reach the global minima of this function.



### 5.1 Polar optimization

We start by representing the overlap in circuit diagram as we did in circuit 5.2 but we will keep the gates as part of the diagram this time to get

$$(5.3)$$

We will illustrate the optimization procedure for gate  $U_1$ , however the same is repeated for each gate in the circuit, for multiple sweeps. We start by removing the gate and regarding the resulting circuit as a 4 legged tensor

$$(5.4)$$

We can now think what can we put back in order to increase the overlap. If  $E$  would be unitary then the best gate to put back is actually  $\overline{E}$ , where the over-line indicates complex conjugation. Indeed putting this back would give us 1 overlap. The problem with this is that  $E$  and  $\overline{E}$  are not necessarily unitary therefore  $\overline{E}$  is not an appropriate gate to put back. However, we can get the best gate by carrying out singular value decomposition (SVD) on the matrix  $\overline{E}$  as shown below

$$\overline{E} = W S V^\dagger. \quad (5.5)$$

We then take as our new gate

$$U'_1 = W V^\dagger \quad (5.6)$$

where  $U'_1$  is now unitary. Finally, we reshape it and put it back inside the circuit

$$(5.7)$$

We show in B that this new inner product is larger than the previous one in circuit 5.3 and that  $U_1'$  is the unique optimal unitary to put back. By carrying this procedure for all the gates in the circuit and for multiple sweeps we are aiming to reach convergence.

As mentioned in the previous chapter, the optimization path depends on the way we initialize the gates. Figure 1 shows infidelity against number of sweeps for both initialization methods. We plot 10 runs for each method. It is observed that for the *close to identity* method it is harder for the optimization to get stuck in a local minima and that all runs roughly take the same path. Note that a minima is indicated by a flat line in the log-log plots below, we aim to reach the global minima. However, we can never be sure if we reached it or if our algorithm is stuck in a local minima. A good example of this is illustrated in the plot for random gates initialization where for one of the runs we see that the algorithm gets stuck in a minima, and we know it is local since for the other runs  $1 - |\text{overlap}|$  decreases further, this is one of the main reasons why we chose to use *close to identity* as the main initialization method. Nevertheless, this doesn't mean that *close to identity* method cannot encounter a local minima; we can see from the plot that even though it seems that we have reached the global minima, looking more closely, we can see that the lines start to curve downwards right at the end which indicates that this might not be the global minima. We will never know if we hit a global or local minima unless we run the algorithm for more sweeps. This, however, is not a huge problem since it only means that we haven't reached the best possible approximation nevertheless, looking at the numerical values of  $1 - |\text{overlap}|$  it is clear that the approximation is still very good.

As a side remark, one can also come up with a stopping criteria that can be used to determine when the optimization stops. We did initially implement a stopping criteria, however, we then decided to let optimization run for a set number of sweeps and not let the stopping criteria take action. Our implementation for the stopping criteria is based on the percentage change in  $1 - |\text{overlap}|$  between sweeps. We calculate the percentage change after each sweep and if it is below a minimum of 0.0001 then the optimization is halted. In figure 1 and all the below figures, the red crosses signify the points where the stopping criteria would have been triggered, had we not let the optimization run further.

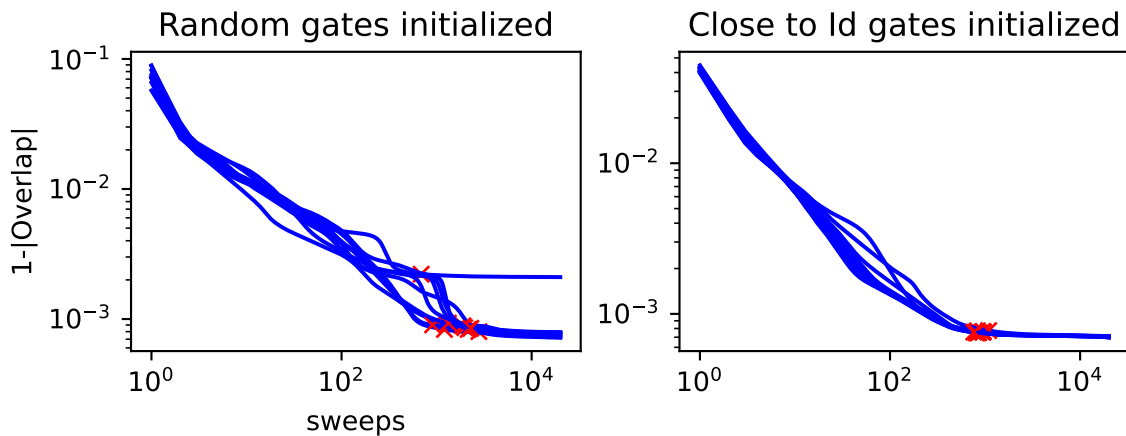


Figure 1: Random vs close to Id initialization compared when we run polar optimization for  $N = 14$ ,  $M = 5$ ,  $J = 1$ ,  $h = 1$ ,  $g = 1$  and 20000 sweeps. The red crosses signify the stopping criteria being triggered.

## 5.2 Gradient descent optimization

It is now time to explain the second optimization method. The setup for this method is identical to the previous one, the only difference arises when we decide which gate to put back in circuit 5.4. Instead of putting back the best possible gate from equation (5.6) we instead replace the gate with

$$U_1'' = U_1 + \eta U_1' \quad (5.8)$$

In other words we perturb the old gate  $U_1$  by the learning rate  $\eta$  in the direction of  $U_1'$ . Again, the only downside to this is that  $U_1''$  is not necessarily unitary, therefore we once again carry out SVD on our new gate

$$U_1'' = W' \Sigma' V'^{\dagger}, \quad (5.9)$$

and then throw away  $\Sigma'$  to get a unitary version of  $U_1''$

$$U_{new} = W' V'^{\dagger}. \quad (5.10)$$

This is now the gate that we put back in circuit 5.4. The learning rate  $\eta$  is a variable that needs to be chosen and tuned carefully, and it turns out that this task is non-trivial and optimization depends heavily on it. Another important thing to note is that this time the overlap doesn't necessarily always increase each time a gate is renewed, therefore there are some fluctuations, but overall over many sweeps (with a sensible tuning of  $\eta$ ) overlap does increase.

Figure 2(top right) shows an example of the gradient descent optimization. For this plot we have used a linear decrease in  $\eta$  tuning: first, we start at  $\eta = 0.5$  and decrease  $\eta$  linearly with the sweep number until it reaches  $\eta = 0.05$  when the sweep number reaches half of the maximum number of sweeps. When the halfway point is reached, we set  $\eta$  to 0.05 for the remaining sweeps. To understand why we chose this specific tuning, one needs to understand the effect of  $\eta$  on optimization. A higher value for the learning rate means that the optimization is faster paced and moves towards the minimum faster. This is at the cost of precision in the sense that after one sweep we might move closer to the minimum but after the next sweep, we might also overshoot and therefore move further away from the minimum again. This is precisely the fluctuations we see in figure 2(top right). Keeping the learning rate small also has its advantages and disadvantages; on one hand, precision is high and we move in very small increments towards the minimum, therefore, we mostly decrease and as a result the fluctuations disappear since it is highly unlikely for an overshoot to occur. On the other hand, high precision comes with the cost that it takes a lot more sweeps to reach the global minimum. With this in mind, we want to take advantage of both of these regimes when choosing a tuning scheme. We start with a higher  $\eta$  in the hope of getting the optimization going faster, once we get the optimization going and start decreasing significantly we lower  $\eta$  in order to reduce overshooting and increase precision. Linear decrease tuning is precisely trying to achieve the above. In figure 2(top right) we can see the above prediction, however we have not yet reached any minimum, as the graph is still decreasing. We also observe that due to the fluctuations, the stopping criteria previously designed doesn't work as intended, this just means that a better stopping criteria needs to be implemented for gradient descent method.

We have tried multiple different  $\eta$  tuning schemes A, however, we couldn't find one which is able to find a better approximation of the target state in fewer sweeps compared to *polar optimization*. This does not mean that such a tuning scheme doesn't exist, in fact finding such a scheme is a subject in itself [35].

### 5.3 Polar Vs gradient descent optimization

Now that we have described both optimization methods, it is time to put them side by side and choose one that will be used for the rest of this paper.

In figure 2(bottom row) we compare the optimization for the two methods where a linear decrease tuning in  $\eta$ , with starting value of  $\eta = 0.3$  decreasing to  $\eta = 0.05$  until  $sweeps = \frac{100000}{4}$  and then continuing at 0.05, was used. First of all, we note that the global minimum has not been reached in either method for those particular parameters. We do have some potential explanations for why this is the case which will be stated in the next subsection 5.4. Even so, we can ask ourselves which method gives the better approximation; we can answer this by looking at the values reached for  $1 - |\text{overlap}|$  at the end of optimization. We can see that *polar optimization* reaches smaller values compared to *gradient descent optimization* which means that the state reached from polar optimization is a better approximation. We also carried out comparisons for more qubits and other tuning methods and we generally find that *polar optimization* comes out on top. Since we are interested in getting the best possible approximation in the fewest number of sweeps, we conclude that *polar optimization* is better (at least for our simplistic  $\eta$  tuning). Moreover, *polar method* is generally more accurate and has no fluctuations which is a bonus. For these reasons we choose to use the *polar method* as the main optimization method.

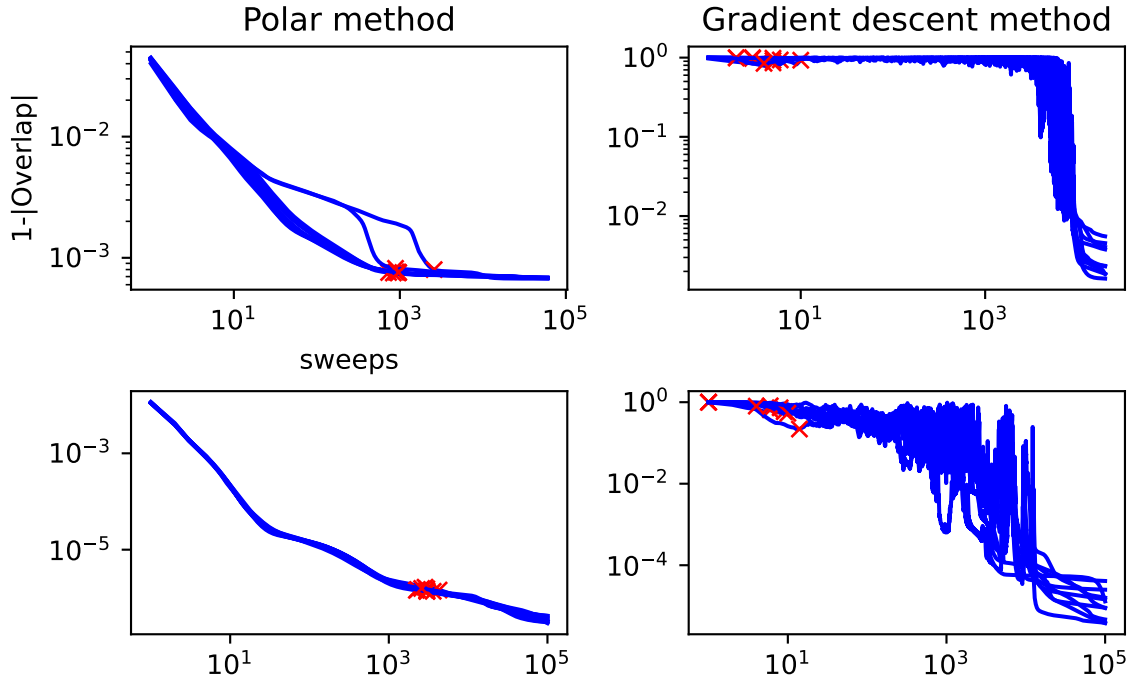


Figure 2: 10 runs each for both the polar and gradient descent method using a linear decrease in  $\eta$  for fixed  $M = 5$ ,  $J = 1$ ,  $h = 1$ ,  $g = 1$  and  $N = 14$ (top row),  $N = 8$ (bottom row).

### 5.4 Final results

Now that we have chosen a method, we can compare the optimization when we change qubit number  $N$ , Hamiltonian model, and number of layers  $M$ .

We will first start with varying  $N$  in figure 3. We keep the number of layers  $M = 5$  and the model  $J = 1, h = 1, g = 1$  constant and only change  $N$ . we look at how optimization differs for  $N = 8, N = 10, N = 12, N = 14$ . Interestingly enough, it seems that a minimum is reached faster for a higher number of qubits, and for  $N = 8$  convergence didn't even start. At the moment there are 2 possibilities: firstly, we see that for a higher number of qubits the minima is reached in the prescribed number of sweeps which could mean that the number of sweeps used is too small and we may be too early in the optimization meaning that the minima seen could be local; secondly, it might be that in fact we reach the global minima for the higher  $N$ . If it is the second possibility then this counter-intuitive behaviour can be explained as a result of the barren plateau problem [36]; the  $N = 8, M = 5$  circuit is deeper compared to the  $N = 14, M = 5$  circuit and it is known that deeper circuits suffer more from barren plateaus, therefore it is harder for the optimization procedure to move towards the minimum which matches with the observation from the plots.

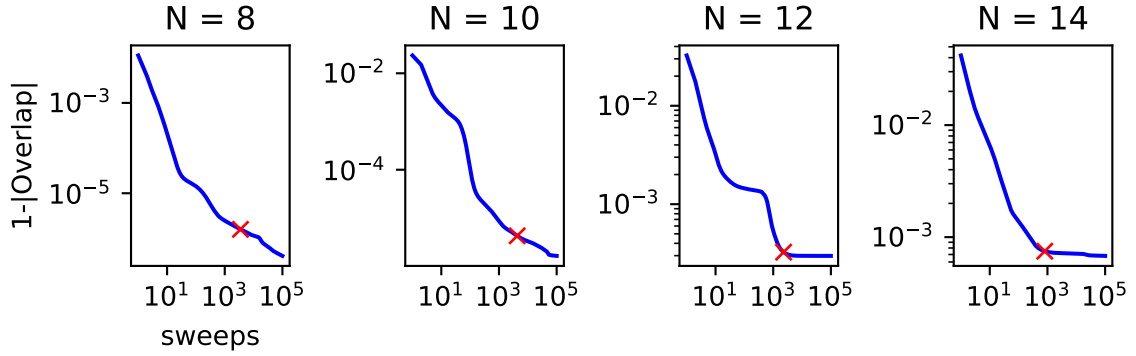


Figure 3: Polar optimization for fixed  $M = 5, J = 1, h = 1, g = 1, 1,000,000$  sweeps, comparing different qubit numbers.

Next, we can look at how optimization varies with the number of layers  $M$  when we keep the model and  $N$  fixed. The results of this comparison gives more insight into what we discussed in the previous paragraph. In figure 4 the circuit structures for  $M = 3$  and  $M = 5$  are shallower than the one for  $M = 7$  and the optimization seems to reach the global minimum for the former whereas it barely does for the latter. This is also in favour for the barren plateaus explanation, since again it shows that the deeper circuits ( $N = 8, M = 5$  and  $N = 14, M = 7$ ) might suffer from this problem and as a result it takes longer for the optimization to be completed.

Lastly we look at optimization for the different models and fix  $N = 14$  and  $M = 5$ . Figure 5 shows this. We can see that the integrable Hamiltonian at the phase transition completes optimization the fastest and the generic integrable one completes it the slowest. On the other hand, optimization provides the best approximation for the generic integrable model, and the worst approximation for the phase transition integrable model when using the final value of  $1 - |\text{overlap}|$  as our gauge. This matches our expectation since a critical point is somewhat special and harder to approximate compared to a generic point. Lastly, one of the main reasons we chose to look at the generic non integrable model is that it cannot be solved analytically to give a closed expression for the ground state while the integrable one can; and as a result, a natural question was if optimization even worked for such a model. We can now safely say that optimization works even for such a model.

In conclusion, we now have a working optimization algorithm that can be run multiple times

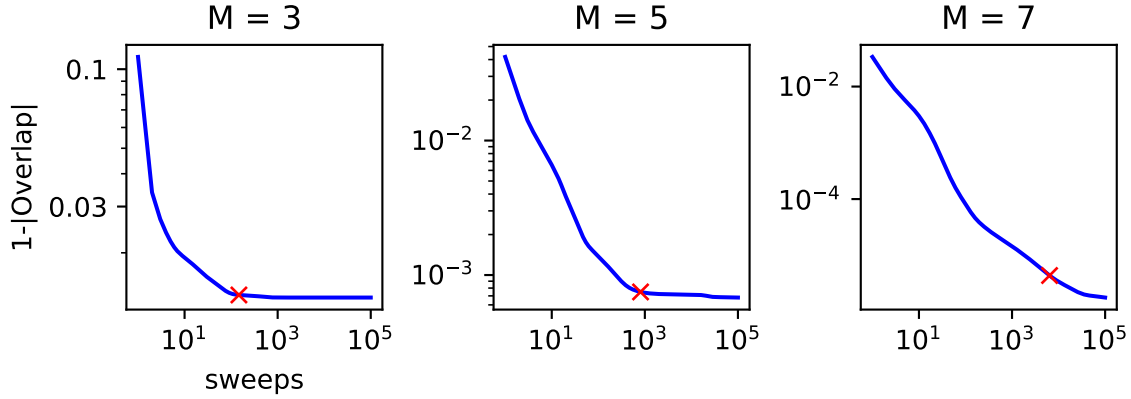


Figure 4: Polar optimization for fixed  $N = 14$ ,  $J = 1$ ,  $h = 1$ ,  $g = 1$ , 100000 sweeps, comparing circuit layers  $M \in \{3, 5, 7\}$ .

to create a database of approximate states to be used in the next chapters. More concretely, the database will contain the optimized gates for  $N \in \{8, 10, 12, 14\}$ ,  $M \in \{3, 4, 5, 6, 7\}$ ,  $(J, h, g) \in \{(1, 1.5, 0), (1, 1, 0), (1, 1, 1)\}$ , this comprises of a total of 60 sets of gates which can then be applied to the first basis state to produce the desired approximated state. The number of sweeps used for the database is 150000. The optimization algorithm was run on Nottingham's HPC Augusta in order to benefit from its parallelism and high computing power.

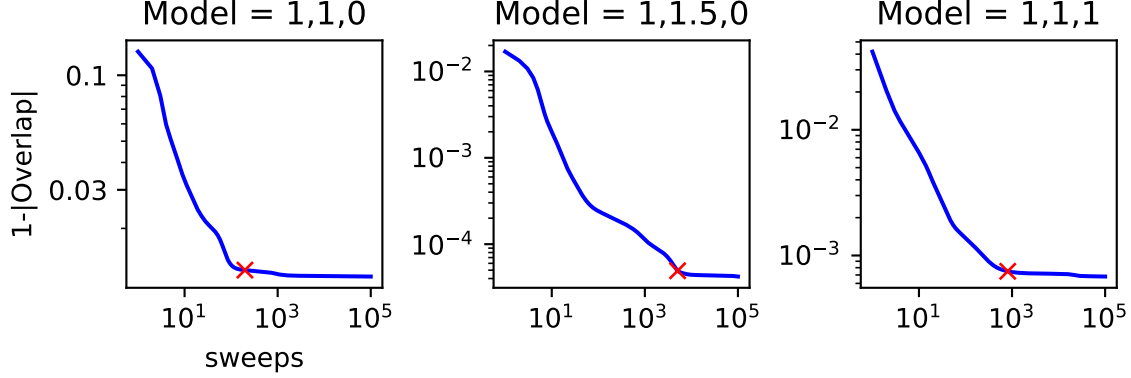


Figure 5: Polar optimization for fixed  $N = 14$ ,  $M = 5$ , 100000 sweeps, comparing models.

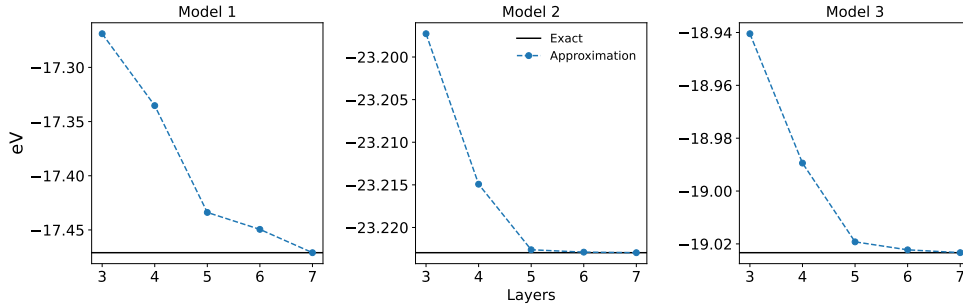


Figure 6: Approximation of the energy with increasing layers for each Model

## 6 State Properties

Now we have shown that we can successfully approximate the ground state of a quantum many-body system by the measure of the overlap, there are other quantities we want to measure on our approximated state. In this section, we discuss several other different information measures which quantify the amount of information and correlations in the quantum system.

### 6.1 Energy

One of the most basic properties we can measure on the ground state is the corresponding energy. The energy is given by

$$E = \langle \psi | H | \psi \rangle \quad (6.1)$$

Where  $H$  is the Hamiltonian and  $\psi$  is the wave function of the ground state. Our approximation is shown in Figure 6. In this figure, we gradually increase the number of layers and find the energy of the corresponding approximated wave function. We can see very similar behaviour for each of the models. For a smaller number of layers, our approximation of the energy is not close to the exact ground state energy. As we add more layers to our quantum circuit, we see the approximation gets better, and for each model, 7 layers gives us a near-perfect approximation of the ground state energy. Model 2 shows the fastest convergence to the ground state energy, while model 1 is the slowest. This could be due to the fact that model 1 is at the critical point, meaning approximating the energy is more difficult. Seeing that 7 layers gives us a good approximation for every model however, shows we can successfully approximate the energy of the ground state.

### 6.2 Quantum Entropy

#### 6.2.1 von Neumann entropy

The second of these measures will be the von Neumann entanglement entropy. The von Neumann entropy quantifies the amount of uncertainty in a given quantum system. It is defined as follows:

Given a quantum system  $A$ , which is in a state  $\rho_A$ , then the von Neumann entropy is defined as

$$S_A = -\text{Tr}_A[\rho_A \ln \rho_A]. \quad (6.2)$$

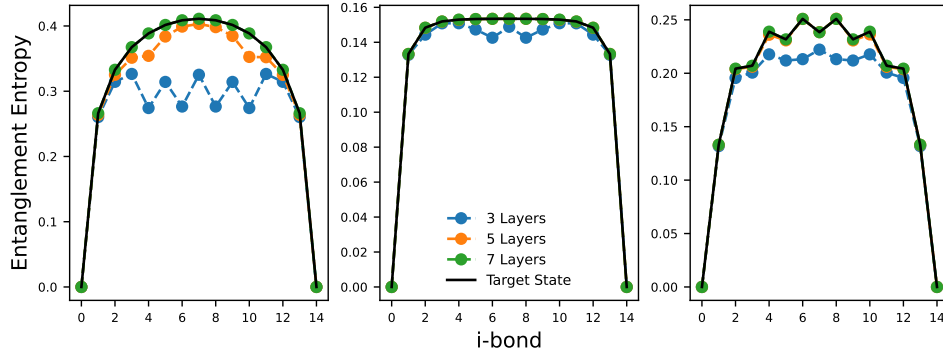


Figure 7: Comparison of the von Nuemann entanglement entropy for different models, with  $N = 14$ . From left to right, we have model 1, model 2, and model 3

We can use an eigen-decomposition of  $\rho_A = U^\dagger \Lambda U$  which results in the von Neumann entropy being equal to

$$S_A = - \sum_i \lambda_i \ln \lambda_i. \quad (6.3)$$

Here,  $\lambda_i$  are the Schmidt values of the density matrix.

The interpretation of the quantum entropy is important in this context. When a state is split into two subsystems, the von Neumann entropy measures how much information we do *not* know about the state of one of the subsystems. In other words, it measures how entangled one subsystem is with the other.

Using this measure, we then looked at how our approximated state compared to the exact ground state. We can see this in Figure 7. Here, we plot the entanglement entropy against the i-bond, where the i-bond bond dimension of the qubits. Model 1 is at the critical point for the Ising model, and as outlined in section 2.1, this point lies where the model undergoes a phase transition. We can see here that for 3 layers, we are not able to match the entanglement entropy of the target state perfectly; there is still a visible difference between the target state and our approximated state. The zig-zag pattern we see could be attributed to the alternating pattern of our brick wall circuit. For fewer layers, the circuit is trying to increase the entropy up to the required value, and so with more layers, it is able to refine this spectrum more to match the required entropy. When moving to 5 layers, we see a very close match between our approximated state and the target state for models 2 and 3, while for model 1 we still see a visible difference between the exact and approximated wave function. Increasing to 7 layers, we see an almost perfect approximation of the entanglement entropy for each model. This represents a promising result, as for each model we have an almost exact approximation of the quantum property we can measure on a ground state. This represents an interesting result as it shows that whether the model is integrable or not does not impact our approximation using a quantum circuit, and a similar argument applies to whether we are at a critical point or not.

In section 2.1.2 we introduced the correlation length. The shape of each graph for the von Neumann entanglement entropy is important when it comes to examining our approximations, and these shapes can be explained by considering the correlation length for each model. For model 1, at the critical point, we see a more domed shape, and the value of the entanglement entropy is much larger in comparison to models 2 and 3. We can see that the entanglement



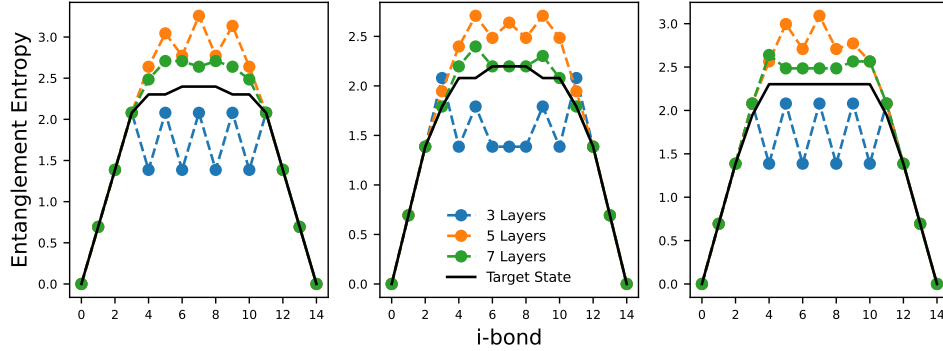


Figure 8: Approximation of Rényi entropy for each of our models for  $N = 14$ ,  $\alpha = 0$ . From left to right, we have model 1, model 2, and model 3

entropy has grown larger due to the correlation length diverging at this point. This explains the shape we see. The entanglement entropy naturally drops off near the boundaries of our system of 14 qubits, however, it increases as we reach the center of the qubits. Due to the infinitely long correlation length, the central qubit would be highly entangled with every other qubit, which is why there is a peak. In models 2 and 3, we are not at a critical point, meaning the correlation length decays exponentially between sites. The graph corresponding to model 2 is exactly what we would expect, where away from the boundaries, the entanglement entropy remains at a similar point giving more of a flatter shape, meaning entanglement is highest between nearest neighbour sites. Interestingly, although model 3 represents a non-critical point of the Ising model, we see a different shape from what we would expect, such as in model 2, with the shape instead appearing jagged. This is due to the low number of qubits, and the large but finite correlation length. The effect of the boundary being this close to the central qubits has caused the shape to appear the way it does. If we increased the number of qubits, the shape of model 3 would more closely resemble that of model 2.

### 6.2.2 Rényi entropy

We can generalise the idea of von Neumann entropy by using the same Schmidt values we calculated in equation (6.3) to compute the entanglement entropies known as the Rényi entropies. The equation for the Rényi entropy is given by

$$S_A^\alpha = \frac{1}{1-\alpha} \ln \sum_{i=1} \lambda_i^\alpha \quad \alpha \geq 0, \alpha \neq 1. \quad (6.4)$$

From this equation, we see that the Rényi entropy is not just one kind of entropy, but a whole family of entropies characterized by the value  $\alpha$ . Note that the von Neumann entropy is recovered as  $\alpha$  tends to 1. Some notable cases of the Rényi entropy are when  $\alpha = 0$ , which is known as the max-entropy or Hartley entropy[37]. This quantifies the logarithm of non-zero eigenvalues of the reduced density matrix[38]. Another notable case is when  $\alpha \rightarrow \infty$ , known as the min-entropy. The min-entropy is a measure of the maximum amount of information that can be extracted from a system. It is closely related to the concept of randomness, and has uses in cryptography[39]. It is also proportional to the greatest Schmidt value.

We can see our approximation in Figure 8, for  $\alpha = 0$ . Our approximated state, in this

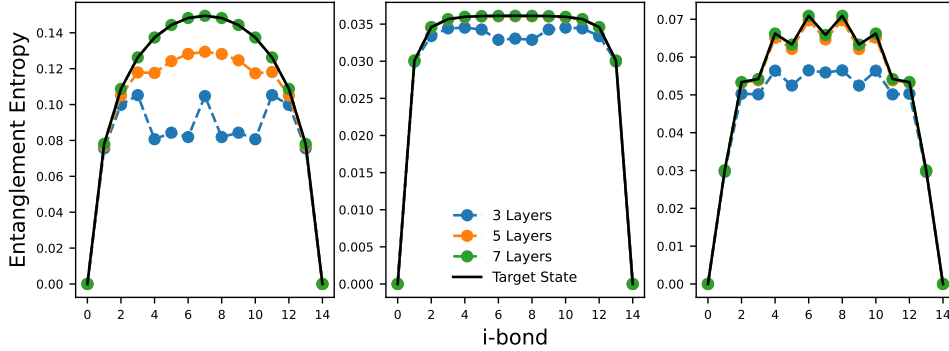


Figure 9: Approximation of Rényi entropy for each of our models for  $N = 14$ , large  $\alpha$ . From left to right, we have model 1, model 2, and model 3

case, does not quite match the target state particularly well. To explain why this happens, we have to remember that the max-entropy only indicates the number of non-zero eigenvalues for each  $i$ -bond. Our approximated state takes into account *all* possible eigenvalues. This includes the eigenvalues with a value of zero, thus increasing the entanglement entropy in these cases. For a larger number of layers, our approximated state simply quantifies the total number of eigenvalues there is for the corresponding  $i$ -bond. This means that increasing the layers even further would not give us a better approximation for the max-entropy. Knowing exactly how many non-zero eigenvalues there are is not the most interesting property we can measure on our states however, so we will instead look at different values of  $\alpha$  and see if our approximation is more accurate in those cases.

The results for a larger value of  $\alpha$ , which is used to look at the min-entropy, is plotted on figure 9. Here we can see a better approximation of the target state as we would expect. Similarly for  $\alpha = 0$ , we see that having 3 layers does not match the target state for each model, however we see a much more accurate match for more layers. Models 2 and 3 have an almost perfect match of the entanglement entropy for just 5 layers, whereas model 1 requires 7 layers before we see an accurate match with the target state. Interestingly, the shapes of the plots match those seen in section 6.2.1, with the only difference being the values of entanglement entropy are much lower. We expect the shapes to be the same as we are still measuring the entanglement entropy, and the values of entanglement entropy are lower due to the different weightings of Schmidt values for the Rényi entropy with a large  $\alpha$ .

### 6.3 Mutual Information

Another quantity we want to measure on our approximated states is the mutual information. In classical information theory, mutual information is the measure of how much information two random variables,  $A$  and  $B$ , share. In the quantum case, we extend this idea to our quantum systems. For a bipartite state,  $\rho_{AB}$ , we define the quantum mutual information as

$$I(A : B) = S(A) + S(B) - S(A \cup B) \quad (6.5)$$

Where  $S(A \cup B)$  is the von Neumann entropy of the joint density matrix of the two systems, and  $S(A)$  and  $S(B)$  are the von Neumann entropies of the individual systems,  $A$  and  $B$ .

We can represent this data in a variety of ways. One such representation is the creation

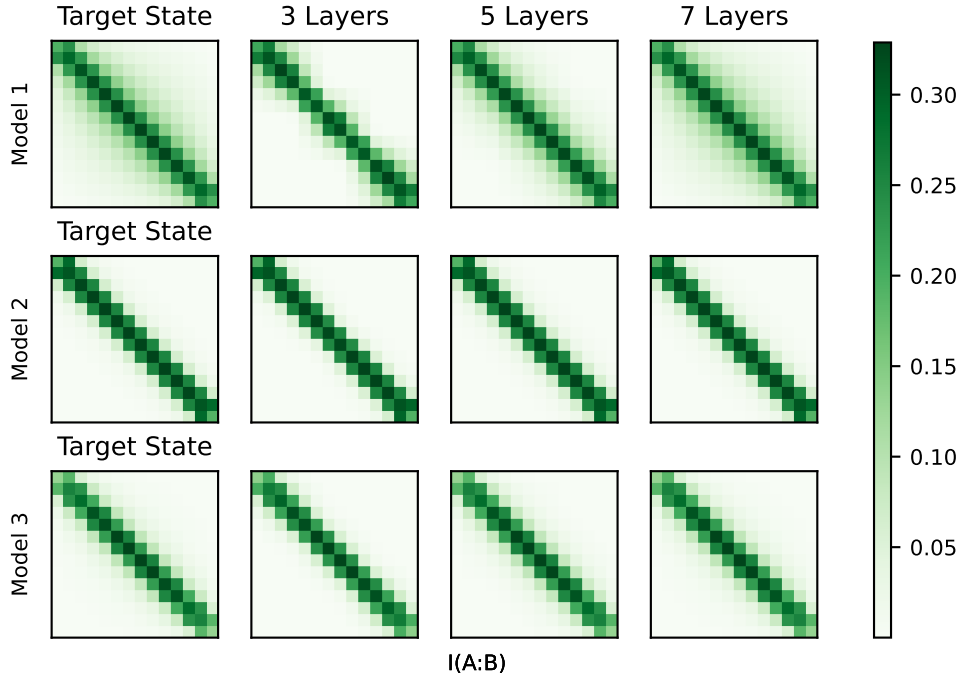


Figure 10: Colour matrices for each model, for  $N = 14$ . Each square corresponds to a different value of  $I(A : B)$ , where  $A, B \in [1, 14]$ . The darker an individual square is, the larger the mutual information on that site.

of a matrix as depicted in figure 10, in which the darkness of a square corresponds to the magnitude of  $I(A : B)$ . Here, we are taking  $A$  and  $B$  to be single sites within our system. The entanglement between sites can be measured by analyzing the mutual information between the qubits. Looking at the target state for each model, we can see that the colour map is darkest down the leading diagonal. These sites will be the same as place each other, hence the amount of mutual information between them will naturally be higher. As we measure the mutual information between sites further apart, we can see the mutual information decreasing. For our target state in model 1, we see the mutual information between sites decay as a power law relation as we move to sites further away. As mentioned in section 6.2.1, this is due to the correlation length diverging for this critical point, meaning sites share mutual information with an infinite correlation length. It is once again the boundaries of our system which cause the mutual information to drop as we approach the boundaries. For models 2 and 3, which are not critical, we see the correlation length drops off exponentially between sites, and most mutual information is concentrated across adjacent sites. Something to note is how this colour map has a line of symmetry down the leading diagonal, and this is due to the symmetry of measuring between sites; as  $I(A : B) = I(B : A)$ .

As we increase the number of layers for each model, we see a good approximation of the target state for each model. This is a promising sign as not only are we able to approximate the mutual information, but we can do so for a high degree of accuracy. For model 1, we can see that having 3 layers does not quite capture the decay of mutual information as we move

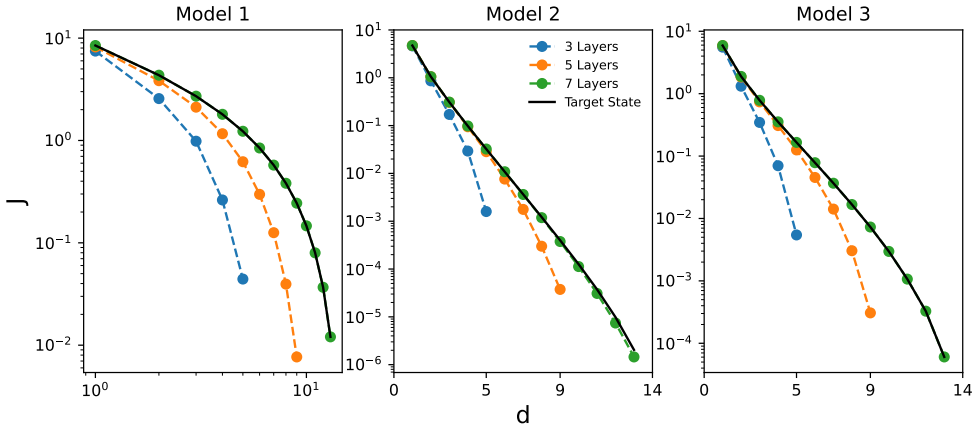


Figure 11: Sum of the mutual information for Models 1, 2, and 3, as given by equation (6.6). Model 1 is plotted on a log-log scale whereas models 2 and 3 are plotted on a log scale.

to sites further away, while 5 and 7 layers capture this property more. This will be expanded on when we look at a different way to represent the mutual information. Models 2 and 3 show a very good approximation with just 3 layers, which potentially means we can use fewer resources in our quantum circuit to approximate the mutual information for these models.

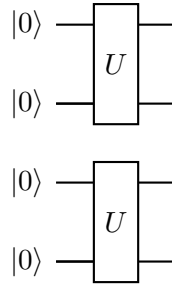
Another way to graphically look at the mutual information is to look at the *sum* of mutual information for a given distance between the sites. For neighbouring qubits  $i$  and  $j$ , we denote this as

$$J(d) = \sum_{|i-j|=d} I(i:j), \quad d = |i-j|. \quad (6.6)$$

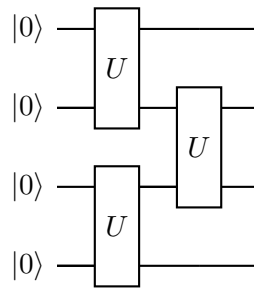
So we can see  $J$  is the sum of the mutual information from equation (6.5). In figure 11, we can again study our results for the different models. Unlike in our colour maps, we can now directly see the differences for each number of layers. Although 3 and 5 layers for each model capture the general shape of the target state, they don't accurately match the correct value of the summed mutual information. For models 2 and 3, for the target state, we see roughly a straight-line decrease of mutual information between sites. Since these models are not at a critical point, and only the y-axis is done on a log scale, this shows exactly the exponential decrease we would expect for the mutual information based on our discussion of the correlation lengths. Model 1 is critical so we would expect to see a straight-line decrease when both axes are on a log scale, representing the linear decrease due to the correlation length diverging according to a power law relation. We instead see a curved line. This is due to the system size being too small in this case, and the boundaries being too close to each other. If we were to increase the number of qubits, we would instead see a straight line decrease for the target state.

A key feature of these graphs is the fact that for 3 and 5 layers, the summed mutual information stops at  $d = 5$  and  $d = 9$  respectively. For two sites to have any form of entanglement, and hence share any mutual information, they need to be connected by a unitary operator. As an example, consider the very basic 4 qubit circuit with 1 layer of gates

as follows



When taking a measure on the qubits after the gates have been applied, we would be able to measure a non-zero entanglement between qubits 1 and 2, and also qubits 3 and 4. This means that these pairs of qubits share mutual information. However, if we measure the entanglement between qubits 1 and 3, for example, we would see an entanglement of zero, as there are not any gates which create a path from qubit 1 to qubit 3. This means the maximum value of  $d$  is 1, as only adjacent qubits share mutual information. If we were to add one more layer however, the circuit now has the form



As we know from before, just one layer causes the pairs of qubits 1, 2 and 3, 4 to be entangled. Adding the second layer which is one extra gate allows qubit 2 to become entangled with qubit 3. Now qubits 2 and 3 are entangled and therefore share mutual information, on top of the qubits which shared mutual information from just one layer. This means the maximum value of  $d$  is now 2.

Another way to visualize this is by looking at whether two qubits are connected by a gate when tracing from the end to the beginning of the quantum circuit. Qubits 1 and 3 must share mutual information as we can see the two gates that connect them. This is the idea behind why the plots in figure 11 have the patterns they do for 3 and 5 layers. For a 14-qubit circuit with 3 layers, mutual information can only be shared between qubits where  $d$  has a maximum value of 5. Similarly, 5 layers only allows for a maximum  $d$  of 9 for a non-zero mutual information. 7 layers then, allows all 14 qubits to have a non-zero value of mutual information. This explains the behaviour we saw on our colour maps for model 1, where for 3 layers we didn't see an exponential decrease in mutual information between sites, but an abrupt cut-off as we move away from the leading diagonal. 3 layers are not sufficient for the qubits with  $d > 9$  to share mutual information at all. This means that despite the infinite correlation length, the fact there was no mutual information shared in the first place explains these results.

## 7 Discussion

In this paper, we introduced a quantum brick wall circuit in order to approximate a ground state generated using exact diagonalization. By using the transverse-field Ising model with and without a longitudinal field as an example for our ground states, we were successfully able to build an approximation to the ground state using a quantum circuit algorithm which captured different state properties.

We then looked at other state properties to see if the approximation captured these measures. The energy, quantum entanglement entropy, and mutual information were all matched to a high degree of accuracy as we increased the number of layers. Our aim was to see if our quantum circuit could successfully reproduce the properties for low  $M$ , and it seems that for each model, we saw similar points where we successfully matched the target state. This is a property of the fact we used the fidelity as a measure of when our approximated state converged. Our approximated state will try and increase the accuracy of all of the state properties simultaneously, without favouring a specific one. Using the energy from section 6.1 as the objective function in the optimization to obtain an approximated state may mean different state properties are prioritized as we move to our final approximated state. So doing optimization in these two ways and then measuring different correlation functions one will find out which properties are better approximated by what optimization method. This can be useful in practice since one can choose which method to use to approximate the state depending on what properties of the state are required to be well approximated. Additionally, a more thorough investigation of the  $\eta$  tuning is required, especially if the energy approximation method is pursued. This is because for that method one cannot use polar optimization and relies solely on gradient descent optimization for which we have seen that  $\eta$  plays a major role in how good or bad the optimization does.

These results still leave us with some potential questions. When looking at the state properties, we used 14 qubits as a constant to measure the various quantum properties of our approximated state and the exact wave function. Although 7 layers seemed to provide good approximations, it is key to remember that the more layers we add, the more resources we use when applying our algorithm to the trivial wave function. Using as few layers as possible while also giving us a good approximation of the state is the aim. This is important because it seems that model 1 generally required the most layers in order to accurately match our target state for each of the state properties. We know this is at the critical point of the Ising model and so it required the most amount of resources to build a good approximate wave function. One potential question is whether these approximate wave functions had converged for model 1, so we looked in our database of wave functions mentioned in section 5.4 and found that the results very closely resemble figure 3, which are our results for model 3. For 3 and 5 layers we reached a minimum while 7 layers showed no signs of a minimum being reached. This indicates then that model 1 is simply a complex ground state to approximate so naturally requires a deeper quantum circuit in order to get an approximation which has similar state properties to the exact one.

An interesting follow-up could be investigating the structure of quantum circuits used. As mentioned in section 4, different circuit structures could have been considered, such as the sequential ansatz in reference [32]. Investigating a different circuit structure different to our brick wall approach could potentially approximate a target state in even fewer layers. Another variable we could change is the number of qubits. The largest number of qubits used was 14, so adding additional qubits could possibly give us better approximations of the state

properties. It is important to note however that the time taken to reach an approximation scales exponentially with system size, so we would have to take this into consideration. We could also look at different Hamiltonians. Using the quantum Ising model was a choice, so seeing if similar results can be replicated for different models would increase the justification for the use of quantum circuits.

Using our knowledge that we can successfully approximate a ground state of the Ising model, implementing these ideas of approximating a ground state on an actual quantum computer could also be the next step we could take. Instead of using ED to generate a target state, we would instead use one of the methods used in quantum computers, such as VQE or QAOA. Unlike our approach, which uses the fidelity to approximate the state, these methods use the energy to approximate a state. This would make for an interesting comparison of whether the same quantum properties can be approximated using this approach.

## **Acknowledgments**

We thank our supervisor Dr. Adam Smith (University of Nottingham) for his guidance and helpful discussions. We thank the University of Nottingham for allowing us to use Augusta - The University's High-Performance Computer.

## Appendices

### A $\eta$ tuning schemes

In this short section we will present some different  $\eta$  tuning schemes we tried.

Firstly we considered a constant  $\eta = 0.3$  throughout all the sweeps. The results are plotted in figure 12. Compared to figure 2, we see that the fluctuations remain high throughout the optimization and not getting smoother towards the end, this is precisely the expected behaviour from the discussion in 5.2. As a result, keeping  $\eta$  constant doesn't seem to be of much use due to the big imprecision.

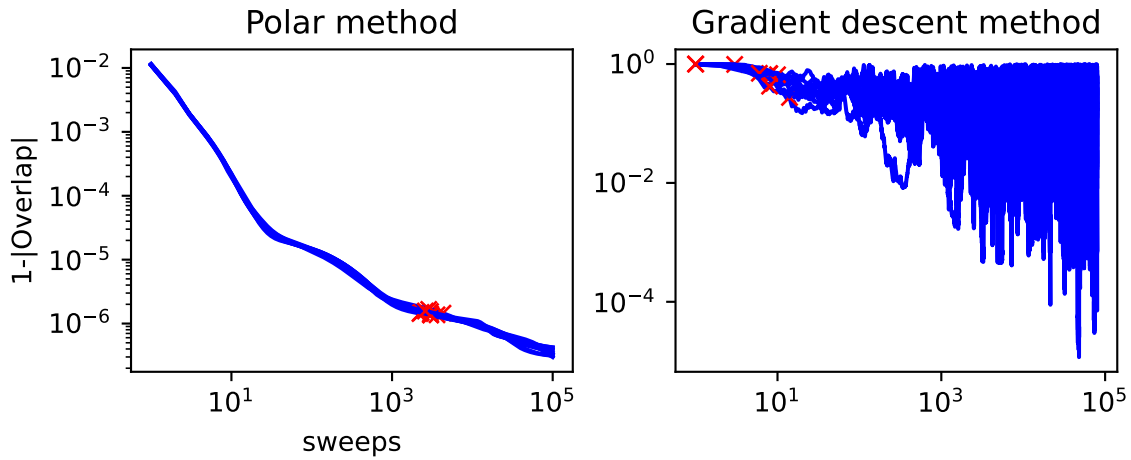


Figure 12: 10 runs each for both the polar and gradient descent method for  $N = 8$ ,  $M = 5$ ,  $J = 1$ ,  $h = 1$ ,  $g = 1$  and 100000 sweeps. Constant  $\eta = 0.3$ .

The last approach we tried is a type of decay in  $\eta$ . In the beginning  $\eta = 0.3$  and every quarter of maximum sweeps,  $\eta_{new} = 0.65 \times \eta_{old}$  until the number of sweeps reaches half of the maximum sweeps, after which  $\eta$  continues as a constant. Figure 13 shows the result for this tuning method. Compared to the linear decrease tuning in figure 2, we see that there are less fluctuations and that at the end of the optimization,  $1 - |\text{overlap}|$  is slightly lower meaning a better approximation; nevertheless, polar optimization still gives lower values.

As stated before, these are just another 2 of many possible tuning methods and as seen polar method comes on top when considering these.

### B Increasing overlap

In this appendix we will present a proof of why the overlap always increases whenever we replace the old gates with the new ones discussed in subsection 5.1.

First let us recall we have  $E_{i,j}$  from equation (5.4) written in matrix form. Let us name the old gate  $U_{i,j}$ , then the old overlap is

$$|\sum_{i,j} E_{i,j} U_{i,j}| = |\text{Tr}(EU^T)|, \quad (\text{B.1})$$



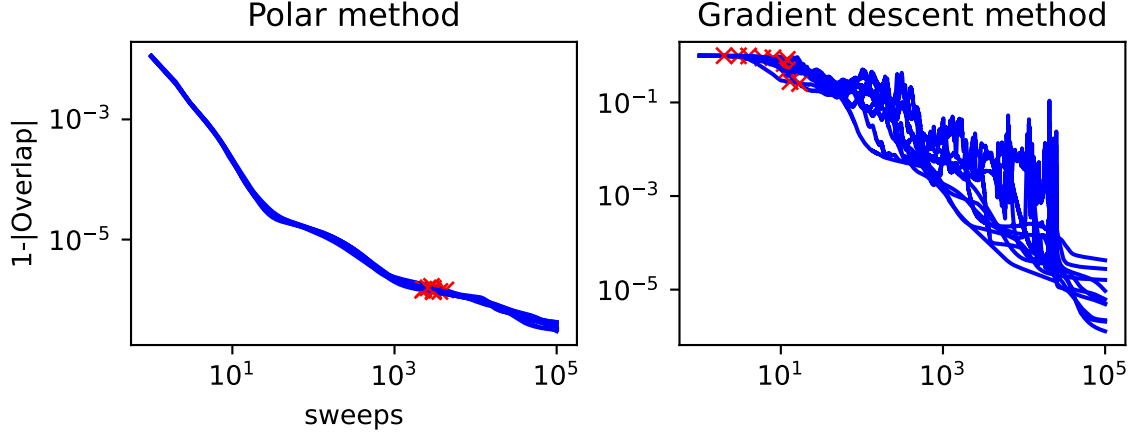


Figure 13: 10 runs each for both the polar and gradient descent method for  $N = 8$ ,  $M = 5$ ,  $J = 1$ ,  $h = 1$ ,  $g = 1$  and 100000 sweeps. Decay in  $\eta$ .

where we have written the contraction as the trace of the matrix product of  $E$  and the transpose of  $U$ . The goal is now to find a new gate  $U'_{i,j}$  such that the overlap is maximized, so we want

$$|\sum_{i,j} E_{i,j} U'_{i,j}| = |\text{Tr}(E U'^T)| \quad (\text{B.2})$$

to be maximal. We find that this is achieved for  $U'$  as given in equation (5.6).

To show this let us first rename the gate  $U'$  to  $\overline{M}$

$$U' = \overline{M}, \quad (\text{B.3})$$

where the over-line indicates complex conjugation. The task is now to find a unitary  $M$  such that  $|\text{Tr}(E M^\dagger)|$  is maximized. Let us carry out SVD on  $E$  to get

$$E = H S T^\dagger, \quad (\text{B.4})$$

and now plugging this in the trace we get

$$\text{Tr}(E M^\dagger) = \text{Tr}(W S). \quad (\text{B.5})$$

Here  $W$  is a unitary given by

$$W = T^\dagger M^\dagger H. \quad (\text{B.6})$$

One then finds the following inequality

$$|\text{Tr}(E M^\dagger)| = |\sum_j W_{j,j} S_j| \leq \sum_j |W_{j,j} S_j| = \sum_j |W_{j,j}| S_j \leq \sum_j S_j = \text{Tr}(S). \quad (\text{B.7})$$

We have used the fact that  $S$  is a positive diagonal matrix to manipulate  $S$ . The first inequality is the triangle inequality; and the second one follows from the unitarity of  $W$  as follows

$$W W^\dagger = \mathbb{1} \implies \sum_j W_{i,j} \overline{W}_{k,j} = \delta_{i,k} \implies \sum_j |W_{i,j}|^2 = 1 \implies |W_{i,j}| \leq 1 \quad \forall i, j. \quad (\text{B.8})$$

Now in order to maximize  $|Tr(EM^\dagger)|$  we would like the two inequalities in equation (B.7) to be equalities, therefore let us see what we get if we force these conditions. First take the second inequality and force equality to get the following

$$\sum_j |W_{j,j}| S_j = \sum_j S_j \iff |W_{j,j}| = 1 \forall j \implies W_{j,j} = e^{i\theta_j} \text{ and } W_{i,j} = 0 \text{ for } i \neq j. \quad (\text{B.9})$$

The  $W_{i,j} = 0$  follows from equation (B.8). Equation (B.9) then gives us

$$W_{i,j} = e^{i\theta_j} \delta_{i,j}. \quad (\text{B.10})$$

Now if we also force the first inequality into an equality we get

$$|\sum_j e^{i\theta_j} S_j| = \sum_j |e^{i\theta_j} S_j| \iff e^{i\theta_j} = e^{i\theta} \forall j. \quad (\text{B.11})$$

Therefore equations (B.10) and (B.11) together give

$$W = e^{i\theta} \mathbb{1}, \quad (\text{B.12})$$

and using equation (B.6) we get

$$M = e^{-i\theta} H T^\dagger. \quad (\text{B.13})$$

To make this match with what was discussed in subsection 5.1, use equation (B.3) to get

$$U' = e^{i\theta} \overline{H} T^\dagger, \quad (\text{B.14})$$

also note that if we take equation (B.4) and take complex conjugation we get

$$\overline{E} = \overline{H} S \overline{T}^\dagger, \quad (\text{B.15})$$

which is the SVD of  $\overline{E}$  shown in equation (5.5) with  $W = \overline{H}$  and  $V^\dagger = \overline{T}^\dagger$ . Finally this relabeling gives

$$U' = e^{i\theta} W V^\dagger, \quad (\text{B.16})$$

which is precisely equation (5.6) with the only difference being the phase in front; this however is not a problem since as we know phases can be ignored as they are nonphysical and don't alter quantum mechanical measurements.

This completes the proof that whenever we put back equation (B.16) in the circuit the overlap increases, and not only that, but the gate in (B.16) is the most unique optimal gate to put back that will increase the overlap the most.

## References

- [1] J. Preskill, "Quantum computing in the NISQ era and beyond," *Quantum*, vol. 2, p. 79, Aug. 2018. DOI: 10.22331/q-2018-08-06-79. [Online]. Available: <https://doi.org/10.22331/q-2018-08-06-79>.
- [2] D. J. Bernstein, "Introduction to post-quantum cryptography," 2009.
- [3] X. Gao, Z. Zhang, and L. Duan, "An efficient quantum algorithm for generative machine learning," Nov. 2017.
- [4] D. Jaksch, J. I. Cirac, P. Zoller, S. L. Rolston, R. Côté, and M. D. Lukin, "Fast quantum gates for neutral atoms," *Physical Review Letters*, vol. 85, no. 10, pp. 2208–2211, Sep. 2000. DOI: 10.1103/PhysRevLett.85.2208. [Online]. Available: <https://doi.org/10.1103/PhysRevLett.85.2208>.
- [5] Y. Nakamura, Y. A. Pashkin, and J. S. Tsai, "Coherent control of macroscopic quantum states in a single-cooper-pair box," *Nature*, vol. 398, no. 6730, pp. 786–788, Apr. 1999. DOI: 10.1038/19718. [Online]. Available: <https://doi.org/10.1038/19718>.
- [6] J. I. Cirac and P. Zoller, "Quantum computations with cold trapped ions," *Phys. Rev. Lett.*, vol. 74, pp. 4091–4094, 20 May 1995. DOI: 10.1103/PhysRevLett.74.4091. [Online]. Available: <https://link.aps.org/doi/10.1103/PhysRevLett.74.4091>.
- [7] C. Monroe, D. M. Meekhof, B. E. King, W. M. Itano, and D. J. Wineland, "Demonstration of a fundamental quantum logic gate," *Phys. Rev. Lett.*, vol. 75, pp. 4714–4717, 25 Dec. 1995. DOI: 10.1103/PhysRevLett.75.4714. [Online]. Available: <https://link.aps.org/doi/10.1103/PhysRevLett.75.4714>.
- [8] E. Knill, R. Laflamme, and G. Milburn, *Efficient linear optics quantum computation*, 2000. arXiv: quant-ph/0006088 [quant-ph].
- [9] H.-S. Zhong, H. Wang, Y.-H. Deng, *et al.*, "Quantum computational advantage using photons," *Science*, vol. 370, no. 6523, pp. 1460–1463, Dec. 2020. DOI: 10.1126/science.abe8770. [Online]. Available: <https://doi.org/10.1126/science.abe8770>.
- [10] B. Kane, "Silicon-based quantum computation," *Fortschritte der Physik*, vol. 48, no. 9–11, pp. 1023–1041, Sep. 2000. DOI: 10.1002/1521-3978(200009)48:9/11<1023::aid-prop1023>3.0.co;2-j. [Online]. Available: [https://doi.org/10.1002/1521-3978\(200009\)48:9/11<1023::aid-prop1023>3.0.co;2-j](https://doi.org/10.1002/1521-3978(200009)48:9/11<1023::aid-prop1023>3.0.co;2-j).
- [11] M. T. Madzik, S. Asaad, A. Youssry, *et al.*, "Precision tomography of a three-qubit donor quantum processor in silicon," *Nature*, vol. 601, no. 7893, pp. 348–353, Jan. 2022. DOI: 10.1038/s41586-021-04292-7. [Online]. Available: <https://doi.org/10.1038/s41586-021-04292-7>.
- [12] M. Kalos, "Monte carlo methods in quantum many-body problems," *Nuclear Physics A*, vol. 328, no. 1, pp. 153–168, 1979, ISSN: 0375-9474. DOI: [https://doi.org/10.1016/0375-9474\(79\)90217-3](https://doi.org/10.1016/0375-9474(79)90217-3). [Online]. Available: <https://www.sciencedirect.com/science/article/pii/0375947479902173>.
- [13] G. Pan and Z. Y. Meng, *Sign problem in quantum monte carlo simulation*, 2022. arXiv: 2204.08777 [cond-mat.str-el].

- [14] E. Y. Loh, J. E. Gubernatis, R. T. Scalettar, S. R. White, D. J. Scalapino, and R. L. Sugar, "Sign problem in the numerical simulation of many-electron systems," *Phys. Rev. B*, vol. 41, pp. 9301–9307, 13 May 1990. DOI: 10.1103/PhysRevB.41.9301. [Online]. Available: <https://link.aps.org/doi/10.1103/PhysRevB.41.9301>.
- [15] D. Hangleiter, I. Roth, D. Nagaj, and J. Eisert, "Easing the monte carlo sign problem," *Science Advances*, vol. 6, no. 33, Aug. 2020. DOI: 10.1126/sciadv.abb8341. [Online]. Available: <https://doi.org/10.1126/sciadv.abb8341>.
- [16] R. Finsterhölzl, M. Katzer, A. Knorr, and A. Carmele, "Using matrix-product states for open quantum many-body systems: Efficient algorithms for markovian and non-markovian time-evolution," *Entropy*, vol. 22, no. 9, p. 984, Sep. 2020, ISSN: 1099-4300. DOI: 10.3390/e22090984. [Online]. Available: <http://dx.doi.org/10.3390/e22090984>.
- [17] C. Guo, "Density matrix renormalization group algorithm for mixed quantum states," *Physical Review B*, vol. 105, no. 19, May 2022. DOI: 10.1103/physrevb.105.195152. [Online]. Available: <https://doi.org/10.1103/physrevb.105.195152>.
- [18] U. Schollwöck, "The density-matrix renormalization group," *Reviews of Modern Physics*, vol. 77, no. 1, pp. 259–315, Apr. 2005. DOI: 10.1103/revmodphys.77.259. [Online]. Available: <https://doi.org/10.1103/revmodphys.77.259>.
- [19] J. Hauschild and F. Pollmann, "Efficient numerical simulations with tensor networks: Tensor network python (TeNPy)," *SciPost Physics Lecture Notes*, Oct. 2018. DOI: 10.21468/scipostphyslectnotes.5. [Online]. Available: <https://doi.org/10.21468/scipostphyslectnotes.5>.
- [20] C. Bravo-Prieto, J. Lumbreras-Zarapico, L. Tagliacozzo, and J. I. Latorre, "Scaling of variational quantum circuit depth for condensed matter systems," *Quantum*, vol. 4, p. 272, May 2020. DOI: 10.22331/q-2020-05-28-272. [Online]. Available: <https://doi.org/10.22331/q-2020-05-28-272>.
- [21] F. G. Brandão, W. Chen, N. Hunter-Jones, R. Kueng, and J. Preskill, "Models of quantum complexity growth," *PRX Quantum*, vol. 2, no. 3, Jul. 2021. DOI: 10.1103/prxquantum.2.030316. [Online]. Available: <https://doi.org/10.1103/prxquantum.2.030316>.
- [22] A. Kandala, A. Mezzacapo, K. Temme, *et al.*, "Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets," *Nature*, vol. 549, no. 7671, pp. 242–246, Sep. 2017. DOI: 10.1038/nature23879. [Online]. Available: <https://doi.org/10.1038/nature23879>.
- [23] A. Peruzzo, J. McClean, P. Shadbolt, *et al.*, "A variational eigenvalue solver on a photonic quantum processor," *Nature Communications*, vol. 5, no. 1, Jul. 2014. DOI: 10.1038/ncomms5213. [Online]. Available: <https://doi.org/10.1038/ncomms5213>.
- [24] E. Farhi, J. Goldstone, and S. Gutmann, *A quantum approximate optimization algorithm*, 2014. arXiv: 1411.4028 [quant-ph].
- [25] L. Zhou, S.-T. Wang, S. Choi, H. Pichler, and M. D. Lukin, "Quantum approximate optimization algorithm: Performance, mechanism, and implementation on near-term devices," *Physical Review X*, vol. 10, no. 2, Jun. 2020. DOI: 10.1103/physrevx.10.021067. [Online]. Available: <https://doi.org/10.1103/physrevx.10.021067>.

- [26] G. Vidal, "Class of quantum many-body states that can be efficiently simulated," *Physical Review Letters*, vol. 101, no. 11, Sep. 2008. DOI: 10.1103/physrevlett.101.110501. [Online]. Available: <https://doi.org/10.1103%2Fphysrevlett.101.110501>.
- [27] R. Haghshenas, J. Gray, A. C. Potter, and G. K.-L. Chan, "Variational power of quantum circuit tensor networks," *Physical Review X*, vol. 12, no. 1, Mar. 2022. DOI: 10.1103/physrevx.12.011047. [Online]. Available: <https://doi.org/10.1103%2Fphysrevx.12.011047>.
- [28] F. Barratt, J. Dborin, M. Bal, V. Stojovic, F. Pollmann, and A. G. Green, "Parallel quantum simulation of large systems on small NISQ computers," *npj Quantum Information*, vol. 7, no. 1, May 2021. DOI: 10.1038/s41534-021-00420-3. [Online]. Available: <https://doi.org/10.1038%2Fs41534-021-00420-3>.
- [29] P. Calabrese, F. H. L. Essler, and M. Fagotti, "Quantum quench in the transverse-field ising chain," *Phys. Rev. Lett.*, vol. 106, p. 227 203, 22 Jun. 2011. DOI: 10.1103/PhysRevLett.106.227203. [Online]. Available: <https://link.aps.org/doi/10.1103/PhysRevLett.106.227203>.
- [30] S. Gopalakrishnan and A. Lamacraft, "Unitary circuits of finite depth and infinite width from quantum channels," *Physical Review B*, vol. 100, no. 6, 2019. DOI: 10.1103/physrevb.100.064309.
- [31] A. V. Uvarov, A. S. Kardashin, and J. D. Biamonte, "Machine learning phase transitions with a quantum processor," *Phys. Rev. A*, vol. 102, p. 012 415, 1 Jul. 2020. DOI: 10.1103/PhysRevA.102.012415. [Online]. Available: <https://link.aps.org/doi/10.1103/PhysRevA.102.012415>.
- [32] S.-H. Lin, R. Dilip, A. G. Green, A. Smith, and F. Pollmann, "Real- and imaginary-time evolution with compressed quantum circuits," *PRX Quantum*, vol. 2, no. 1, 2021. DOI: 10.1103/prxquantum.2.010342.
- [33] Eremenko, *Spectral theorems for hermitian and unitary matrices - purdue university*, Oct. 2017. [Online]. Available: <https://www.math.purdue.edu/~eremenko/dvi/spectral.pdf>.
- [34] F. Verstraete, M. Popp, and J. I. Cirac, "Entanglement versus correlations in spin systems," *Physical Review Letters*, vol. 92, no. 2, 2004. DOI: 10.1103/physrevlett.92.027901.
- [35] L. N. Smith, "Cyclical learning rates for training neural networks," *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2017. DOI: 10.1109/wacv.2017.58.
- [36] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven, "Barren plateaus in quantum neural network training landscapes," *Nature Communications*, vol. 9, no. 1, 2018. DOI: 10.1038/s41467-018-07090-4.
- [37] R. V. L. Hartley, "Transmission of information1," *Bell System Technical Journal*, vol. 7, no. 3, pp. 535–563, 1928. DOI: <https://doi.org/10.1002/j.1538-7305.1928.tb01236.x>. eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1002/j.1538-7305.1928.tb01236.x>. [Online]. Available: <https://onlinelibrary.wiley.com/doi/abs/10.1002/j.1538-7305.1928.tb01236.x>.

- [38] A. Nahum, J. Ruhman, S. Vijay, and J. Haah, "Quantum entanglement growth under random unitary dynamics," *Phys. Rev. X*, vol. 7, p. 031016, 3 Jul. 2017. DOI: 10.1103/PhysRevX.7.031016. [Online]. Available: <https://link.aps.org/doi/10.1103/PhysRevX.7.031016>.
- [39] M. Sonmez, E. Barker, J. Kelsey, K. McKay, M. Baish, and M. Boyle, *Recommendation for the entropy sources used for random bit generation*, en, 2018-01-10 2018. DOI: <https://doi.org/10.6028/NIST.SP.800-90b>.