Implementing Tensor Network Algorithms on Quantum Computers

James Dborin

Department of Physics & Astronomy
University College London

A thesis presented for the degree of
Doctor of Philosophy

Declaration

I, James Dborin, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Abstract

Quantum computing is an exciting area of research with potentially significant impacts on studying physics, developing new medicines, and understanding complex materials. However, current quantum hardware is too noisy to achieve the anticipated speed-ups that quantum computing promises over classical computers. While waiting for powerful devices that can perform quantum error correction, so-called NISQ devices will become available with the capability to perform computations beyond those possible classically. However, whether these NISQ devices will permit the acceleration of a scientifically or industrially relevant problem is still unknown. Tensor network methods are state-of-the-art techniques for the simulation of 1D and 2D quantum systems. NISQ circuits can generate and manipulate tensor network states with larger bond dimensions than can be simulated classically. This thesis explores methods to map insights and algorithms from the classical tensor network toolbox to NISQ devices to extend and improve NISQ-powered quantum simulation. This thesis outlines algorithms for representing and time-evolving infinite, translationally-invariant quantum states on finite NISQ devices utilising translationally invariant MPS states. This thesis also outlines an initialisation technique for variational quantum algorithms based on pre-training classical matrix product states and demonstrates its effectiveness for quantum simulation and machine learning.

Impact Statement

The work presented in this thesis relates to the simulation of tensor network states on near-term quantum devices. More specifically, we introduce algorithms for the simulation of large quantum states on small near-term quantum devices and novel state-of-the-art initialisation schemes for quantum neural networks. Tensor network algorithms are amongst the best techniques for the classical simulation of quantum systems. This thesis uses ideas from classical tensor network methodology to permit the simulation of systems larger than the physical device used to simulate them. This brings closer the simulation of large systems on small near-term devices. Examples of such large systems are translationally invariant condensed matter systems, large molecules, and complex materials. The new initialisation technique introduced in this thesis is more effective than the current state-of-the-art at producing trainable quantum neural networks. One of the more speculative yet potentially impactful applications of NISQ devices is accelerating machine learning. Quantum machine learning techniques are studied widely within the academic and industrial community for uses ranging from optimisation of supply chains to identification of promising molecules for medicinal applications. One of the major stumbling blocks in this research is the difficulty in training large quantum circuits. The techniques introduced in this thesis increase the trainability of quantum neural networks, potentially moving towards a quantum advantage for machine learning and optimisation with near-term devices.

Acknowledgements

I must firstly thank my supervisor, Prof. Andrew Green, for three and a bit years of encouragement and guidance through this thesis and other work in the three years not included within. I would also like to thank the wider *Green group*, all of whom have contributed to a rich environment to work in.

I would like to thank my family for unending support and patience during endless rambling about quantum-whatevers. Meryem, my mother, and the Arik family have been an endless supply of support.

Finally I would like to thank the friends made along the journey. Fergus, Alexis, and Oscar have all helped pass the time with interesting discussions, fruitful collaborations, and much needed distractions.

Contents

1 Introduction **7**
1.1 Outline of Thesis . . . 8
1.2 Motivation . . . 9
1.3 Preliminaries . . . 10
1.3.1 Quantum Computing . . . 10
1.3.2 Tensor Networks . . . 11
1.4 Summary . . . 15

2 Representing QMPS **17**
2.1 MPS as Quantum Circuits . . . 18
2.1.1 Finite MPS . . . 20
2.1.2 Translationally Invariant MPS . . . 22
2.2 Optimizing Quantum MPS . . . 30
2.3 NISQ Implementation . . . 32
2.3.1 Experiment . . . 32
2.3.2 Results . . . 34
2.3.3 Error Mitigation Techniques . . . 34
2.4 Summary . . . 38

3 Time Evolving QMPS **39**
3.0.1 Pseudocode for Finite MPS Time Evolution . . . 41
3.1 Translationally Invariant Variational Time-Evolution . . . 42
3.1.1 Time Evolution Simulations . . . 52
3.1.2 Power Method Approximation . . . 56
3.2 Summary . . . 68

4 Extensions **69**
4.1 Mid Circuit Measurement . . . 70

4.2 Qubit Efficient Time Evolution 73
4.3 Generating Symmetric States 75
4.3.1 Thermofield States 75
4.3.2 Quantum Thermofield MPS 77
4.4 Summary 80

5 MPS Initialisation 82
5.1 MPS Optimization 84
5.1.1 Imaginary Time Evolution 85
5.1.2 DMRG-Inspired Optimisation 87
5.2 Initialising Circuits with Quantum MPS 88
5.2.1 MPS Pre-training 88
5.2.2 Finding Ground States 90
5.2.3 Combinatorial Optimization 92
5.2.4 Image Classification 94
5.2.5 Discussion 96
5.2.6 Summary 101

6 Conclusion 103

Chapter 1

Introduction

In the beginning God said $i\hbar\frac{\partial}{\partial t}|\psi(t)\rangle = H|\psi(t)\rangle$. *This has made a lot of people very angry and been widely regarded as a bad move.* - Douglas Adams (sort of)

Quantum mechanics is the most effective description of the universe that humanity has come up with to date. Ignoring the rather thorny matter of how gravity fits into this whole picture, it is widely understood that at a fundamental level nature is quantum. To understand the world around us it is necessary to understand and manipulate quantum states of matter, except in the (many) cases where good theories can be constructed in the classical limit. This is unfortunately a case of something being easier said than done.

Simulating systems that behave quantum mechanically is notoriously difficult. Physicists who try to *shut up and calculate* quickly run into difficulty. As quantum systems grow large the resources needed to perfectly simulate the system grow *exponentially*. Anyone who has recently lived through a pandemic will be well aware of how quickly exponential growth becomes unmanageable. The growth in the resources required to simulate large dimensional systems is often referred to as the *curse of dimensionality*. The curse of dimensionality effectively sets a hard limit on the size of quantum systems that can be studied directly, where only a few dozen particles quickly saturate the capabilities of the world's largest supercomputers. This is rather paltry considering the $\sim 10^{23}$ particles that might be found in a single gram of any almost element.

Much effort has been spent trying to overcome the difficulty of simulating quantum systems. There are many exciting and potentially valuable applications that could be opened up with the ability to efficiently simulate quantum systems. The field of quantum chemistry searches for ways to push back against the exponential quantum wall to accelerate drug discovery[1]. The properties of solid-state materials are ultimately determined by the interactions between the many particles in the substance. A better ability to simulate complex quantum systems could lead to advances in difficult-to-solve materials science problems, such as designing better batteries[2]. Efficient simulation of quantum systems could teach us more about fundamental theories of physics, by directly probing nature in extreme environments; like around black holes; or in extremely high-energy environments, much like is already done with very large and expensive particle colliders.

1.1 Outline of Thesis

This thesis will be focusing on the problem of using quantum computers to simulate many-particle quantum systems. In particular, we focus mainly on studying quantum condensed matter systems, with a brief layover into quantum machine learning with so-called *quantum neural nets*. Below I outline the structure of this thesis:

Chapter 1 — The remainder of this chapter introduces quantum computing, and tensor networks as a classical tool for studying quantum condensed matter systems.

Chapter 2 — Introduces quantum tensor networks, and outlines work done in representing and optimising translationally invariant quantum matrix product states. The work in this chapter is based on publications [3, 4].

Chapter 3 — Develops a time evolution algorithm for quantum MPS, facilitating the efficient study of the dynamics of translationally invariant quantum matrix product states on quantum devices. The work in this chapter is based on publications[3, 4].

Chapter 4 — Considers ways to extend the results of the previous two chapters. We consider how to best translate quantum tensor networks to devices with all-to-all connectivity, and how to represent quantum thermofield states on quantum devices.

Chapter 5 — Outlines a quantum neural network initialisation scheme based on classical pre-training of classical tensor networks. The work in this chapter is based on the publication[5].

1.2 Motivation

The motivating concept behind this thesis is to use quantum tensor networks to enhance the study of quantum condensed matter systems and, perhaps more broadly, use insights from the tensor network field to advance the younger discipline of near-term quantum computing. The quantum tensor network framework combines state-of-the-art techniques in the classical simulation of quantum systems with the expressive power of quantum computing hardware, hoping both fields have something to gain from their union.

Effectiveness of classical tensor networks — Tensor network methods are amongst the best tools available for the classical simulation of quantum systems. In particular, the *Matrix Product State* (MPS)[6] has been highly successful for both theoretical insights and numerical simulations of 1D and 2D systems. Likewise, the *Multi-Scale Entanglement Renormalization Ansatz* (MERA)[7] states have provided insights into critical systems found in both condensed matter and quantum gravity research[8].

Quantum condensed matter as a target for NISQ devices — Quantum condensed matter problems are scalable to match the qubit count of qubit-restricted near-term hardware. Furthermore, condensed matter interactions are often local, which is appropriate for the limited connectivity of NISQ devices. Quantum chemistry problems, on the other hand, have highly non-local interactions[9]. Finally, classical methods in simulating condensed matter systems are very well understood, making it potentially easier to identify quantum speed-ups, which has proven difficult in previous claims of quantum advantage[10, 11].

Benefits of Quantum Tensor Networks — Quantum tensor networks have the potential to augment the classical tensor network approach with the increased expressiveness that quantum hardware brings. Quantum tensor networks might extend the capabilities of classical tensor networks and result in higher accuracy numerical simulations of interesting quantum systems[12, 13]. Inspired by classical tensor networks, quantum tensor network states explore regions of Hilbert space with limited entanglement. One of the fundamental insights underlying the success of the tensor network formalism in quantum mechanics has been understanding entanglement as a computational resource managed during simulation. Entanglement in quantum simulation can be translated to classical computing resources, which can then be allocated as required across the system being simulated. NISQ devices directly use entanglement as a resource for computation. The quantum tensor network formalism can hopefully help make better use of limited entanglement resources on near-term quantum devices.

1.3 Preliminaries

1.3.1 Quantum Computing

Quantum information processing involves the manipulation of *qubits* rather than *bits*[14]. Bits exist in one of two fixed states, 0 or 1. Qubits have the property that they can exist in a superposition of two states, e.g. $\alpha|0\rangle + \beta|1\rangle$. Interactions between qubits can generate *entanglement*, a uniquely quantum mechanical resource required for the applications of quantum computers.

Quantum computers may give a speedup for *quantum simulation*[15]. Quantum systems are challenging to simulate classically but may be simulatable by a programmable quantum computer via Hamiltonian simulation. There may also exist quantum speedups for classical problems. Algorithms exist for solving discrete logarithm problems[16], searching unstructured lists[17], and random walks on graphs[18]. All the above use cases are instances of the Quantum Singular Value Transform (QSVT)[19], a general framework for accelerating matrix manipulations with quantum devices.

Despite a solid theoretical understanding of these algorithms, implementing these algorithms presents a significant engineering challenge. Qubits are exposed to the environment and interact with systems outside the quantum device. These spurious interactions result in errors during the execution of these algorithms. It is possible to correct such errors using *quantum error correction*[20], where these interactions with the environment are identified and fixed during execution. Unfortunately, quantum error correction introduces significant overhead in the number of qubits and operations. Current best estimates are of 10^5 physical qubits per logical qubit and require additional computational steps[21] - far beyond the capability of current generation devices. Quantum devices are partitioned into two families, depending on whether or not they can perform error correction. The so-called Near-Term Intermediate Scale Quantum (NISQ)[22] devices can not perform error correction, whereas future devices may be able to.

NISQ devices are limited in qubit number, with current state-of-the-art[11] and soon-to-be-released[23] devices having $\sim$ 100 qubits. These devices are also noisy and can implement $\sim$ hundreds of operations before noise entirely corrupts the output. NISQ devices have now potentially reached the stage where they can perform computations that can not efficiently be classically simulated[11]. This may change as the classical simulation of quantum systems improves in response to such claims[10, 24]. The open challenge is to find a practically relevant problem that these NISQ devices can accelerate before error-corrected devices are available. The strongest candidate for NISQ-advantage seems to be quantum simulation. The Variational Quantum Eigensolver (VQE)[25, 26] algorithm attempts to find solutions to the time-independent Schrodinger Equation

$$H|\psi\rangle = E|\psi\rangle \tag{1.1}$$

where $|\psi\rangle$ is an eigenvector of the Hamiltonian of interest, and E is the associated energy level. Time evolution algorithms attempt to find solutions to the time-dependent Schrodinger equation

$$|\psi(t)\rangle = e^{-iHt}|\psi(0)\rangle \tag{1.2}$$

which defines the dynamics of a system. This thesis is centred around finding applications of NISQ devices to condensed matter physics, using insights from classical tensor network methods.

1.3.2 Tensor Networks

Tensor network methods are state-of-the-art techniques for simulating quantum lattices in 1D and 2D[27, 28, 29]. Tensor network methods revolve around viewing entanglement as a resource to be managed during numerical simulations of quantum systems. By restricting simulations to smaller regions of Hilbert space, it is possible to simulate even large quantum systems using classical computers. Accurate classical simulation is only possible for systems that have restricted entanglement, which is the case for many systems of interest.

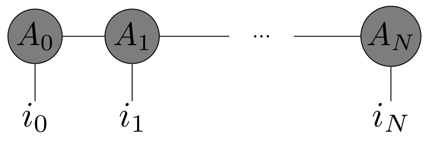
Matrix Product States

The most well studied class of tensor network states are the matrix product states (MPS)[28]. Amplitudes of MPS are given by

$$|\psi_{i_0,i_1,\ldots,i_N}\rangle = Tr[\mathbf{A}_{i_0}\mathbf{A}_{i_1}\ldots\mathbf{A}_{i_N}]|i_0 i_1 \ldots i_N\rangle \tag{1.3}$$

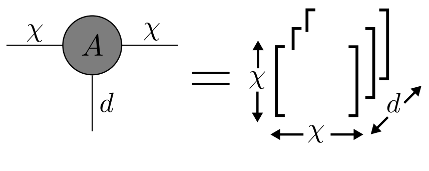
where $\mathbf{A}_n$ is a complex valued matrix of size $\chi \times \chi$, where χ is known as the bond dimension of the MPS. Tensor networks can be expressed using a very convenient graphical notation known as *Penrose notation*[30]. In this graphical notation matrix product states are given by

$$|\psi_{i_0,i_1,\ldots,i_N}\rangle =$$



(1.4)

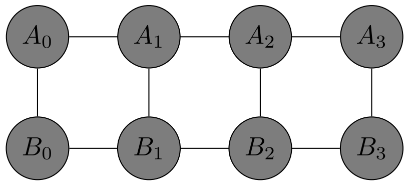
where the nodes of the network are tensors, and connected lines indicate multiplication of tensors along the connected dimensions. A single MPS site is described by a $d \times \chi \times \chi$ tensor



(1.5)

where d is the physical dimension of the site, and χ is known as the bond dimension. The bond dimension of an MPS is a parameter that can be adjusted to control the expressiveness of the state. Large bond dimension MPS can capture higher entanglement states, but require more computational resources to do so.

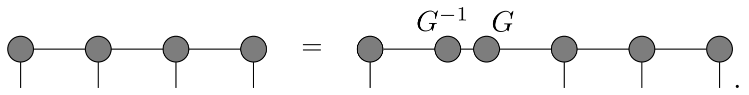
Low bond dimension matrix product states are efficiently manipulated with classical computers. For an N site MPS, with physical dimension d and bond dimension χ , only $Nd\chi^2$ complex numbers are needed to store the state, one complex number for each element in each tensor of the network. A general quantum state on N sites with a physical dimension d requires d^N complex numbers to store. Matrix product states also permit the efficient manipulation of quantum states. For example, the inner product between two MPS requires $\mathcal{O}(Nd\chi^3)$ operations to calculate, with an exponential improvement in N compared to the naive computation. The overlap between a pair of 4-site states, $|\psi_A\rangle$ and $|\psi_B\rangle$, is given by

$$\langle\psi_B|\psi_A\rangle =$$ 

(1.6)

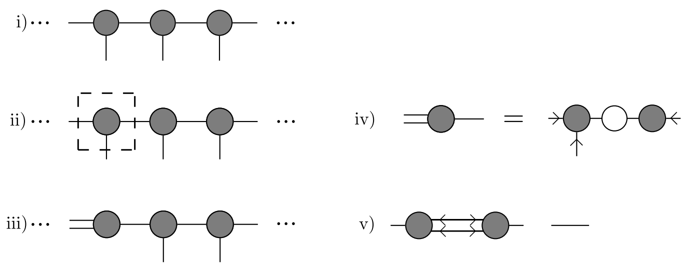
MPS Canonical Form

Matrix product states have a gauge freedom in that they are invariant under the transformation $\mathbf{A} \rightarrow \mathbf{G}^{-1}\mathbf{A}\mathbf{G}$. These gauge transformations are given by inserting resolutions of the identity between pairs of tensors,



(1.7)

This gauge freedom allows MPS to be constructed that are particularly compact and easy to manipulate. One such gauge choice puts the MPS in *canonical form*[6]. Canoncial form can be constructed with a series of Singular Value Decompositions (SVDs) along the lattice sites. Canonicalisation proceeds by identifying a site (1.8ii) in the chain, and reshaping it by combining an auxiliary leg with the physical one. (1.8iii). The reshaped tensor is decomposed using an SVD (1.8iv), where the outer nodes are isometrics (1.8v) and the inner node is a diagonal tensor where the entries are the singular values of the tensor across the separated legs.



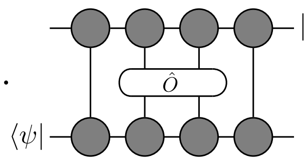
(1.8)

Absorbing the diagonal tensor to the node on the right and repeating this process with the next pair of sites gives a procedure to put a matrix product state in *left canonical form*. Any MPS can efficiently be put into this form without changing the state. Similarly there exist *right* and *mixed canonical form* constructions, which involve sweeping from right to left, or left to right and back again.

Translationally Invariant MPS

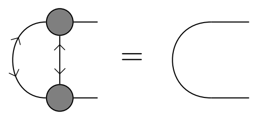
The utility of the canonical form is apparent when considering the tensor network to calculate the expectation value of a local operator in an infinite translationally invariant MPS

$$\langle\psi|\hat{O}|\psi\rangle = \cdots$$



$$\cdots \tag{1.9}$$

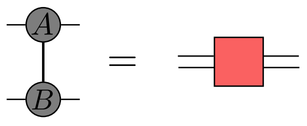
where there are an infinite number of sites on both sides of the operator. Naively this would be impossible to calculate, as it would require the contraction of infinitely many tensors. However the isometry condition in Eqn. 1.8v allows the nodes left of operator can be simplified using



(1.10)

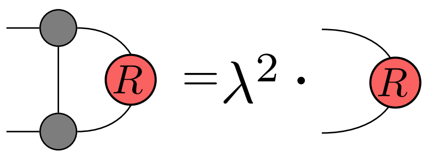
where the right hand side is the identity matrix. The right hand side of the operator can also be greatly simplified. The nodes to the right of the operator form an infinite chain of matrix multiplications, where the multiplied matrix is given by the *transfer matrix*, E^A_B,

$$E^A_B =$$



(1.11)

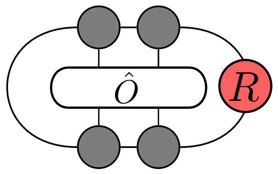
which is equivalent to the red node. A and B refer to the MPS tensors that make up the transfer matrix. A and B will be the same when calculating expectation values, however for situations like time evolution which will be presented later, they are not necessarily the same. Multiplying a matrix many times has the effect of projecting any vector onto the largest weight eigenvector of that matrix. We can replace the infinite multiplications of the transfer matrix with the largest right eigenvector of E, denoted R, defined using the fixed point equation



(1.12)

where λ is the largest singular value of the mixed transfer matrix. Now the calculation of this expectation value is given by the much simpler expression

$$\langle\psi|\hat{O}|\psi\rangle =$$



(1.13)

This way of expressing a translationally invariant MPS forms the basis of the following section, which outlines how to simulate such translationally invariant states on NISQ devices.

1.4 Summary

This section has introduced the two major fields of study underlying this thesis. NISQ devices are the current state-of-the-art quantum hardware, however have limitations on what computations they are able to perform. It is of great scientific and industrial interest to figure out if there are any computations that can be efficiently and accurately performed by a NISQ device that cannot be done classically. Tensor network methods allow for the efficient manipulation of exponentially small sets of quantum states, which are non-the-less of relevance for studying many physical systems. The main takeaway from this thesis will be that insights from the tensor network field can be used to improve the performance of NISQ devices. The following chapters introduce NISQ algorithms and methods inspired by classical tensor networks. Firstly methods are introduced for representing and time evolving translationally invariant states on NISQ devices, drawing inspiration from classical simulations of translationally invariant MPS. Then an algorithm for initialising quantum neural networks is introduced using classical MPS machine learning.

Chapter 2

Representing QMPS

This chapter is based on the publications; Parallel Quantum Simulation of Large Systems on Small Quantum Devices[3], and Simulating groundstate and dynamical quantum phase transitions on a superconducting quantum computer[4]. In these works, we developed the framework for generating translationally invariant matrix product states on quantum computers and implemented these on near-term quantum hardware. In [3], we outline how to construct a quantum circuit that can represent any translationally invariant MPS in canonical form, by efficiently solving a fixed point equation on device. We use this construction to define a subset of translationally invariant MPS that can be constructed using shallow quantum circuits, and argue for the possibility of exponential advantage in the simulation of these states. We use these states as ansatz for calculating the ground states of translationally invariant Hamiltonians. In this work I helped develop the efficient quantum implementation of the fixed point equation, and adapted this to work as a target for VQE. I also worked on the implementation of all the numerics used in the work. In [4] we implement translationally invariant MPS on near-term hardware and achieve very good translationally invariant ground-state estimates using multiple error-mitigation strategies. In this work I implemented the quantum circuits and error mitigation techniques used on device, and adapted the ground state optimizations to work better on a noisy device.

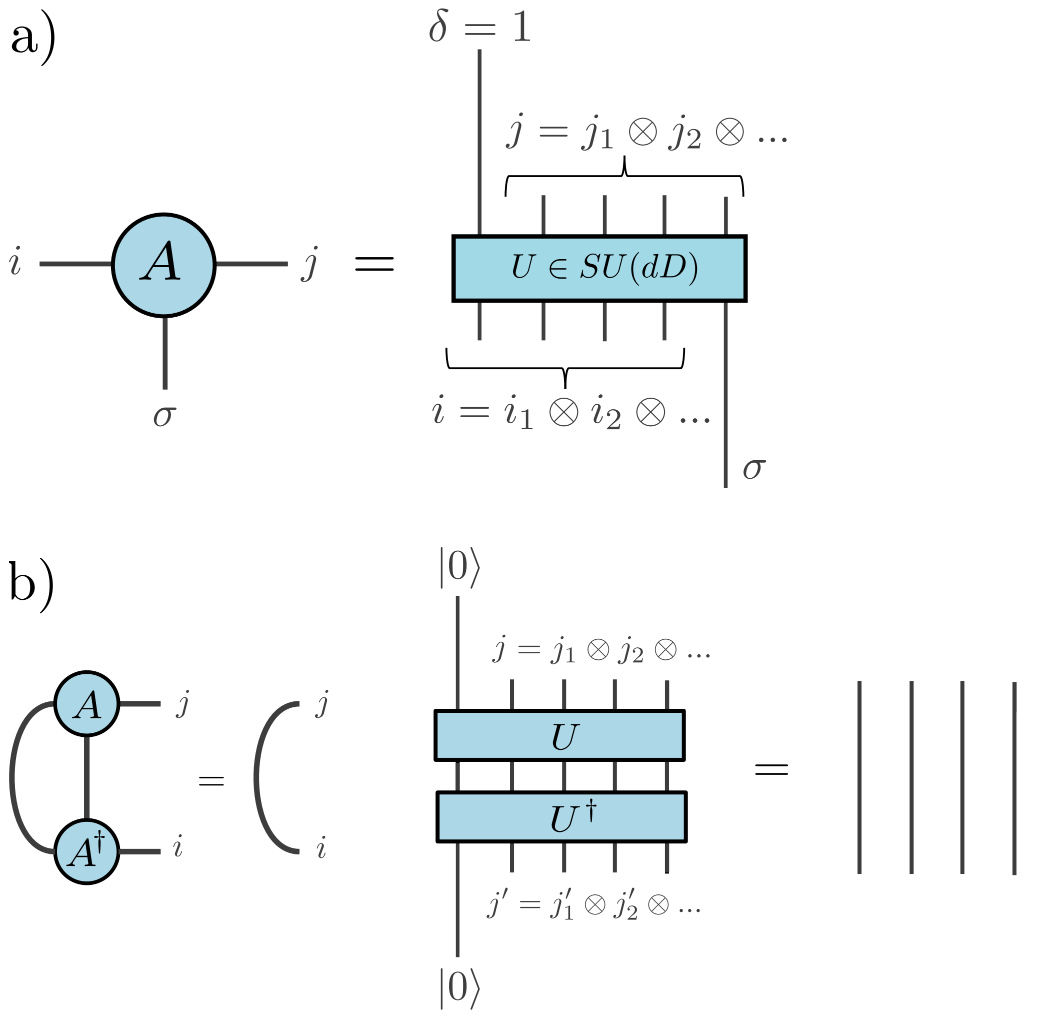
Despite the anticipation that quantum computers may accelerate a wide range of computational tasks, many problems are out of reach for near-term quantum hardware to deliver an advantage over classical approaches. Finding challenging problems to simulate classically but which NISQ devices may provide an advantage is an active area of research. As well as benchmarking the state of NISQ devices, such problems could benefit from quantum advantage once powerful enough devices are available.

Condensed matter systems are a good candidate for such problems. Condensed matter problems are readily scalable to match the qubit count of the target device, often have local interactions that are well suited to the architecture of NISQ devices, and yet still retain scientific and technological relevance. Strongly correlated condensed matter systems are among the most complex condensed matter systems to simulate classically and hence are the most likely to benefit from a quantum advantage. One strongly correlated phenomenon is quantum criticality[31, 32, 33]. Quantum criticality refers to systems near zero temperature phase transitions. These systems exhibit diverging correlation lengths near criticality, which can be challenging for classical numerics due to finite size scaling effects. Tensor networks are amongst the best numerical tools to study strongly correlated quantum systems. In particular, tensor networks are suitable for studying critical systems as tensor network states can be studied directly in the thermodynamic limit, eliminating errors due to finite scaling. Unlike other classical numeric techniques, many tensor networks are equivalent to quantum circuits[34], and this mapping can potentially yield quantum advantage for tensor network simulations. The following chapter details how to perform this translation for both finite and translationally invariant MPS and demonstrates that these translationally invariant quantum MPS can capture properties of critical ground states on current NISQ devices.

The previous section outlined algorithms for manipulating translationally invariant tensor networks by putting them in canonical form and using this simplified representation to simulate infinitely large systems efficiently. This simplified representation is equivalent to a class of quantum circuits which perform identical computations. The following section outlines this translation and discusses the potential for accelerating these computations on quantum hardware for large bond dimension tensor networks.

2.1 MPS as Quantum Circuits

This section outlines the connection between MPS and quantum circuits. Any MPS has an equivalent canonical form, where all the tensors in the



a)
$\delta = 1$
$j = j_1 \otimes j_2 \otimes ...$
i
A
j
$=$
$U \in SU(dD)$
σ
$i = i_1 \otimes i_2 \otimes ...$
σ
b)
$|0\rangle$
$j = j_1 \otimes j_2 \otimes ...$
A
j
$=$
j
U
$=$
$U^\dagger$
$A^\dagger$
i
i
$j' = j'_1 \otimes j'_2 \otimes ...$
$|0\rangle$

Figure 2.1: **Translation between MPS and Quantum Circuits:**a) The translation between an MPS tensor in canonical form and a unitary matrix. Given a bond dimension D MPS tensor $\log_2(D)$ qubits are needed to encode the axillary legs of the tensor. b) The left canonical condition in classical MPS is automatically satisfied by unitary matrices.

MPS are isometries. Isometries are equivalent to unitary matrices acting on reference input states, Fig. 2.1. For a MPS with bond dimension χ the unitary embedding requires $\log_2(\chi)+1$ qubits. One qubit is required to act on the reference state, and the remaining qubits act as a binary expansion of the bond dimension of the tensor to index the auxiliary state. Unitary matrices automatically satisfy the left canonical condition, shown graphically in Fig. 2.1b.

2.1.1 Finite MPS

Given this mapping, we can now construct a quantum circuit equivalent to a MPS. Staircase circuits generate quantum MPS states. Left and right canonical MPS are equivalent to staircase circuits rising in opposite directions. For MPS with a bond dimension larger than two, the unitaries will act more than two qubits. Fig 2.2 gives an example of an MPS in left canonical form, with a maximum bond dimension of four, as a quantum circuit. This bond dimension four MPS circuit requires interactions across three qubits to simulate the state tensors directly. Most NISQ devices are limited to nearest neighbour interactions. Fig 2.2 shows one way of decomposing high bond dimension tensors into local gates which preserves the high bond dimension. This *reverse staircase ansatz* is but one choice of decomposition; for gates acting over many qubits, one can imagine many such decompositions. Therefore even when limited to local interactions, capturing high bond dimension MPS states is still possible, but shallow nearest-neighbour circuits capture only a subset of all high bond dimension MPS. The expressiveness of these local high bond dimension MPS is still an open question, although there is evidence that quantum tensor networks may achieve speed-ups over their classical counterparts. Shallow quantum tensor network states have a high overlap with states generated during time evolution and are potentially more expressive per parameter than densely represented MPS circuits[12, 13]. One can tentatively hope such states could achieve quantum advantage, performing approximate groundstate optimization and time evolution using extremely high bond dimension MPS. Naively one may assume that such decompositions give the possibility of an exponential advantage over classical MPS simulations. The computational resources requires to simulate a bond dimension χ circuit grow only as $\mathcal{O}(log(\chi))$, whereas classically the difficulty is $\mathcal{O}(poly(\chi))$. To exactly simulate a system of size N, in general, requires a bond order $\mathcal{O}(exp(N))$, making high accuracy simulations of large systems

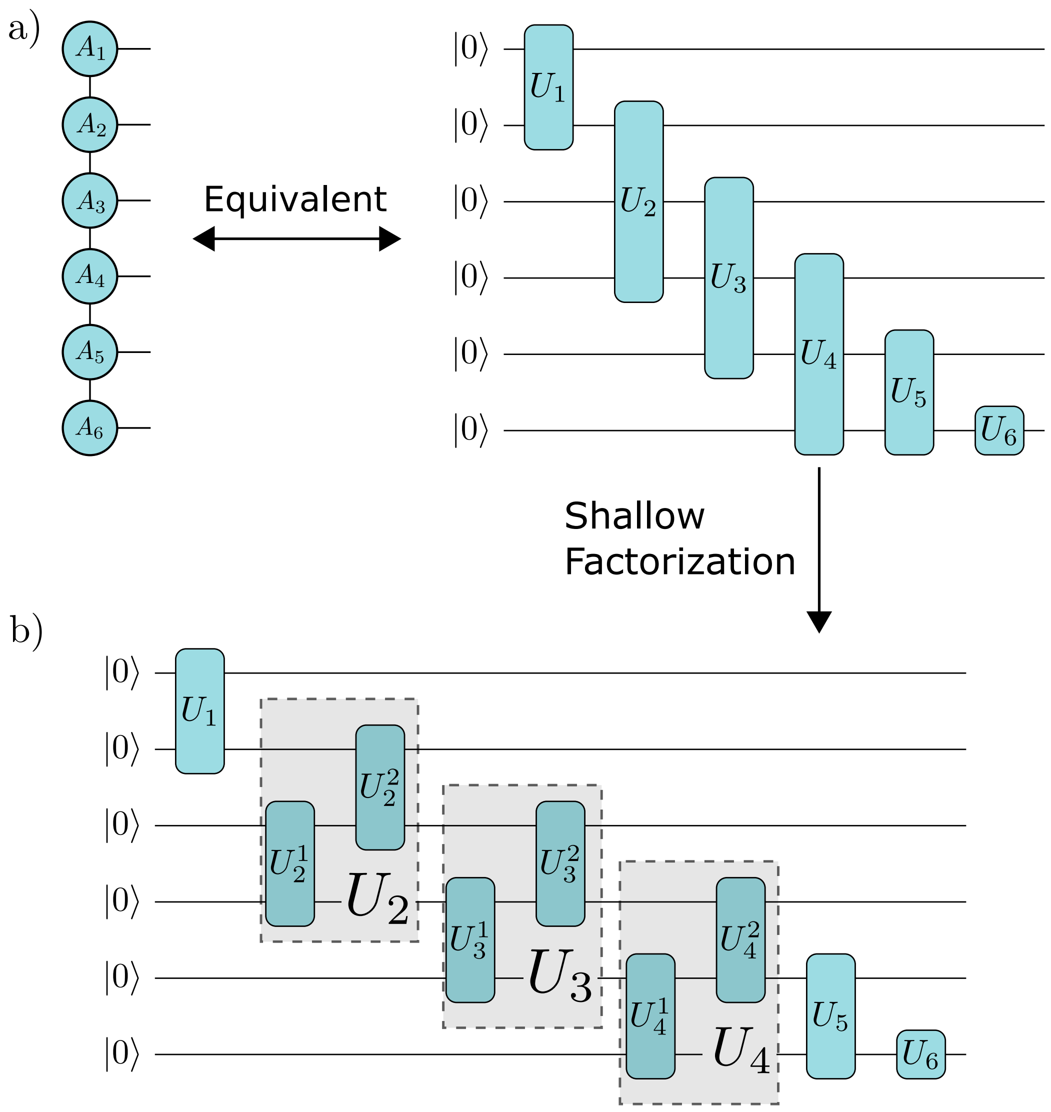


Figure 2.2: **Representing Finite MPS on a Quantum Circuit:** a) A bond dimension 4 MPS in canonical form can is equivalent to quantum circuit. The bond dimension 4 tensors are represented using 3 qubit unitary matrices. b) Multi-qubit matrices can be approximately decomposed into local circuits. Shown here is a *reverse stair-case* decomposition, shown to be able to effectively express states generated during time evolution.

in general impossible. However, since Quantum MPS states on NISQ devices will be limited to subsets of high bond order MPS generated by low depth quantum circuits, accurate simulations of some large, highly entangled systems may still be out of reach.

2.1.2 Translationally Invariant MPS

Simulating translationally invariant systems at first glance appears to be a difficult task for qubit-limited devices. Although the system is infinitely large, typically, one can get a handle on calculations in the translationally invariant limit by studying larger systems and extrapolating to the infinite case[35, 36]. This procedure results in undesirable finite scaling approximation errors. Notice that for calculations of interest, such as finding the expectation value of an operator on a small number of sites, it is possible to split the infinite chain into terms to the left and right of the site of interest. When using MPS, calculations like this can be significantly simplified. When the MPS is in canonical form, terms to the left of the site vanish, which is precisely the left canonical condition satisfied by quantum MPS tensors. In this situation, terms to the site's right do not vanish. However the influence of these terms is mediated via the partition highlighted in Fig 2.3a. Hence the influence of the infinite chain can be replaced by a single unitary acting on only $2\log_2(D)$ qubits, where D is the number of non-zero Schmidt coefficients across the partition. The tensor that summarises the influence of the semi-infinite chain of unitary gates acting to the right of the main site is called the *environment*, shown as V in Fig 2.3b. In the case of translationally invariant MPS states V is given by the *fixed point equation* given in Fig 2.3c. This is the same equation as shown earlier in Section 1, Eqn. 1.12.

Solving the Fixed Point Equation

To faithfully represent a translationally invariant MPS on a quantum circuit, it is necessary to identify an environment tensor that satisfies the fixed point equation. One method to solve this is to vary a parametrised decomposition of V for a fixed state tensor U. The variational algorithm proceeds by calculating the distance between the LHS and RHS of Fig 2.3c at each proposed parameter and updating the parameters by calculating the gradient of this distance. This distance can be evaluated efficiently with shallow quantum circuits using the *Destructive SWAP* test[37]. The Destructive SWAP test

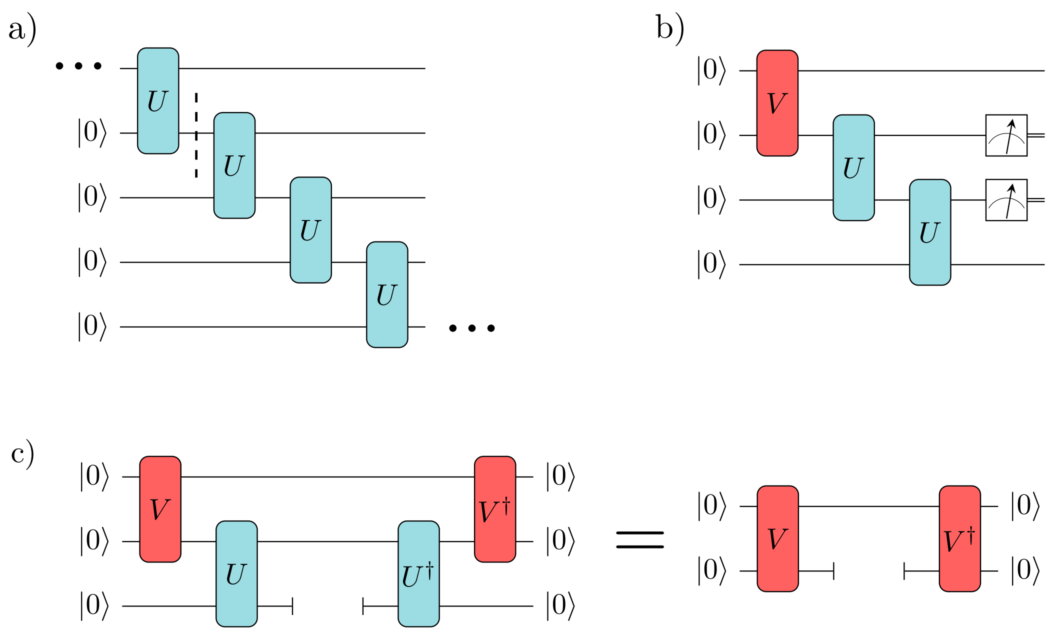


Figure 2.3: **Translationally Invariant MPS:** a) Translationally invariant MPS are represented as infinitely deep and wide quantum circuits. Sites above the dashed line can be replaced with a single tensor summarizing their influence on local observables. b) The effect of the gates to the top left of an operator can be summarized by a single *environment* tensor. The size of this tensor is determined by the entanglement between the state tensor and the semi-infinite chain of gates acting before it. c) The V tensor is the solution to the fixed point equation shown here.

acting on two reduced density matrices measures the trace of the product of the two density matrices,

$$DESTRUCTIVE\ SWAP(\sigma, \rho) = Tr[\sigma\rho] \tag{2.1}$$

The output of the destructive swap test is two bit strings, $\mathbf{O^1}$ and $\mathbf{O^2}$. An example of this can be seen in Fig 2.4. The CNOT and Hadamard gates constitute the SWAP test, and the measured bit strings are used to calculate the trace of the product of the reduced density matrices. Given the output strings $\mathbf{O^1}$ and $\mathbf{O^2}$ the result of the SWAP test is given by the probability that the dot product of the output bits is even,

$$Tr[\sigma\rho] = 2P(\mathbf{O^1} \cdot \mathbf{O^2}\ mod\ 2 == 0) - 1 \tag{2.2}$$

The destructive SWAP test is used as a subroutine to calculate the *Trace Distance, D*

$$D(\rho, \sigma) = Tr[(\rho - \sigma)^2] \tag{2.3}$$

where ρ and σ are the reduced density matrices being compared. Using the linearity of the trace this is equal to

$$D(\rho, \sigma) = Tr[\rho^2] + Tr[\sigma^2] - 2Tr[\sigma\rho] \tag{2.4}$$

where each term in 2.4 can be evaluated with a destructive SWAP test. Fig 2.4 gives the three terms in the cost function evaluated with a destructive SWAP test for a bond dimension 4 environment. These three circuits can be executed in parallel or sequentially on the device.

Parametrising D=2 Environments

The environment of a translationally invariant MPS is equivalent to the largest eigenvector of the MPS transfer matrix. This matrix, R, is a trace 1 Hermitian matrix. V embeds the Cholesky decomposition, r, of the environment tensor. The Cholesky decomposition of a trace 1 matrix will have a Frobenius norm of 1. We can parametrise the singular values of any 2-by-2 matrix with Frobenius norm 1 with $\cos(\gamma)$ and $\sin(\gamma)$. Taking the SVD of the matrix $r = W^\dagger \Lambda W$ with the parametrisation

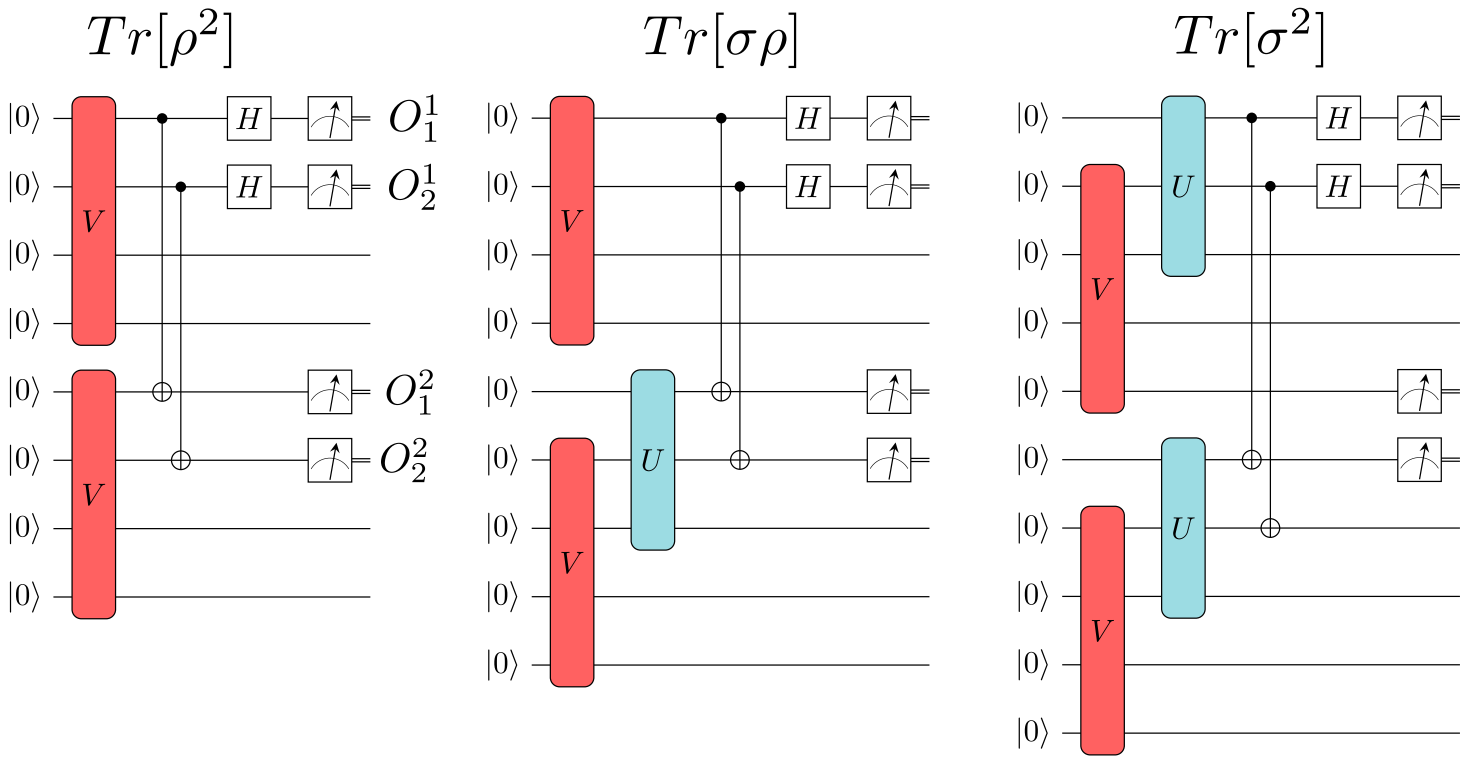
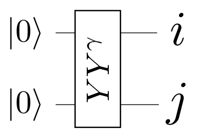
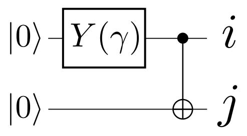


Figure 2.4: **Shallow Trace Distance Circuits:** The three circuits needed to calculate the trace distance between the LHS and RHS in equation given in Fig 2.3c. V is the correct environment for a state tensor U if the trace distance $Tr[(\rho(V) - \sigma(U,V))^2] = 0$. Shown here are the circuits to find a U, V pair for a bond dimension 4 MPS.

$$\Lambda_{ij} = \begin{pmatrix} \cos(\gamma) & 0 \\ 0 & i\sin(\gamma) \end{pmatrix} =$$



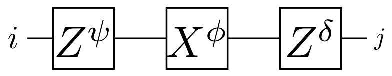
$=$



(2.5)

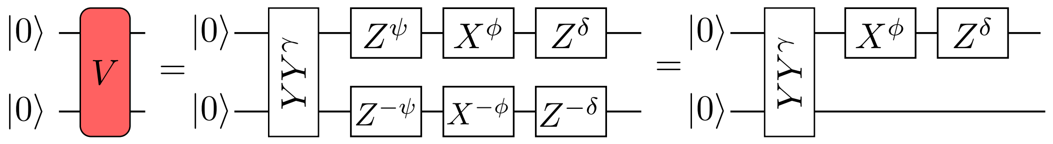
and decomposing the single qubit rotations W using

$$W_{ij} =$$



(2.6)

gives an exact parametrisation for bond dimension 2 environments as



(2.7)

where the last equality uses the fact that all diagonal gates commute, allowing the two Z gates to be pulled through the diagonal YY gate and cancel, and absorbing the single qubit rotations into the state tensor U.

Pseudocode for Translationally Invariant MPS

```python
import cirq
import numpy as np

# 4 qubits for 2 site translationally invariant MPS
qubits = cirq.LineQubit.range(4)

# 11 params, 4 for each U, 3 in V
p = np.random.rand(11)

circuit = cirq.Circuit([
    # Environment circuit
    cirq.YY.on(qubits[0], qubits[1])**p[0],
    cirq.X.on(qubits[1])**p[1],
    cirq.Z.on(qubits[1])**p[2],

    # U1
    cirq.Z.on(qubits[1]) ** p[3],
    cirq.X.on(qubits[1]) ** p[4],
    cirq.Z.on(qubits[2]) ** p[5],
    cirq.X.on(qubits[2]) ** p[6],
    cirq.CNOT(qubits[1], qubits[2])

    # U2
    cirq.Z.on(qubits[2]) ** p[7],
    cirq.X.on(qubits[2]) ** p[8],
    cirq.Z.on(qubits[3])) ** p[9],
    cirq.X.on(qubits[3]) ** p[10],
    cirq.CNOT(qubits[2], qubits[3])
  ]
)
```

Listing 2.1: Example python pseudocode for optimizing the right environment using the simulation engine Cirq and scipy optimize. A circuit is created for each of the three terms. The circuits are simulated and the results processed. This is dont in the inner loop of an optimization. The final parameters represent the D

Pseudocode for finding D=2 Environments

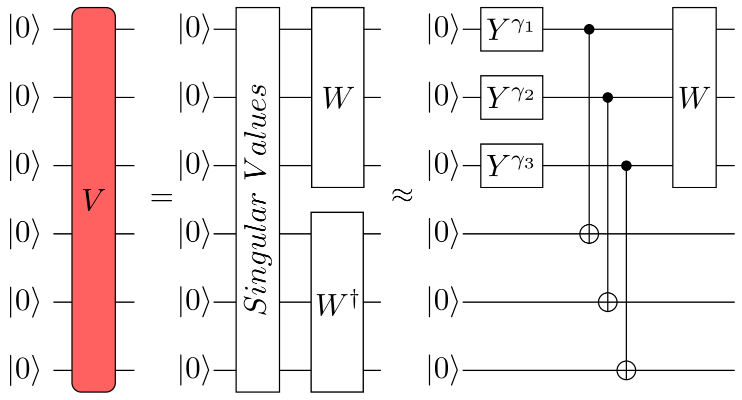
```
import cirq
from scipy.optimize import minimize
import nump as np

# set parameters
gamma, phi, delta = ...

# create the qubits
VV_qubits = cirq.LineQubit.range(4)
UVV_qubits = cirq.LineQubit.range(5)
UVUV_qubits = cirq.LineQubit.range(6)

sim = cirq.Simulator()

def simulate_circuit(gamma, phi, delta, sim)
  VV_circuit = cirq.Circuit(
    # first copy of V
    cirq.YY(VV_qubits[0:2]) ** gamma,
    cirq.X(VV_qubits[0]) ** phi,
    cirq.Z(VV_qubits[0]) ** delta,

    # second copy of V
    cirq.YY(VV_qubits[2:]) ** gamma,
    cirq.X(VV_qubits[2]) ** phi,
    cirq.Z(VV_qubits[2]) ** delta,

    # SWAP test
    cirq.CNOT(VV_qubits[0], VV_qubits[2]),
    cirq.H(VV_qubits[0])
    cirq.measure(VV_qubits[0], VV_qubits[2])
  )

  # simulate the circuit
  result = sim.run(circuit, repetitions=20)

  # postprocess the results
  positive_score_VV =  ( np.dot(result[0], result[1]) % 2 )
  == 0 / len(result)

  trace_VV = 2 * positive_score - 1

  # do the same for the other terms:
  ...
```

```
    return trace_vv + trace_uvuv - 2*(trace_uvv)

  res = minimize(simulate_circuit, [gamma, phi, delta])
```

Listing 2.2: Example python pseudocode for optimizing the right environment using the simulation engine Cirq and scipy optimize. A circuit is created for each of the three terms. The circuits are simulated and the results processed. This is dont in the inner loop of an optimization. The final parameters represent the D

Parametrising Large Bond Dimension Environments

In general, for larger bond dimensions, there is no shallow circuit that can capture the environment exactly. Repeating the same procedure above, diagonalising the r tensor gives a parametrisation of the right environment that generalises to a high bond dimension



(2.8)

where the $Singular\ Values$ gate encodes the singular values of the $2n$ qubit gate, V, along the diagonals of the output matrix, which maps the lower n qubits to the upper n qubits. The W gates are arbitrary 3-qubit gates, which are themselves decomposed into shallow circuits. The right-hand side of Eqn. 2.8 shows one way to approximately encode these gates by extending the $D = 2$ parametrisation to act across multiple qubits. The factorised singular value matrix, acting over $2n$ qubits, accurately captures the original matrix's most significant n singular values. As the largest singular values are the most significant, this factorisation may prove effective at extending to higher bond dimensions, although more work is necessary to find if this is indeed the case. All unitary circuits acting on an all-zeros input state can be recast as a non-unitary matrix mapping from a subset of the qubits to the remaining qubits. The insight that a low-rank decomposition of this matrix is capturable with low depth circuits has been used to construct expressive low depth variational quantum circuits[38].

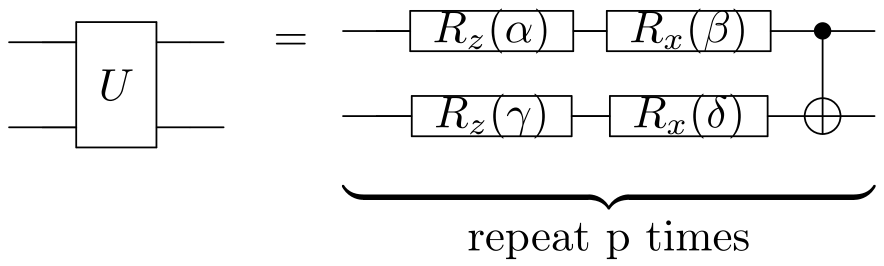
2.2 Optimizing Quantum MPS

The previous section has demonstrated that both finite and translationally invariant MPS are equivalent to quantum circuits. The goal is now to make use of these representations to be able to obtain valuable information about a system of interest. For example, quantum MPS can be used as an ansatz to study the ground states of a Hamiltonian variationally. Using finite quantum MPS as a VQE ansatz is relatively straightforward. Using a translationally invariant MPS as a VQE ansatz requires slightly more care. A simple gradient update will produce states where the V and U pair are no longer consistent with a translationally invariant state. The naive solution to this might be to re-calculate V for each new set of parameters to get an accurate estimate of the energy and gradients of the state parameters. Such nested optimisation loops are likely to be very slow. Instead, one can augment the usual VQE cost function to include a term that penalises producing inconsistent state and environment pairs. For a Hamiltonian H the cost function is given by

$$C(\theta_U,\theta_V) \;= \underbrace{(1-\gamma)\langle\psi(\theta_U,\theta_V)|H|\psi(\theta_U,\theta_V)\rangle}_{\text{Energy Term}} + \underbrace{\gamma Tr[(\rho(\theta_V)-\sigma(\theta_U,\theta_V))^2]}_{\text{Environment Consistency}} \tag{2.9}$$

where $\gamma \in [0,1]$ is a parameter to prioritise optimisation towards improving the energy or the state/environment consistency. This term allows for more sophisticated optimisation schemes. A particularly effective one is to start with $\gamma \approx 0$ to prioritise quick improvements in energy, followed by linearly increasing γ to enforce state/environment consistency later in the optimisation. Varying γ helped avoid local energy minima during optimisation. In practice, one uses a shallow factorisation of the state unitary, U, to optimise over. Fig 2.5 shows that such shallow factorisations are able to effectively capture the $D = 2$ iMPS ground states effectively.

a)



b)

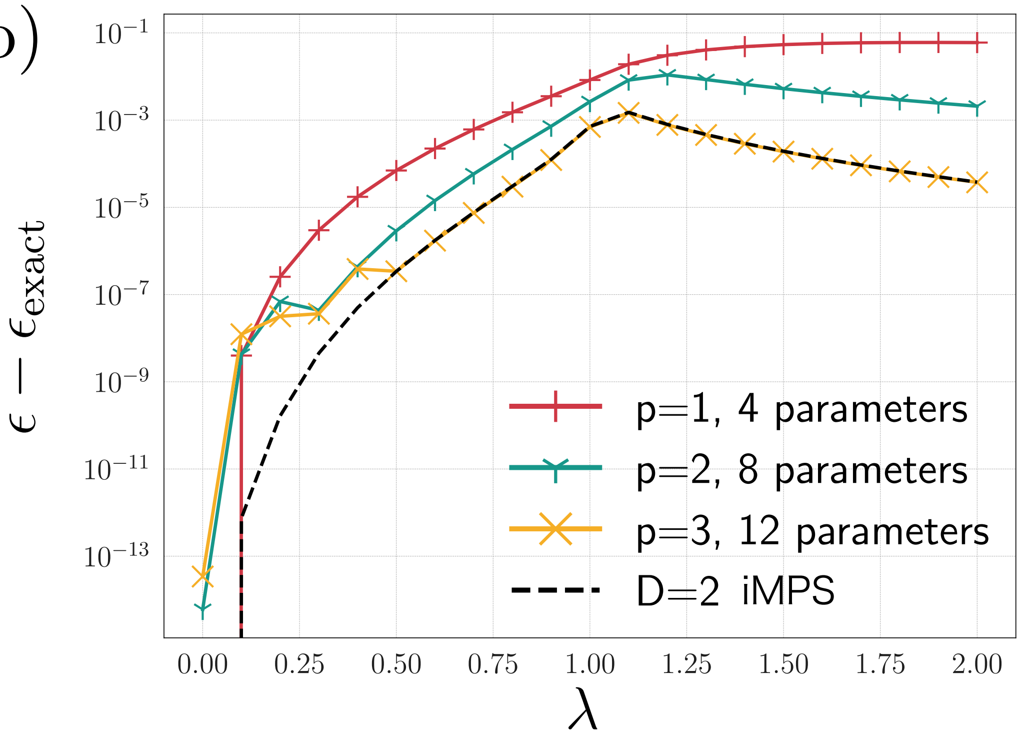


Figure 2.5: **Ground State Simulation:** a) The shallow factorization of the state unitary, U. b) The error in the ground state of the transverse field Ising model where $\lambda = \frac{J}{g}$. Shallow factorisations are able to accurately capture the exact $D = 2$ iMPS ground state.

2.3 NISQ Implementation

This section is based on work done in [4]. Optimising $D = 2$ quantum iMPS circuits has been realised on Google's Rainbow processor, which shares the Sycamore architecture used in Google quantum supremacy experiment[11]. There is an open question regarding what problems can be solved effectively by NISQ circuits. One such problem might be studying systems at criticality. Critical systems are characterised by a divergent correlation length. Using finite-sized circuits to study systems at criticality can result in finite-size scaling effects. The long-range correlations typical of critical systems make low bond dimension MPS methods prone to error and could benefit from the ability of quantum circuits to express high bond dimension tensor networks. It might be the case then that using a large bond dimension, translationally invariant quantum MPS could provide an advantage when studying systems at criticality on near-term hardware. This work demonstrates that systems at criticality can be effectively prepared and optimised on noisy quantum hardware using error mitigation.

Optimizing circuits on noisy, near-term hardware brings unique challenges to overcome. Extracting valuable insights from these machines requires error mitigation; otherwise, errors on the device will invalidate the results. There have been many proposals to mitigate noise in NISQ devices, some of which have good synergies with quantum MPS[39, 40].

2.3.1 Experiment

This work studies the transverse field Ising model (TFIM) using quantum MPS. The TFIM Hamiltonian with exchange coupling J and transverse field g is given by

$$H = \sum_i [J Z_i Z_{i+1} + \frac{g}{2}(X_i I_{i+1} + I_i X_{i+1})] \tag{2.10}$$

where Z and X are Pauli operators. This model demonstrates a quantum phase transition in the ground state at $\frac{g}{J} = 1$. A hybrid classical-quantum algorithm variationally searches for the translationally invariant ground state of the TFIM at criticality using a $D = 2$ MPS ansatz. The TFIM Hamiltonian has three terms to measure to estimate the energy. There are two-site ZZ interaction terms and two onsite terms, XI and IX. The translationally invariant ground state is the state and environment pair that simultaneously

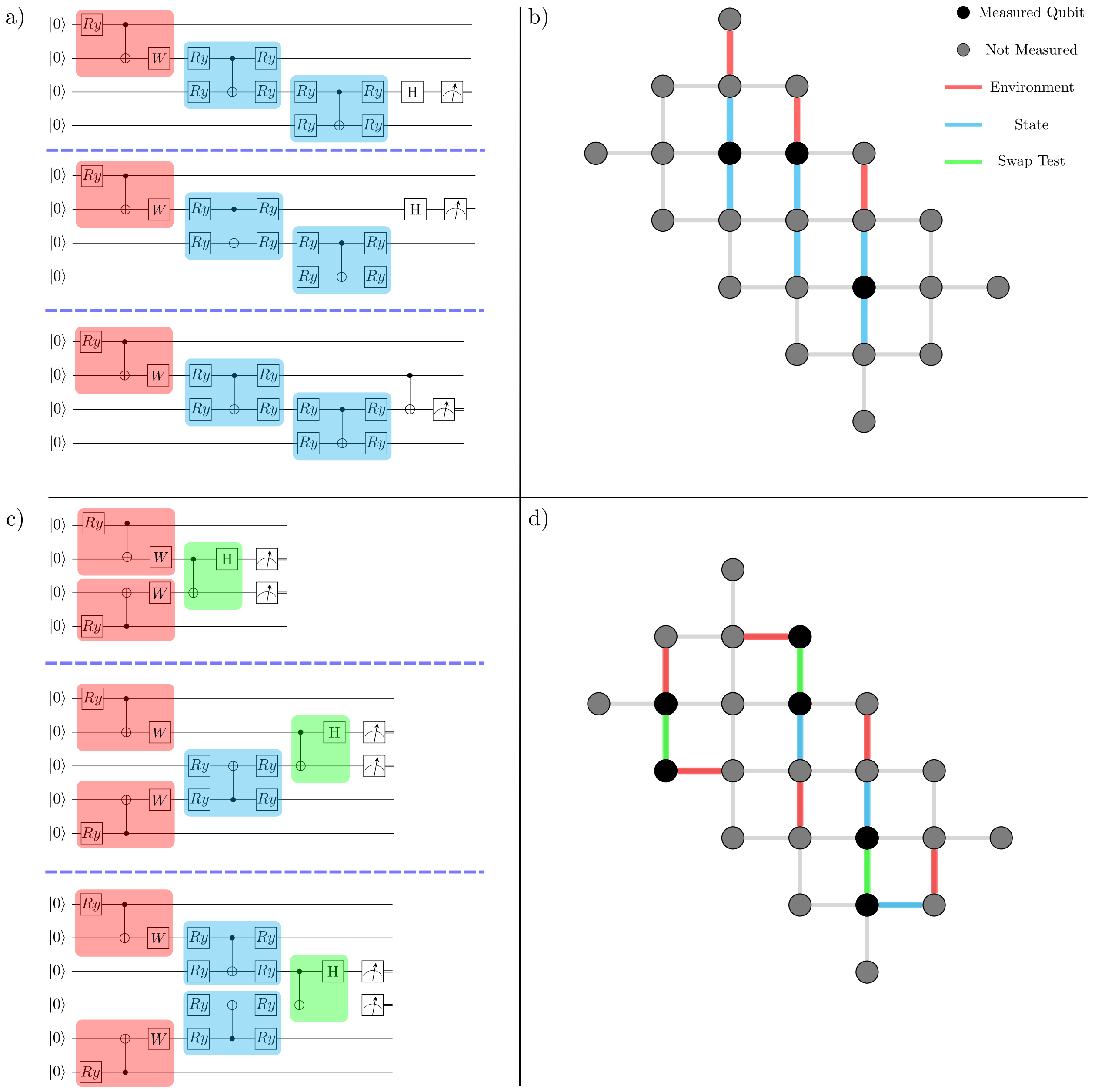


Figure 2.6: **Rainbow Optimization Circuits:** a) and c) The circuits used to calculate the 6 terms in the cost function to optimize transverse field Ising model on the Rainbow device. Three terms are needed to calcualte the energy, and three terms are needed to calculate the consitency between the state and environment tensors. b) and d) How these circuits can be laid out on the Rainbow device, showing the connectivity and layout of the qubits on the Rainbow device.

minimises the Hamiltonian's energy while keeping the state and environment consistent. In total, there are six terms in the cost function to minimise, given by the six circuits in Fig 2.6a and c. These can be laid out on the Rainbow device in parallel as given in Fig 2.6b and d.

2.3.2 Results

Fig 2.7 gives the results of optimizing the MPS inspired quantum circuits. These circuits simultaneously optimise the energy and the state environment consistency equations. A reduced factorisation of the state unitary, U, was used to minimise circuit depth and reduce the impact of noise, as shown in Fig 2.6. Even with this reduced parametrisation, the optimised circuits were very close to the true ground state of the TFIM at criticality. For $\frac{g}{J} < 1$ the major source of errors were oscillations in energy expectations measurements on the device of unknown origin. These are shown clearly in Fig 2.7c. For $g = 0.4$ the error was particularly noticeable. For $\frac{g}{J} > 1$ we find the parametrisation less effectively captures the ground state.

A quasi-adiabatic optimisation strategy was used alongside an SPSA optimiser to optimise the parameters. The circuits are initialised at a value of g where optimisation was easy. Initialisation begins with the state at the ground state of the $g = 0$ Ising Hamiltonian. Then g is incrementally increased, and at each step, the state is re-optimised with the previous ground state as the starting point. This procedure was necessary to prevent optimisation from terminating at local minima.

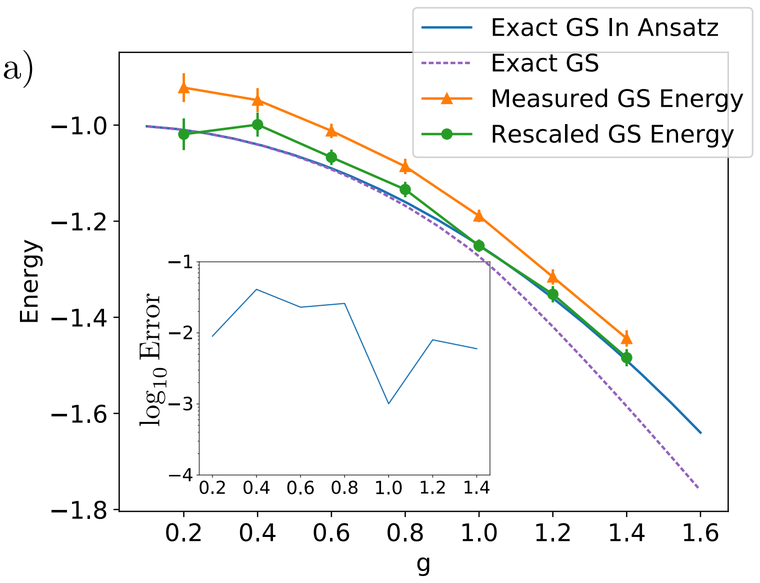
2.3.3 Error Mitigation Techniques

NISQ devices are often too noisy to directly use the results from the device. *Error Mitigation* is the name given to a broad set of techniques that improve the performance of these devices using some classical pre and post-processing, which can identify and eliminate some noise sources from the device. Three error mitigation strategies were applied concurrently in finding the ground state.

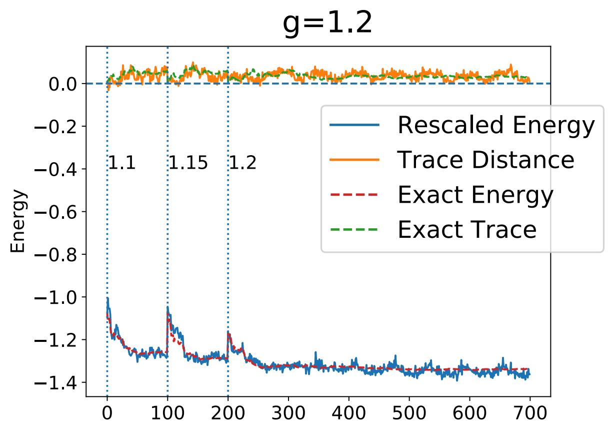
Confusion Matrix

One source of noise on the Rainbow device was classical bit-flip errors that occurred in readout. The measured output bit string, s_1 would be incor-

a)



b)



c)

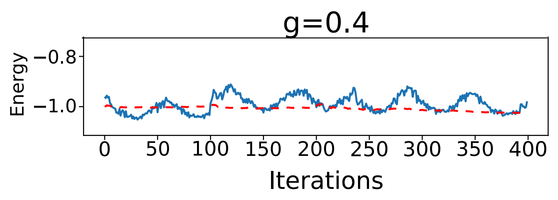


Figure 2.7: **Rainbow Ground State Optimization Results:** a) The ground state energies as measured on the Rainbow device, before and after depolarization rescaling. Measured results show good agreement with the minima attainable within the shallow ansatz class used, and good agreement at crticality with the true ground state. Erroneous ground state energies at $g = 0.4$ can be explained by oscillations in the measured energy on the device of unknown origin. b) Example quasi-adiabatic optimisation to find the ground state at $g = 1.2$. g is varied from 1 to 1.2 and at each step the ground state is found and used to initialise the next step. Throughout the optimization the state is able to retain good consitency between U and V (top curve). c) Example oscillations in the measured energy at $g = 0.4$. Note that the energy implied by the parameters exactly (the red dashed curve) varies slowly while the measured energy varies rapidly, indicating an error on the device.

rectly measured as another bit string, s_2, with a probability $P_{s_1 \to s_2}$. Readout errors are corrected by estimating the transition probabilities for all possible bit strings and using the values to correct for these transitions on average. The transition probabilities are stored in a *confusion matrix* given by $M_{s_1,s_2} = P_{s_1 \to s_2}$. The true probability of measuring a bit string, P_s^T, given the confusion matrix and the measured probability P_s^M, is found by inverting the confusion matrix:

$$P_s^T = M^{-1} P_s^M \tag{2.11}$$

The size of the confusion matrix is exponential in the length of the measured bit strings. This method scales exponentially for problems where the measured operators act on many qubits. Many interesting systems, including the Ising model, require measuring terms of fixed size and do not scale with system size. However, the three terms in the cost function needed to calculate the state-environment consistency require measurements on $\mathcal{O}(log(\chi))$ qubits. Therefore the exponential overhead caused by readout measurement error mitigation can overwhelm any computational speed-up for high bond order Quantum MPS. Hopefully, later generation devices will not need this expensive form of error mitigation. For time evolution this mitigation strategy wasn't necessary. Numerically it was found that if the goal was the maximise the probability of a particular bit string then the readout errors didn't affect the resulting state in a meaningful way.

Floquet Calibration

The Sycamore architectures have an uncertainty in the interaction implemented on the device due to a parasitic C-PHASE operation on the device[41]. Floquet calibration works by repeatedly applying a two-qubit gate on the chip to amplify errors in the angle applied on the gate. The calibration scheme corrects the errors in the applied angles on the circuit[39]. Floquet calibration has the effect of increasing the fidelity of the applied circuit with the desired circuit. The methodology, as reported in [39], only corrects angles in the single qubit native gates on the Sycamore devices. A similar form of error mitigation was proposed for two-qubit gates in [42]. This was not used in this work, as the error mitigation strategies already being used were sufficient to get good estimates of the ground state energy.

Depolarizing Noise

Depolarization is a significant source of error in the measured outputs of these quantum circuits. Depolarization errors are corrected using a known reference circuit. Consider the effect of the depolarizing channel on a single-qubit system,

$$\varepsilon(\rho) = \frac{pI}{2} + (1-p)\rho \tag{2.12}$$

on the expectation of the energy

$$\langle E \rangle = tr[H\rho]. \tag{2.13}$$

The energy of the depolarized state is given by

$$\begin{aligned}
\langle E \rangle_\varepsilon &= tr[H\varepsilon(\rho)] && (2.14)\\
&= tr[H(\frac{pI}{2} + (1-p)\rho)] && (2.15)\\
&= \frac{p}{2}tr[H] + (1-p)\langle E \rangle && (2.16)\\
&= (1-p)\langle E \rangle. && (2.17)
\end{aligned}$$

where the last equality holds if the Hamiltonian model terms are traceless. The depolarized energy is related to the true energy by a constant multiplicative factor. Approximating depolarizing noise in this way is valid for more complex multi-qubit systems[43]. The TFIM terms are traceless, meaning the simple expression in Eqn 2.17 can be used to correct depolarizing noise. To correct depolarizing errors, a reference circuit where $\langle E \rangle$ is known can be run on the device and $\langle E \rangle_\varepsilon$ measured. Then energies of similar circuits on the device can be corrected by multiplying by the factor $\langle E \rangle / \langle E \rangle_\varepsilon$. When $g = 0$ and $J = -1$, the state $|0\rangle^{\otimes n}$ is the ground state of the TFIM and gives an energy of J. This state was created by setting all variational parameters in the circuit close to zero. The energy was measured, which can then correct measured energies for $g > 0$. The improvements in the energy approximation when using rescaling is clear in Fig 2.7a.

2.4 Summary

This chapter has introduced quantum matrix product states, a set of quantum circuits which can represent classical matrix product states. This chapter extends finite quantum MPS to translationally invariant quantum MPS by introducing an environment tensor to summarise the influence of an infinite number of site tensors. Quantum MPS circuits for variationally finding the environment tensor are given, including circuits on hardware permitting mid-circuit measurements. Translationally invariant QMPS can efficiently represent the critical ground state of the TFIM, as verified on near-term hardware on Google's Rainbow device. The following section introduces algorithms to time evolve quantum MPS and demonstrates applications to studying dynamical phase transitions and quantum many-body scarring.

Chapter 3

Time Evolving QMPS

This chapter is based on the publications; Parallel Quantum Simulation of Large Systems on Small Quantum Devices[3], and Simulating ground state and dynamical quantum phase transitions on a superconducting quantum computer[4]. In these papers, we develop algorithms to time evolve quantum matrix product states. This builds upon the framework developed in the same paper for representing matrix product states on quantum devices. We develop two related algorithms for time evolving quantum matrix product states, termed the transfer matrix and the power method algorithms. Both of these algorithms are based on the time dependant variational principle. Two different algorithms are provided to calculate the overlap between two translationally invariant quantum states on a quantum device. These are then used to benchmark variational TDVP on translationally invariant quantum matrix product states. We apply one of these procedures on Google's Syamore hardware, and demonstrate that the power-method approach to calculating overlaps is potentially applicable on near term hardware. In this work I helped develop the time evolution algorithms, involved in both the theoretical developments of the transfer matrix and power method variants. I implemented classical simulation code using both methods to test their accuracy, and implemented the power-method algorithm on Sycamore hardware, alongside error-mitigation strategies.

The simulation of quantum systems is one of the most compelling reasons to build quantum computers. Physical laws govern the evolution of systems. Systems evolve following differential equations that originate from the physical laws. Given an initial state, the goal is to describe how the state evolves into the future. Numerical approaches to this problem, both classical and quantum, proceed similarly. Time and space are discretized, and the differential equation is solved iteratively for many time steps on this discretized space. However, quantum systems pose a unique difficulty; there are exponentially many differential equations to solve as the system size grows.

Time-evolution using tensor network states side steps this difficulty by restricting time evolution to low entanglement manifolds of states. Algorithms such as Time Evolving Block Decimation (TEBD)[44] and the Time Dependent Variational Principle (TDVP)[45] allow for the simulation of large quantum states without paying this exponential cost. For early-time, low-entanglement systems, these methods are exact. However, for late-time, large-entanglement systems, these methods are only approximate and introduce an error originating from the projection to lower bond dimension states.

Tensor network time evolution algorithms are a good candidate for implementation on quantum devices. Tensor network states efficiently capture low entanglement states. Low entanglement states naturally appear at short times when time evolving, starting from low-entanglement states. As entanglement grows during time evolution, the increased expressiveness of quantum hardware allows for evolution to longer times and more entangled states without error. Furthermore, NISQ devices are limited in the entanglement that they can generate. Errors in the circuit push the generated states to classical mixtures of non-entangled states. Therefore time evolution algorithms which restrict to low-entanglement states could be valuable for studying quantum dynamics on NISQ devices.

The following chapter introduces *Quantum TDVP*, an algorithm that can time evolve translationally invariant quantum MPS, and demonstrates its applicability on near-term quantum devices.

Variational Time Evolution

Given a parametrised state at time t, $|\psi(A(t))\rangle$, the state at time $t + dt$ is given by

$$|\phi(t + dt)\rangle = e^{-iHt}|\psi(A(t))\rangle \tag{3.1}$$

where newly evolved state $|\phi(t + dt)\rangle$ is not necessarily expressible within the ansatz class $|\psi(A)\rangle$. For example, time evolution under a multi-site Hamiltonian operator will generate entanglement between pairs of interacting sites, necessarily increasing the bond dimension of the state. To remain in

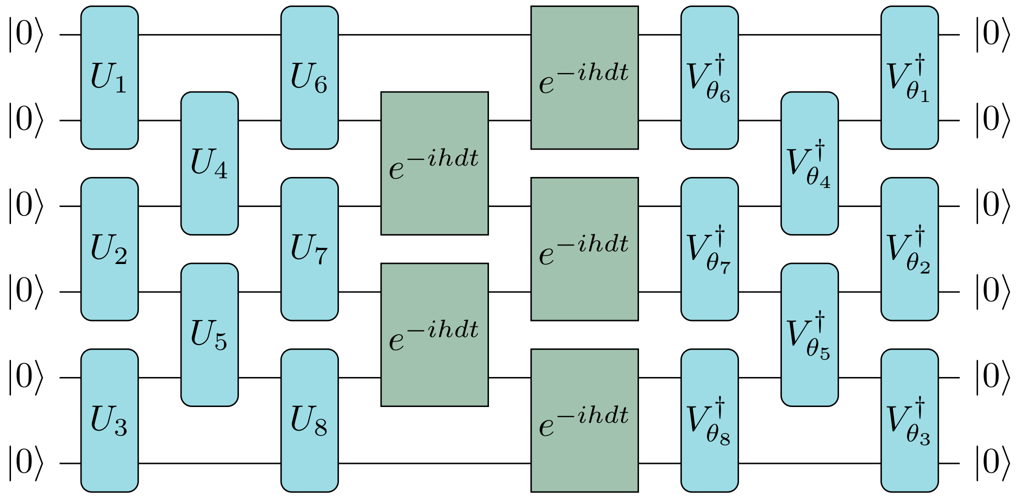


Figure 3.1: **Finite Time Evolution -** A finite state is evolved under a local Trotterized Hamiltonian. The state $|V(\vec{\theta})\rangle$ is varied to maximise the overlap with the original state.

the manifold of fixed bond dimension the time evolved state can be projected back onto the variational manifold according to

$$A(t+dt) = \underset{B}{\operatorname{argmax}} \, |\langle\psi(B)|e^{-iHdt}|\psi(A)\rangle| \tag{3.2}$$

with B a set of variational parameters to search over and $A(t+dt)$ the parameters of the next time step.

For finite quantum MPS circuits, this procedure is straightforward to implement as a quantum circuit, Fig 3.1. Variational time evolution done in this way is efficient due to good initialisations of the parameters[46]. The search for the next parameters begins with the parameters at time t. For small dt, these parameters are a good guess for the next step; over a short time, the parameters can only vary a small amount. This scheme allows for approximate time evolution beyond the coherence time of the physical qubits. The variational updates repeat arbitrarily many times to simulate time evolution over long times.

3.0.1 Pseudocode for Finite MPS Time Evolution

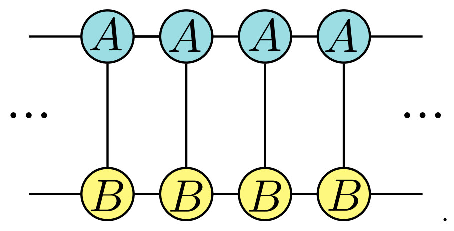
```python
import cirq
from scipy.optimize import minimize

# original parameters:
time_t_params = ...

# guess for the new params is the same as the starting
 point
time_t_plus_delta_params = copy(time_t_params)

def cost_func(new_params, original_params, simulator):
  circuit = cirq.Circuit()
  circuit = add_ansatz_circuit(circuit, original_params)
  circuit = add_hamiltonian_circuit(circuit)
  circuit = add_inverted_ansatz_circuit(circuit, new_params
 )
  result = simulator.simulate(circuit)
  return np.abs(result[0])

# perform a single time step to find the new params
time_t_plus_delta_params = minimize(cost_func, x0=
 time_t_plus_delta_params)
```

Listing 3.1: Example python pseudocode for optimizing a finite quantum circuit using TDVP.

3.1 Translationally Invariant Variational Time-Evolution

The extension of variational time evolution to translationally invariant states is slightly more involved. The basic subroutine required to perform variational time evolution is measuring the overlap between two states. The overlap between two states is used as the cost function to maximize to perform a single update step. For infinitely large states, any two states that are not identical will have an overlap of 0. Instead, the relevant quantity is the *overlap density* of two states, which is the rate at which the overlap between two states decreases as a function of the length of the spin chain considered. In the tensor network picture this quantity is the largest right eigenvalue of the *mixed transfer matrix*. Given a pair translationally invariant states with associated matrix product tensors A and B, the overlap is graphically given by the tensor network diagram

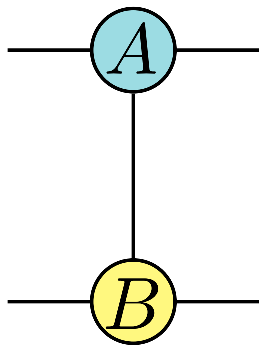
$$\langle \psi_A | \psi_B \rangle =$$



(3.3)

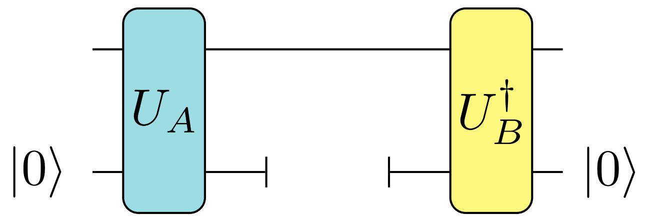
where it is assumed that the chain extends to infinity in both the left and right directions. If A and B do not represent the same state, this overlap will be zero. The mixed transfer matrix is identified as

$$E_B^A =$$



(3.4)

which can be understood as a matrix by grouping the left and right legs together. This transfer matrix can be represented using unitary matrices,



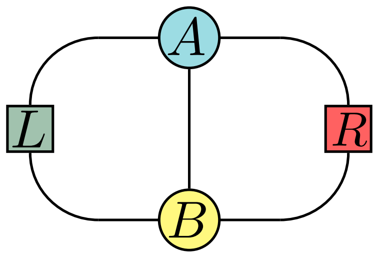
(3.5)

where the bottom open legs are the left legs and the top two legs are the right most legs in Eqn. 3.4.

The overlap is equivalent to the largest right (or left) eigenvalue of this transfer matrix. The mixed transfer matrix is not available for quantum circuit implementations of time evolution using translationally invariant states.

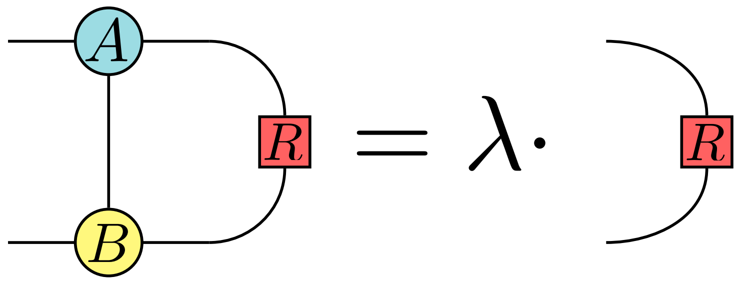
Instead the overlap must be calculated by the contraction of the mixed transfer matrix with the left and right environments,

$$\langle\psi_A|\psi_B\rangle =$$



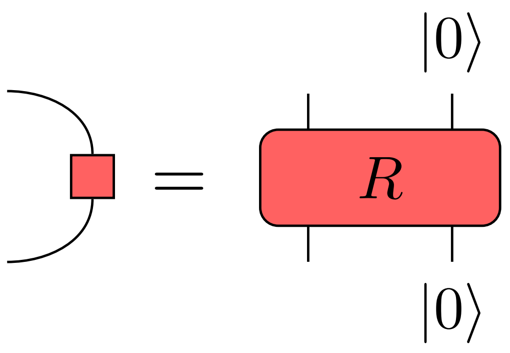
(3.6)

where the right environment satisfies the eigenvalue equation



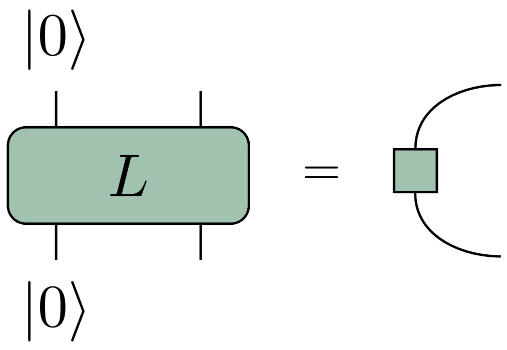
(3.7)

with λ the largest right eigenvalue. A similar equation holds for the left eigenvalue and eigenvector. The right and left-most tensors in Eqn 3.6 are also known as the environment tensors, and perform a similar function as the environment tensors introduced in Chapter 2. These tensors summarise the effect of the infinite chain of tensors to the left and right of the site. The left environment is not the identity in this scenario because the MPS tensors are different, so that the left canonical condition does not hold. Eqn 3.6 is equivalent to a quantum circuit,Fig 3.2, so is suitable as a subroutine for quantum matrix product state time evolution. The right environment is embedded into a unitary matrix as



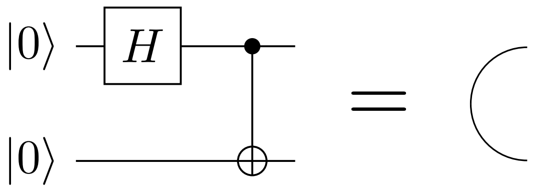
(3.8)

and the equivalent for the left environment is



(3.9)

Eqn 3.6 can be seen to be equivalent to the circuit in Fig 3.2, using the identity



(3.10)

to connect the state tensors to the right environment. The circuits presented here give a way to estimate the overlap of two translationally invariant states, given access to the right and left environments of the transfer matrix embedded in unitary matrices.

Time Evolved Transfer Matrix

To perform variational time evolution with translationally invariant states, it is necessary to identify the mixed transfer matrix needed to estimate the overlap density. The transfer matrix is equivalent to a repeated block of unitary matrices in the infinitely wide and deep circuit needed to calculate the overlap. Fig 3.3 shows the mixed transfer matrix for a second order Trotter decomposition of a time evolution operator. Finding the largest right eigenvalue of this matrix is equivalent to finding the overlap between a time evolved state and a candidate bond dimension two state. Numerically it appears that being limited to translationally invariant states allows a simplification of the time evolution decomposition. Using only the even terms in

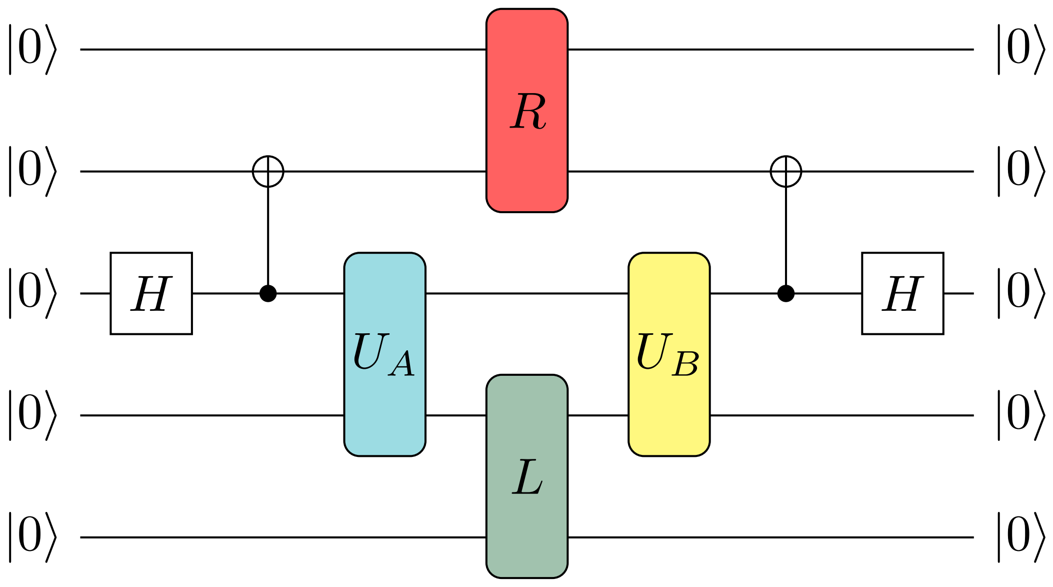


Figure 3.2: **Circuit to calculate overlaps -** Circuits to measure the overlap between two infinite MPS on a quantum circuit, given the right and left environment vectors embedded in unitary matrices. $P(|000...\rangle) = |\langle\psi(U_B|\psi(U_A))\rangle|^2$

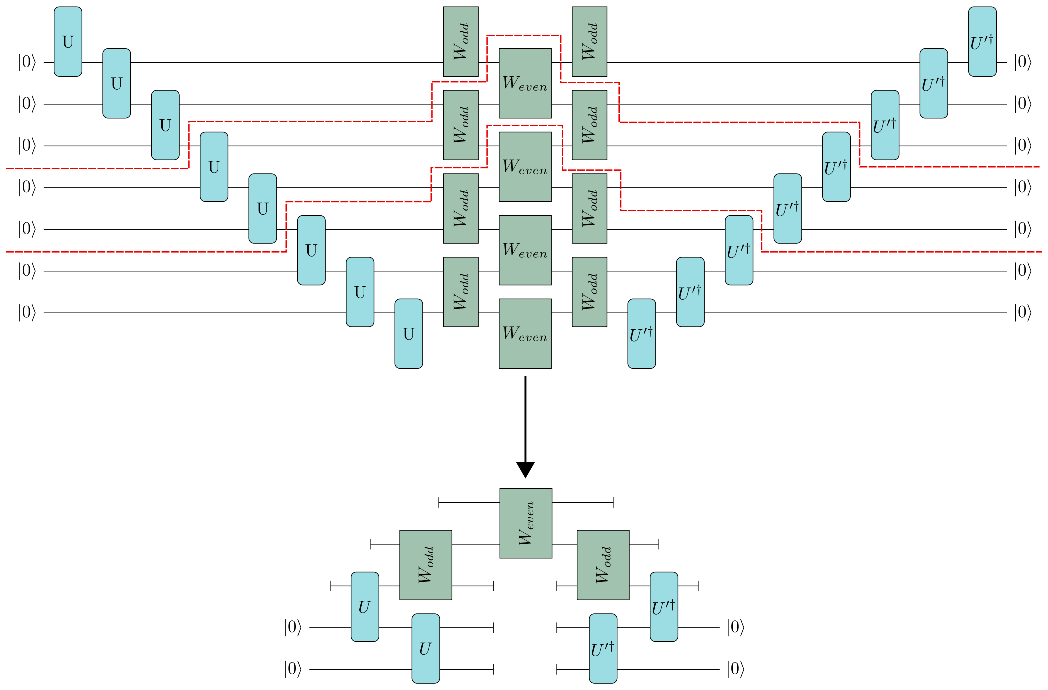


Figure 3.3: **Time Evolution Transfer Matrix -** Transfer matrix of a translationally invariant state with a second-order Trotterized time evolution operator. The whole state (top) is infinitely wide and deep but only the transfer matrix (bottom) needs to be considered to calculate the overlap density. The section outlined in red corresponds to the transfer matrix.

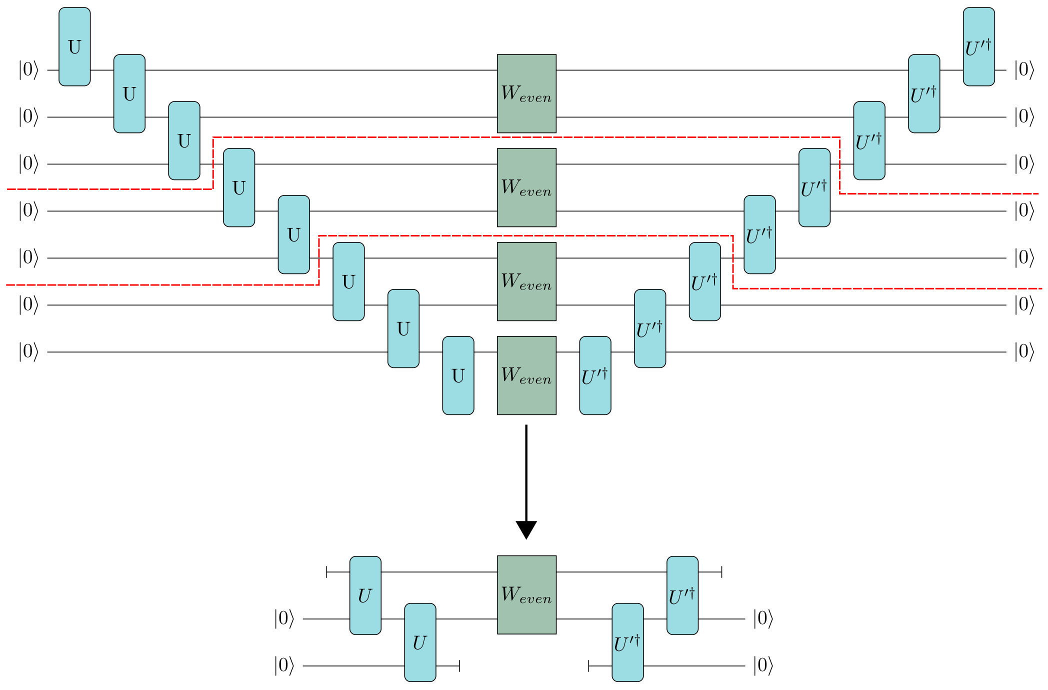


Figure 3.4: **Half Time Step Time Evolution -** The transfer matrix can be simplified by projecting back to a translationally invariant state after just the even or odd terms in the Trotter decomposition. This scheme is effective for translationally invariant MPS, as evolving with just the odd (or even) terms is equivalent to to evolving with both the even and odd terms, but for half the time step.

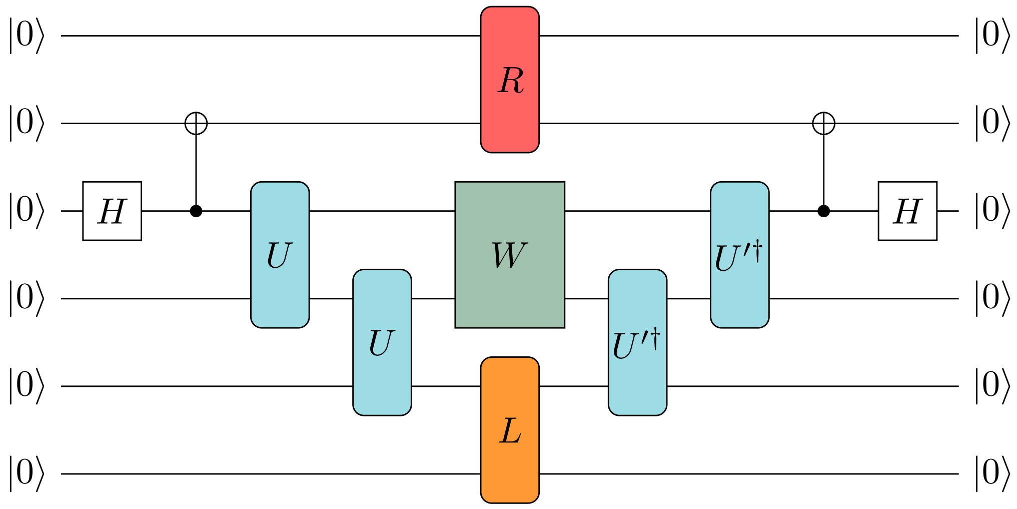


Figure 3.5: **Overlap Circuit For Time Evolution -** The circuit needed to calculate the overlap between a time evolved state and a candidate bond dimension state. R and L are the right and left eigenvectors associated to the largest eigenvalues of the transfer matrix with the time evolution operator in the transfer matrix.

the time evolution decomposition does not accumulate errors as quickly as expected. Evolving using just the even terms is equivalent to the full evolution step of a translationally invariant Hamiltonian, but with half the time step. This *half-step* decomposition is shown in Fig 3.4.

Taking the transfer matrix identified in Fig **??** and connecting the correct left and right environments, we get the circuit needed to calculate the overlap between a time evolved state and a candidate state of fixed bond dimension. The circuit in Fig 3.5 calculates the overlap between a time evolved state and a candidate bond dimension two quantum MPS. The circuit relies on the correct construction of the right and left unitary environments. The following section introduces algorithms to find the left and right environment gates.

Finding Environment Tensors

Using the transfer matrix method introduced above, it's possible to construct a quantum circuit to estimate the overlap between a candidate MPS state and a time-evolved MPS state. This circuit is given in Fig 3.5. This circuit requires that the environment tensors, R and L, are the largest weight right and left eigenvectors of the time evolution transfer matrix. How can such a pair be found? It was shown in Chapter 2 how to find the largest weight eigenvector of Hermitian transfer matrices by solving a fixed point equation. R and L are still the solutions to fixed point equations, Fig 3.6, but without the luxury of the transfer matrix being Hermitian. In this case, the transfer matrix given in Fig 3.4 is not Hermitian, as the two unitary matrices representing the MPS are not necessarily the same. Therefore the Rayleigh-Ritz variational principle for finding the maximum or minimum weight eigenvectors of matrices no longer applies.

Instead finding the largest weight eigenvectors of the transfer matrix can be reframed as a min-max problem over eigenvector and eigenvalue pairs

$$\eta, r = \max_{\eta}\{\underset{(\eta,r)}{\operatorname{argmin}}\,||E^{U}_{U'}r - \eta r||^2\} \tag{3.11}$$

with the transfer matrix given by $E^{U}_{U'}$, the candidate right eigenvector, r, and the candidate largest eigenvalue, η. For time evolution, the overlap between states is often very close to 1, as short time steps ensure the states cannot differ significantly from one another. Hence this equation can be simplified to

$$\eta, r = \underset{\eta,r}{\operatorname{argmin}}\,||E^{U}_{U'}r - \eta r||^2 = \min_{\eta,r}\nu(\eta, r) \tag{3.12}$$

with the approporitate initial condition that η starts at 1, and optimization is restarted if η varies too far ($\approx \mathcal{O}(dt)$) from 1.

Solving Eqn. 3.12 on a Quantum Computer

Expanding Eqn. 3.12 gives the following expression:

$$\nu(\eta, r) = r^\dagger {E^{U}_{U'}}^\dagger E^{U}_{U'} r + |\eta|^2 r^\dagger r - r^\dagger E^{U}_{U'} r - r^\dagger {E^{U}_{U'}}^\dagger r \tag{3.13}$$

The first two terms are equivalent to quantum circuits, Fig 3.7d and f. The last two terms are not necessarily real and therefore aren't equivalent to the probability of measuring a given bit string which is necessarily real. At the minima of the function, the last two terms will be equal to the largest eigenvalue of the mixed transfer matrix. The largest eigenvalue of the transfer matrix will be *almost* real since U and U' will only differ by a small amount, and when they are equal, the largest eigenvalue is 1. Therefore the minima

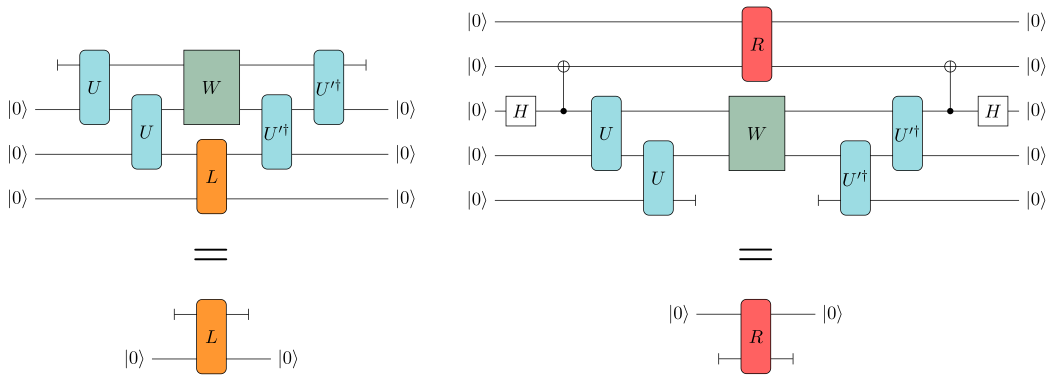


Figure 3.6: **Fixed Point Equations -** The fixed point equations satisfied by the environment tensors R and L. These are analogous to the fixed point equations introduced in section 2.

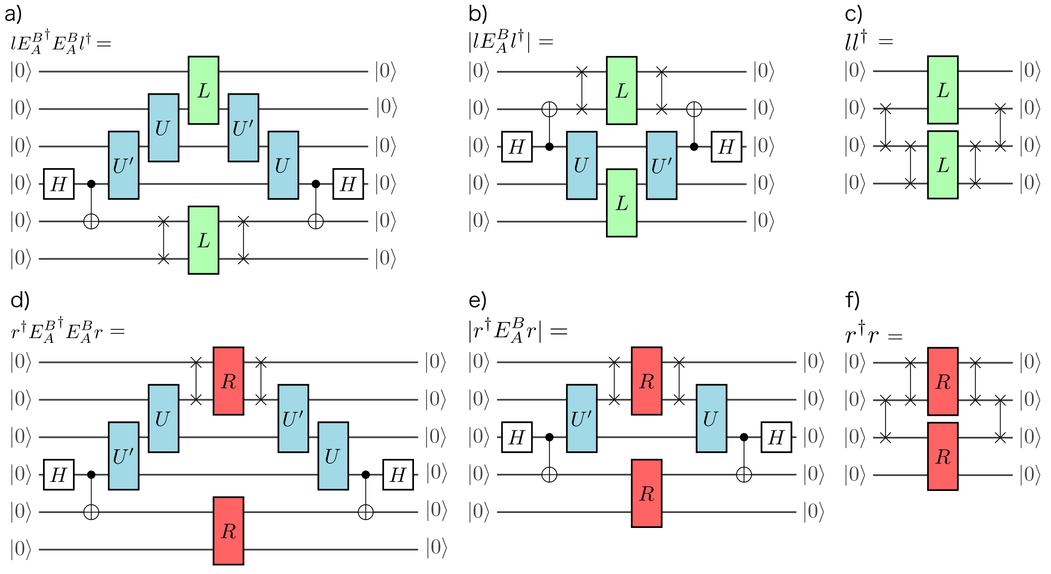


Figure 3.7: **Fixed Point Equation Terms -** Presented are the 6 circuits needed to evaluate the cost function ν' defined in Eqn. 3.14 for both the lest and the right environment. Minimising the cost function will typically find the largest eigenvalue and eigenvector pair, (η, r) of the given transfer matrix. Shown are the ciruits without the time evolution operator. The equivalent circuits with the time evolution operator have twice as many state tensors and are correspondingly deeper and wider.

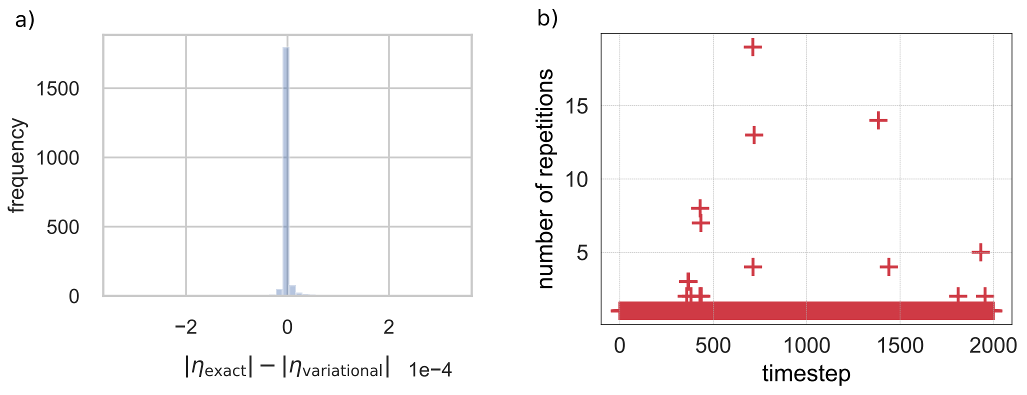


Figure 3.8: **Effectiveness of Approximate Algorithm -** The results of optimizing the approximate cost function $\nu'(\eta, r)$. a) A comparison of the exact and variationally calculated largest eigenvalue of the mixed transfer matrix along 2000 steps of a TDVP trajectory. Using a time step of $\delta t = 0.1$ and the BFGS optimizer. b) The number of times the algorithm had to restart because of a large deviation of η from 1 at each time step.

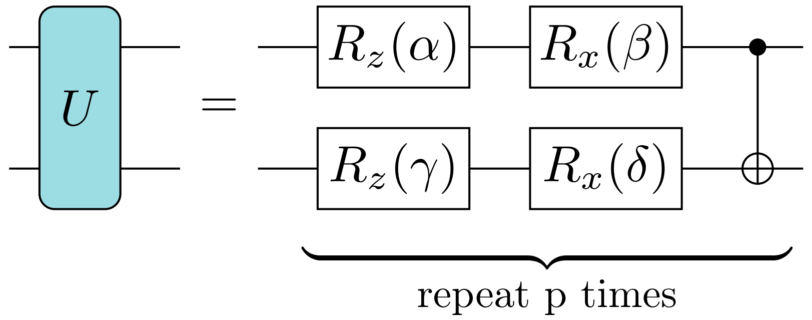
$\nu(\eta, r)$ can be approximated with the minima of the function $\nu'(\eta, r)$, given by

$$\nu'(\eta, r) = r^\dagger {E^U_{U'}}^\dagger E^U_{U'} r + |\eta|^2 r^\dagger r - 2|r^\dagger E^U_{U'} r| \tag{3.14}$$

where the difference between the two functions is $\mathcal{O}(dt)$. Thus $\nu'(\eta, r)$ is minimized instead of $\nu(\eta, r)$, with the condition that η is initialized to 1, and the optimization is restarted if η changes by more that $\mathcal{O}(dt)$. The three terms in $\nu'(\eta, r)$ are calculated using the three circuits given in Fig 3.7a-c for the left environments and d-f for the right environment. In practice this scheme works very well. Fig 3.8 highlights the effectiveness of this scheme, and demonstrates that repeating the minimisation in Eqn. 3.12 is only needed infrequently.

3.1.1 Time Evolution Simulations

With a subroutine to estimate the overlap of translationally invariant MPS on a quantum device, it is now possible to define a variational time evolution algorithm based on the time dependant variational principle. The procedure involves a nested optimisation loop. The inner loop finds an appropriate



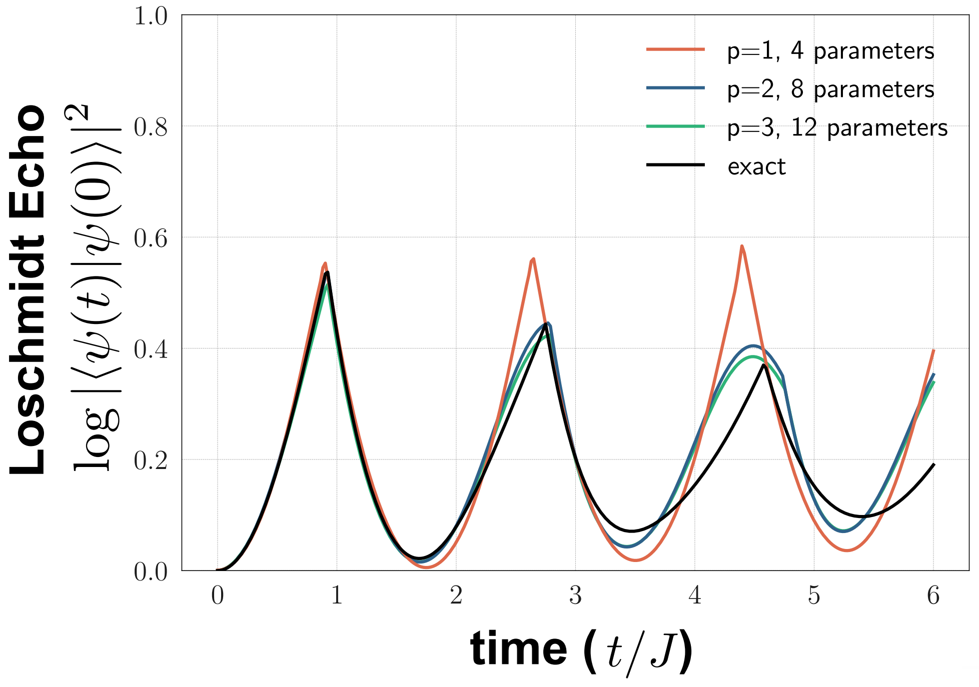


Figure 3.9: **Dynamical Phase Transitions -** There are dynamical phase transitions displayed in the Loschmidt echo evolving a state under the Transverse Field Ising Model. Simulated results show that shallow factorisations of the state unitaries are able to effectively capture the dynamics. Greater depth results in greater agreement with the analytically correct results.

right and left environment for a given U and U' pair needed to evaluate the overlap between the two MPS. Finally, the outer loop updates U' to find the highest overlap to the time evolved state. The doubled optimisation loop is repeated each step up to the desired time.

Dynamical Phase Transitions

This quantum TDVP algorithm effectively captures dynamical phase transitions in Loschmidt echos[47]., even when restricted to shallow factorisations of the state unitaries. The ground state of the Transverse Field Ising Model with $g = 1$ is prepared and then evolved under the same Hamiltonian with $g = 0.2$. At each step, Fig 3.9 plots the overlap between the original and the latest state.

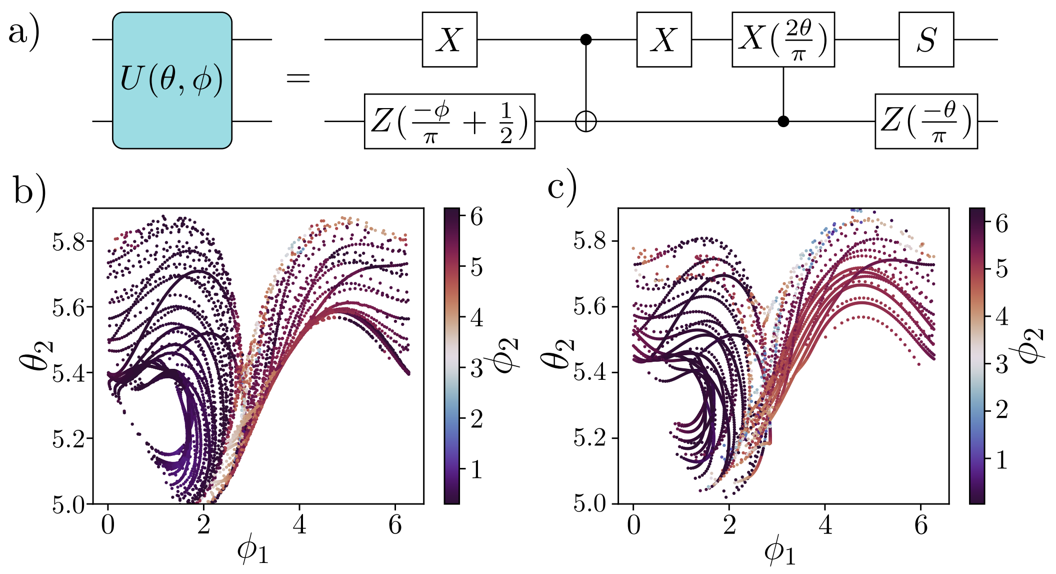


Figure 3.10: **Many Body Scars -** The Hamiltonian $H = \sum_n (1 - \hat{\sigma}^z_{n-1})\hat{\sigma}^x_n(1-\hat{\sigma}^z_{n+1})$ displays a property where certain states display persistent oscillations as they evolve. This is known as many-body scarring. These scarred states can be described using low bond order MPS[48]. a) A two site translationally invariant MPS state with two parameters per site is used. b) A partial Poincare section the plane $\theta_1 = 0.9$ is produced using analytic results and classical simulation of the quantum TDVP code. Initial conditions are chosen on the energy surface such that $\langle H \rangle = 1$. Initial states were chosen along a line with spacing $\delta\phi_1 = 3 \times 10^{-2}$, with $\theta_1 = 0.9$ and $\theta_2 = 5.41$. c) The same Poincare plot produced by the quantum time dependant variational principle. The figure is distorted by integration and optimization errors, but that large structures still remain.

Pseudocode for Many Body Scar States

4

Many-Body Scars

A low depth state ansatz can also represent the states used in [48] to produce Poincare maps, used to study slow quantum thermalization in the PXP model. To produce a Poincare map the parameters of the quantum state are tracked as they evolve through time. Whenever a chosen parameter crosses a plane in the positive direction, the values of the remaining parameters are recorded. This map is plotted in Fig 3.10b and c. Fig 3.10b is generated exactly using TDVP equations, and Fig 3.10c is generated using the time evolution scheme introduced here. Variational time evolution uses discrete time steps, so calculating the exact crossing point through a plane is not immediately possible. Instead, the crossing points are evaluated using an interpolation algorithm. Around the approximate crossing time, the evolution of the parameters is estimated using polynomial function interpolation. A root-finding algorithm is used on the interpolated function of the chosen parameter to find an estimate of the exact crossing time. Then this time is input into the other interpolated functions to get estimates of all other parameters at the crossing times.

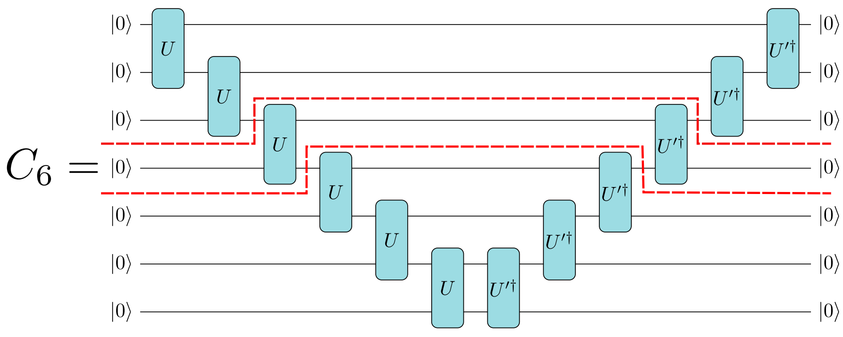
3.1.2 Power Method Approximation

The time evolution algorithm introduced above relied on a variational subroutine to find the right and left environments of the transfer matrix to calculate the overlap between a pair of states. This time evolution algorithm requires a nested optimisation loop, with the inner loop finding eigenvalue/eigenvector pairs of the transfer matrix and the outer loop optimising the parameters of the updated state to maximise the overlap. A nested optimisation loop makes this procedure slow. To accelerate time evolution it is possible to replace the inner loop with a non-variational procedure to estimate the overlap of two states using the power method.

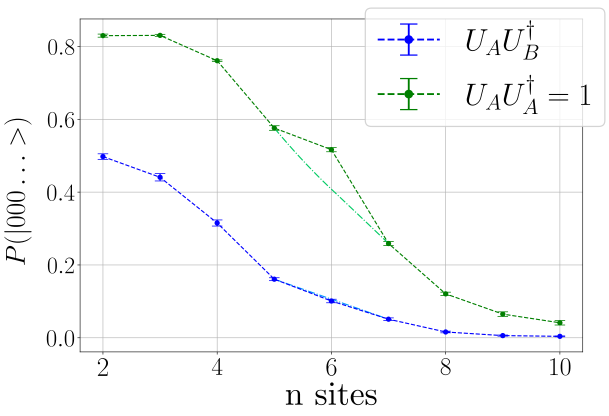
The power method is a technique to approximate the largest eigenvalue of a matrix. The largest eigenvalue of the transfer matrix $E^U_{U'}$ is given by

$$\lambda = \lim_{n\to\infty} \frac{\tilde{L}{E^U_{U'}}^n\tilde{R}}{\tilde{L}{E^U_{U'}}^{n-1}\tilde{R}} = \lim_{n\to\infty}\frac{C_n(U,U')}{C_{n-1}(U,U')} \tag{3.15}$$

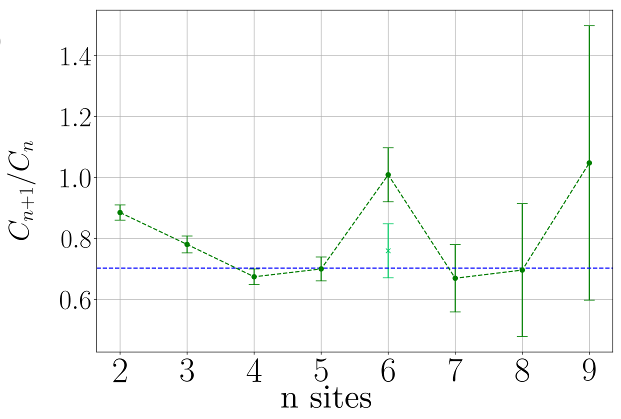
a)



b)



c)



d)

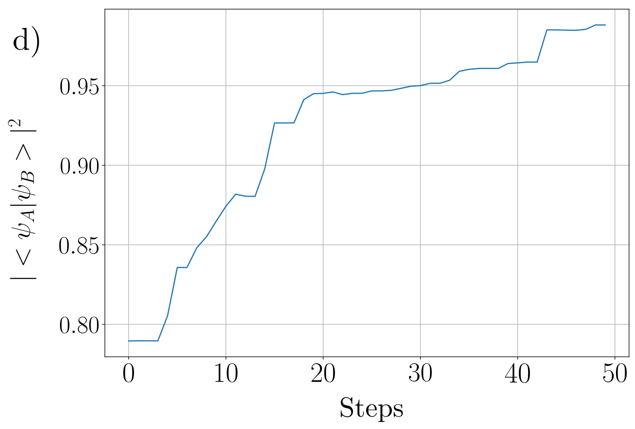


Figure 3.11: **Calculating Overlaps -** a) The overlap of two translationally invariant MPS is measured by estimating the ratio C_n/C_{n-1}. Shown is C_6, with the red dashes highlighting the transfer matrix. b) Measurement of C_n for two states for n up to 10 sites. Depolarization errors are corrected using a Loschmidt echo, shown in green, where the overlap is measured for two identical states (which ought to always be 1). At 6 sites an unknown error in the Loschmidt echo occurs. Interpolation is used to infer the correct value. c) The overlaps implied by the data in b). The blue horizontal line shows the true overlap density of the two states. At $n = 4$ the overlap has converged to a good estimate of the overlap, larger n circuits are impacted heavily by noise. d) Demonstration on Google's Sycamore device that variational optimization of one state can achieve high fidelity with the original state.

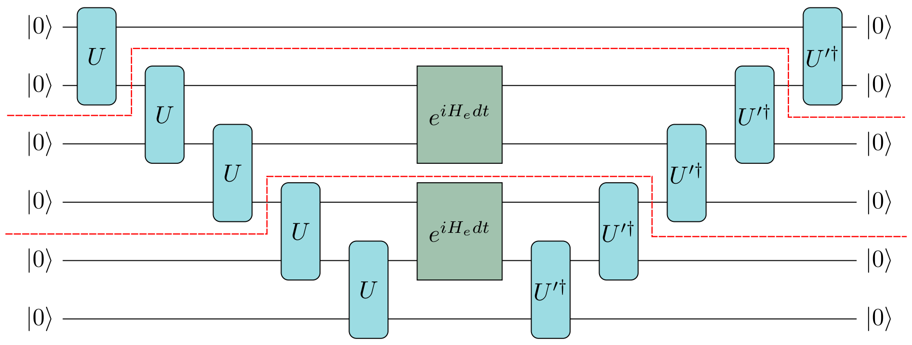
with $\tilde{R}$ and $\tilde{L}$ as approximations to the eigenvectors of the transfer matrix, and n is the power the transfer matrix is raised to. This error in this approximation decreases exponentially as n increases. The choice of $\tilde{R}$ and $\tilde{L}$ is also significant. If these are the exact eigenvectors of the transfer matrix, the power method converges at $n = 1$. The circuit to calculate C_6 is shown in Fig 3.11a. On NISQ devices, there is a trade-off between the accuracy of the approximation and the impact of noise. Large n approximations require deeper circuits, which are in turn more exposed to noise. Fig 3.11b-d investigates this trade-off to identify the ideal circuit depth on the Google Rainbow device. To correct for depolarisation errors, the value for $C_n(U, U')$ is divided by the measured value of $C_n(U, U)$ which always has the value 1 in the absence of noise. These results suggest that choosing n to be 4 or 5 provides the optimal trade-off between approximation errors and noise mitigation on the Rainbow device. The chosen value of n only works when there is no time evolution operator in the transfer matrix. Adding a time evolution operator will increase the depth of the circuit, thereby likely decreasing the optimal n to use before errors become intolerable. For time evolution choosing n to be 2 or 3 provided the best trade-off.

Time Evolution Using the Power Method

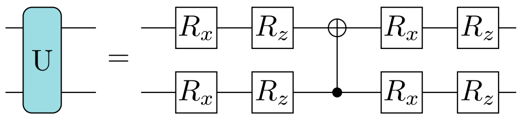
Using the power method to avoid the inner optimisation loop produces a faster time evolution algorithm. Fig 3.12a gives a quantum circuit which evaluates C_2. The transfer matrix is the matrix shown in the red dashes. The probability of measuring $|0\rangle^{\otimes N}$ gives an approximation to the square of the largest eigenvalue of the transfer matrix, λ^2. The environments are approximated as $\tilde{L} = \mathcal{I}$ and $\tilde{R}$ is approximated with an additional U and U' pair. λ is the square root of the output probability. Taking the square root of the output avoids dividing by C_1, which can be a very small number and introduces numerical instability.

A cost function constructed from this estimate of the overlap faithfully tracks the actual overlap between states. A noiseless simulation demonstrates that dynamical quantum phase transitions are present during time evolution using this estimate of the overlap, shown in Fig 3.13a. On the Rainbow device, this cost function tracks the actual value of the overlap during time evolution. For each time step in the true time evolution, the cost function is evaluated along a linear interpolation from the initial parameters to the optimal parameters. At each time step, the optimal point has a higher overlap

a)



b)



c)

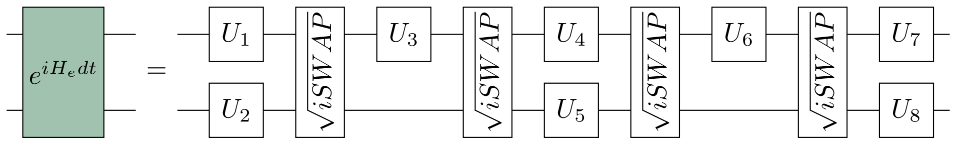


Figure 3.12: **Time Evolution Circuits -** a) The probability of measuring $|0\rangle^{\otimes N}$ as the output gives an approximation to λ^2. First $|0\rangle$ is post-selected on the top two qubits. λ is the largest eigenvalue of the transfer matrix given by the dashed red lines. b) The shallow factorization of the state unitary. c) Factorization of the time evolution operator into the native Rainbow gate set. The time evolution operator is one of the most costly elements of the circuits.

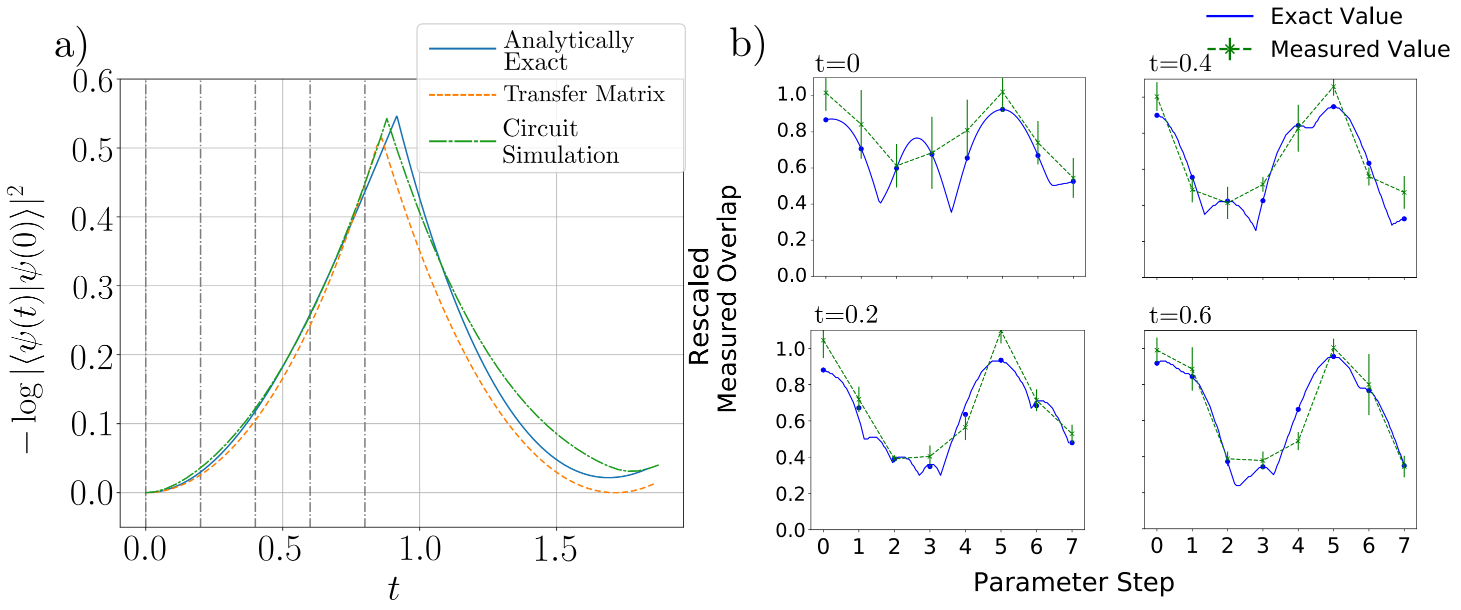


Figure 3.13: **Time Evolution Results I -** a) The dynamics of the transverse field Ising model are given analytically and compared with the outputs of ideal classical and simulated quantum circuits. In blue is given the exact evolution of the TFIM. In orange is given time evolution as obtained by optimization of the state exactly within the chosen ansatz class. In green is shown time evolution as obtained using the power method to approximate the largest eigenvalue of the transfer matrix. This is equivalent to a noise-less simulation of the circuits shown above. Both numerical methods show good agreement with the true results and display dynamical phase transitions. b) The cost function maximized in the green curve is executed on the Rainbow device. The cost function is evaluated along a linear interpolation in the 8 parameters from the initial U to the optimal U' found numerically, and extended beyond. Interpolations are given for each time step in a). The optimum value of the rescaled circuits along the interpolation agrees with the point calculated classically without errors.

with the time-evolved initial state than the initial parameters, Fig 3.13. In principle, this means a suitably chosen optimiser could perform this optimisation and recreate the dynamics shown in Fig 3.13a.

Pseudocode for Power-Method Time Evolution

```
import cirq
import numpy np

NUM_QUBITS = 6

qubits = cirq.LineQubit.range(NUM_QUBITS)

def build_2qubit_U(circuit, qubits, params, invert=False):
  # add a shallow factorized 2-qubit gate
  new_circuit = cirq.Circuit([
    cirq.X.on(qubits[0])**params[0],
    cirq.Z.on(qubits[0])**params[1],
    cirq.X.on(qubits[1])**params[2],
    cirq.Z.on(qubits[1])**params[3],
    cirq.CNOT(qubits[1], qubits[0]),
    cirq.X.on(qubits[0])**params[4],
    cirq.Z.on(qubits[0])**params[5],
    cirq.X.on(qubits[1])**params[6],
    cirq.Z.on(qubits[1])**params[7],
  ])

  if invert:
    new_circuit = new_circuit ** -1

  circuit.append(new_circuit)
  return circuit

def build_1qubit_U(circuit, qubit, params):
  # add a completely general single qubit gate
  circuit.append([
    cirq.X.on(qubit) ** params[0],
    cirq.Z.on(qubit) ** params[1],
    cirq.X.on(qubit) ** params[2],
  ])

  return circuit

def build_H(circuit, qubits, params):
  add the time evolution operator
  circuit = build_1qubit_U(circuit, qubits[0], params[:3])
  circuit = build_1qubit_U(circuit, qubits[1], params[3:6])
  circuit = circuit.append([cirq.ISWAP.on(qubits)])
  circuit = build_1qubit_U(circuit, qubits[0], params[6:9])

  circuit = circuit.append([cirq.ISWAP.on(qubits)])
  circuit = build_1qubit_U(circuit, qubits[0], params
 [9:12])
  circuit = build_1qubit_U(circuit, qubits[1], params
 [12:15])
  circuit = circuit.append([cirq.ISWAP.on(qubits)])
  circuit = build_1qubit_U(circuit, qubits[0], params
 [15:18])
  circuit = circuit.append([cirq.ISWAP.on(qubits)])
  circuit = build_1qubit_U(circuit, qubits[0], params
 [18:21])
  circuit = build_1qubit_U(circuit, qubits[1], params
 [21:24])

  return circuit

initial_params = np.random.rand(8)
H_params = np.random.rand(24)

def cost_func(time_evo_params):
  circuit = cirq.Circuit()

  for i in range(NUM_QUBITS):
    circuit = build_2qubit_U(circuit, qubits[i:i+2],
 initial_params)

  for i in range(1, NUM_QUBITS-2, 2 ):
    circuit = build_H(circuit, qubits[i:i+2], H_params)

  for i in reversed(range(NUM_QUBITS)):
    circuit = build_2qubit_U(circuit, qubits[i-2:i],
 time_evo_params, invert=True)

  simulator = cirq.Simulator()
  result = simulator.simulate(circuit)
  score = np.abs(result[0])

  return score

```

```
  new_params = minimize(cost_func, x0 = initial_params)
```

Listing 3.2: Example python pseudocode for time evolving an MPS by a single time step using the power method appoximation.

Alternative Cost Functions

There are several other options to configure the power method to estimate the overlap. The one presented above was found to perform the best on the Rainbow device. The 4 alternative cost functions which perform worse are given below, in Fig 3.14-3.17.

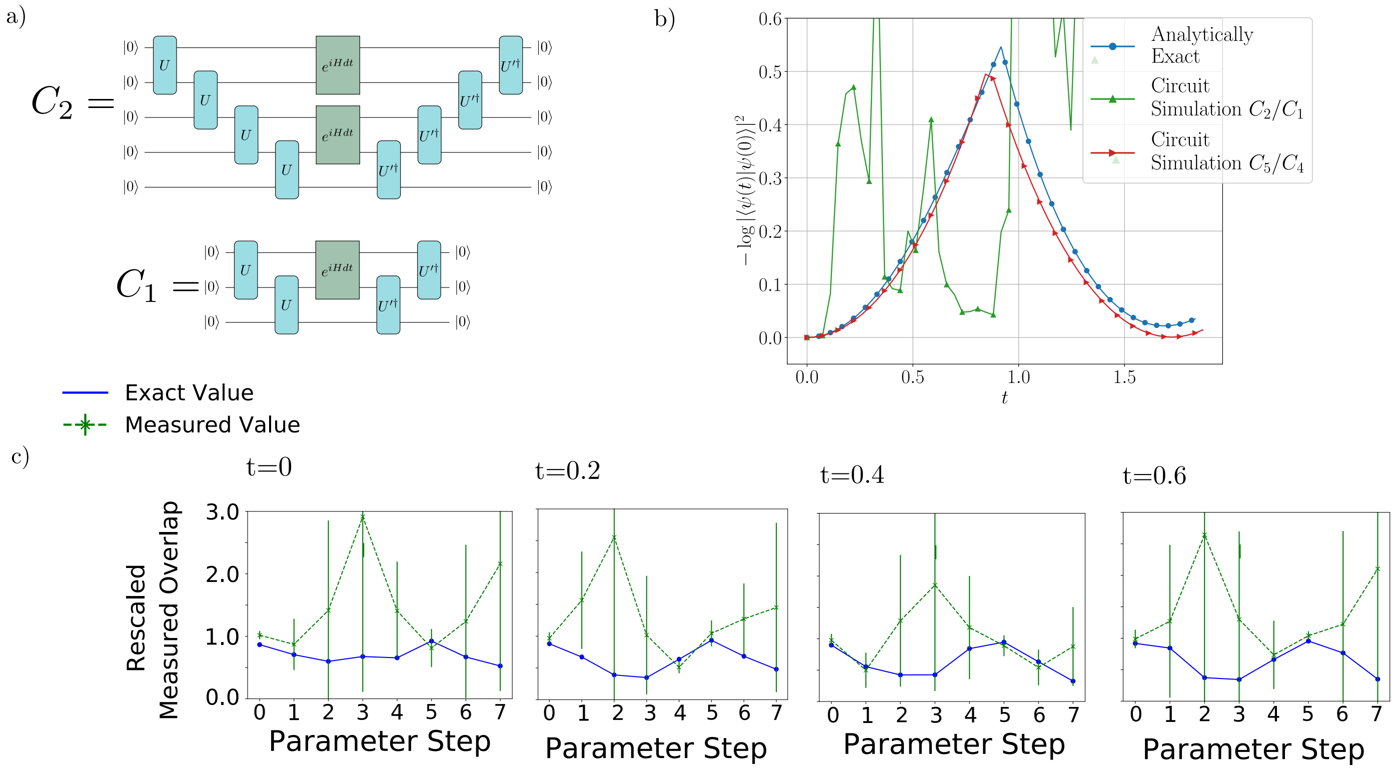


Figure 3.14: **Time Evolution Results II -** a) The overlap is estimated with the power method using $|0\rangle\langle 0|$ as an approximation to the right environment, and taking $\lambda \approx C_2/C_1$. Each value is calculated by measuring the probability of measuring $|0\rangle^{\otimes N}$ at the end of each circuit. b) Classical results of using these circuits as a cost function for time evolution. In blue is the exact time evolution of the TFIM. In green is the curve using C_2/C_1 as a cost function, and in red is the curve using C_5/C_4 as a cost function. C_5/C_4 does a better job at capturing the dynamics during time evolution, and C_2/C_1 fails to capture the dynamics. c) Results on the Rainbow device. The cost function is evaluated through a linear interpolation of the parameters from the initial starting point to the optimal point for each time step in a). Although there is a peak in the cost function at the correct value, it is measured as lower than the initial point, meaning no optimizer would correctly optimize to this point.

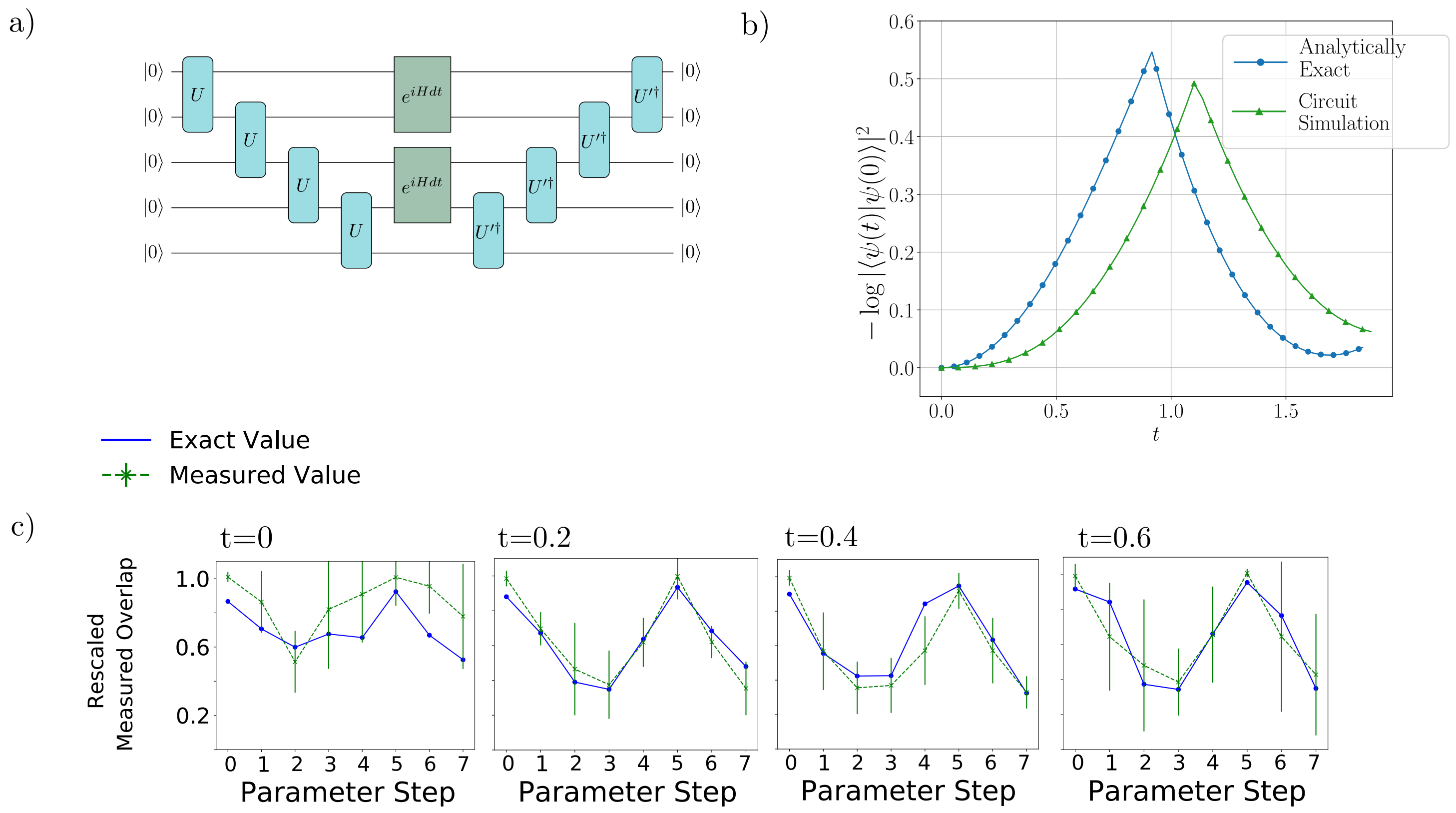
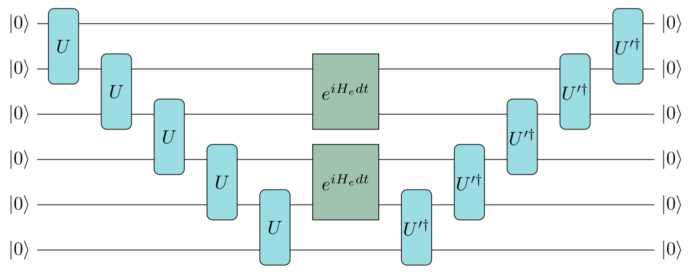
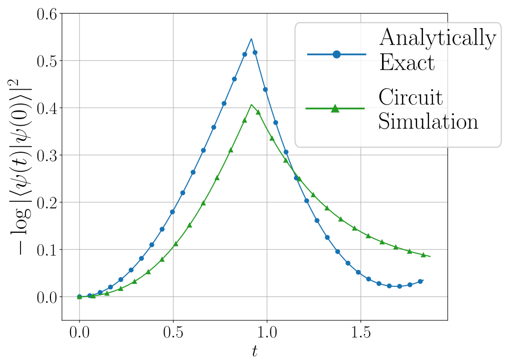


Figure 3.15: **Time Evolution Results III -** a) Approximating the right fixed point with $|0\rangle\langle 0|$ and calculating $\lambda \approx \sqrt{C_2}$. C_2 is calculated as the probability of measuring $|0\rangle^{\otimes N}$, divided by the appropriate Loschmidt echo. b) Classical circuit simulation of this cost function. The circuit can capture key features of the dynamics without noise. c) The cost function evaluated along a linear interpolation of the parameters from the initial starting point to the optimal point at each time step. the measured cost function tracks the correct values well but the peak is not measured at the right point.

a) 

b) 

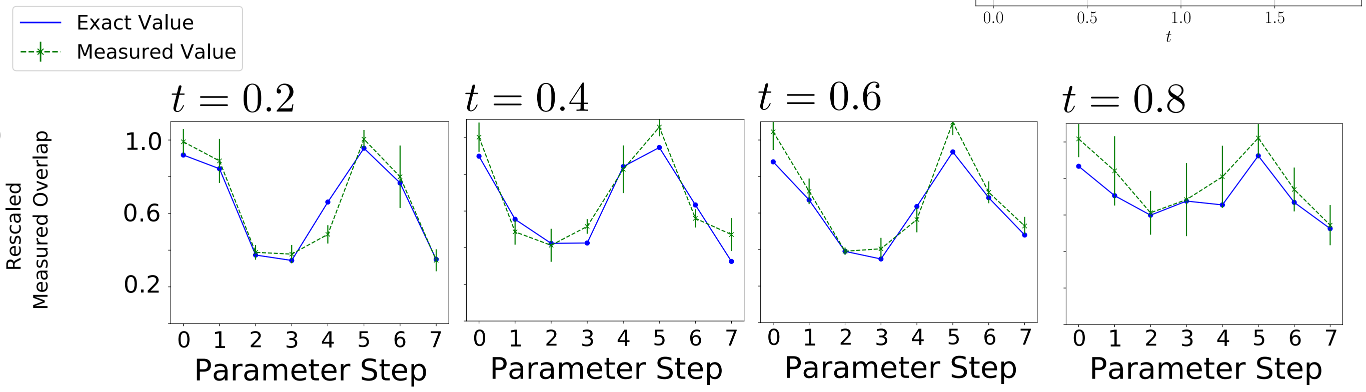
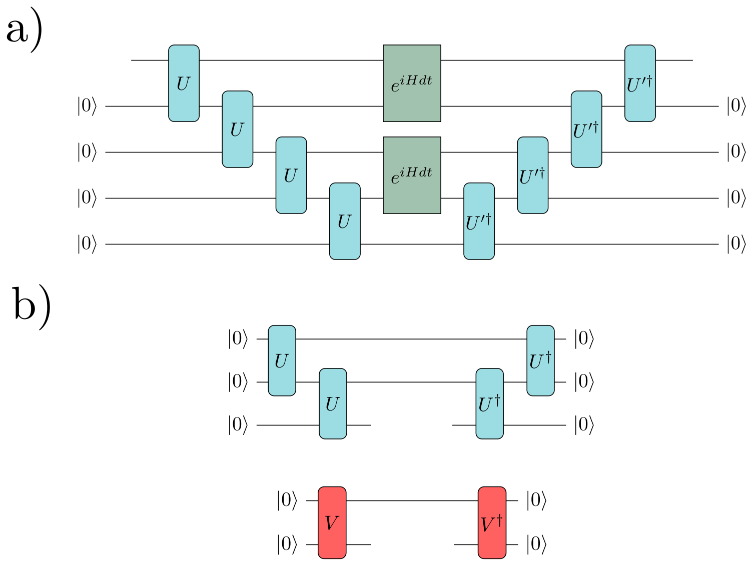
c) 

Figure 3.16: **Time Evolution Results IV -** a) C_2 is measured using U and U' as approximations to the right environment. C_2 is calculated as the probability of measuring $|0\rangle^{\otimes N}$, without post selection. b) Exact analytical results of the time evolution compared to simulated results using these circuits as a cost function. Some features of the dynamical phase transition are captured, although it is not as good as the results seen in Fig 3.13b. c) Cost function measured along a linear interpolation of the parameters between the initial starting point and the optimal point. The measured values again track the correct values but the peak is not measured at the correct point.



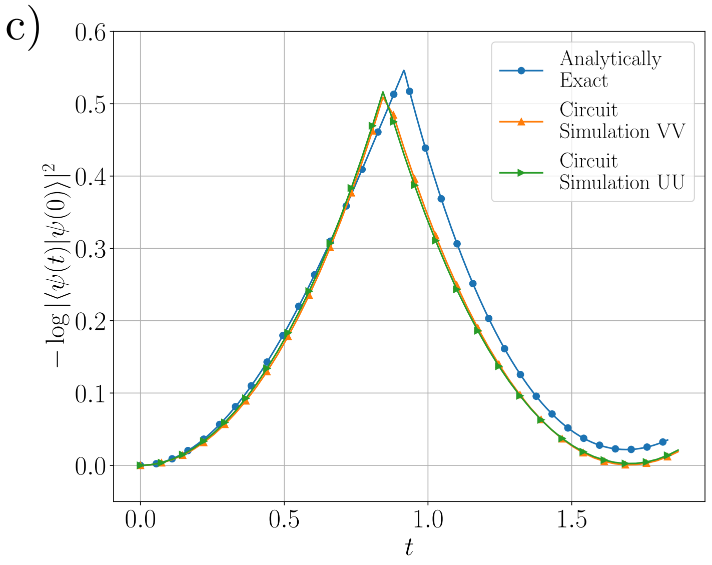


Figure 3.17: **Time evolution Results V -** a-b) The right environment is approximated using two different approximations. One is V, the right environment of the individual MPS, and the other is two copies of U. c) Analytical results compared to simulations of the cost functions without errors. Both these results are considerably closer to the true results compared with the cost functions considered above.

3.2 Summary

This chapter has outlined two alternative algorithms for time evolving translationally invariant MPS states on quantum devices. These algorithms scale efficiently with bond dimension, leaving open the possibility for large bond dimension time evolution of quantum MPS on future hardware. This chapter introduced the quantum TDVP algorithm for time evolution of translationally invariant states. This algorithm requires requires approximation of the right and left environment tensors of the mixed transfer matrix to estimate the overlap, which is then maximised. Two algorithms are presented to find the right and left environments. A slower variational algorithm is shown to effectively capture dynamical phase transitions in Loschmidt echos when a state is time evolved under the TFIM across the critical state, and capture scarred states in the PXP model. A non-variational algorithm for approximating the right and left eigenvectors using the power method was also introduced. This algorithm is faster, and more suitable for current NISQ hardware. This method was tested on the Google Rainbow device, where it was shown that the overlap of two translationally invariant states could be accurately calculated using rescaling to correct for depolarizing errors. These tests form the basis for future experiments on NISQ devices, where complex phenomena can be directly simulated on the device with the time evolution of low bond dimension states.

Chapter 4

Extensions

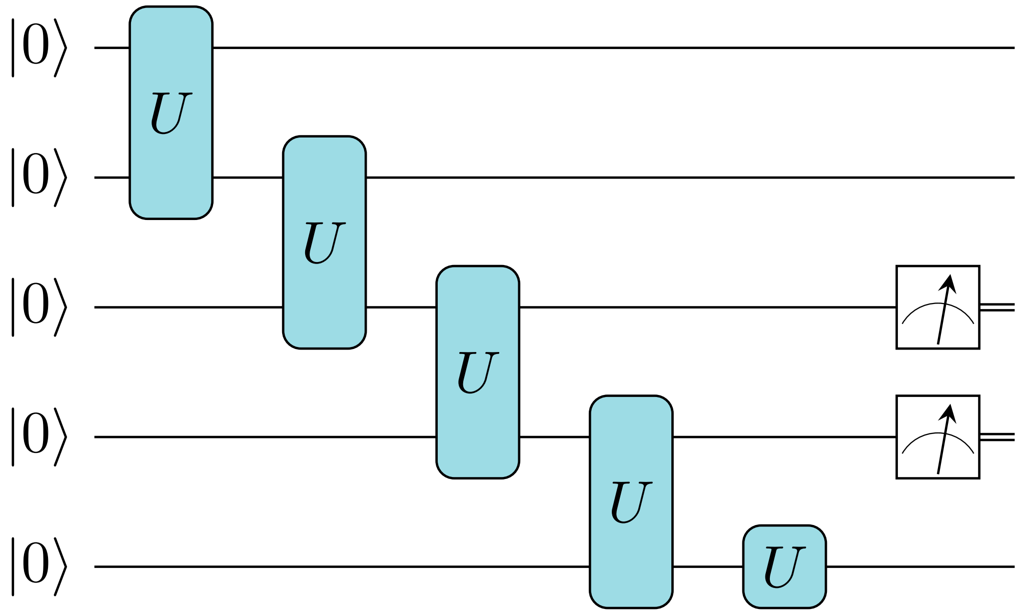
This chapter contains extensions to the work in the previous two chapters, on the representation and time evolution of quantum matrix product states. The content in this chapter represents avenues for future research. I present qubut-efficient circuits, equivalent to the circuits presented in the previous two chapters, using mid-circuit measurements to achieve greater qubit efficiency. I also present extensions of the quantum MPS formalism to quantum thermofield MPS, for the simulation of thermal states among other uses.

4.1 Mid Circuit Measurement

Multiple near-term hardware implementations permit mid-circuit measurement and resetting of individual qubits, including IONQ's, Honeywell's, and IBM's hardware. Access to mid-circuit measurement and resetting gives a means for qubit efficient quantum MPS circuits, using fewer qubits in exchange for larger gate depth[42]. Such a trade-off could be favourable for ion-trap qubits, which have high fidelity gate but are often limited in qubit number.

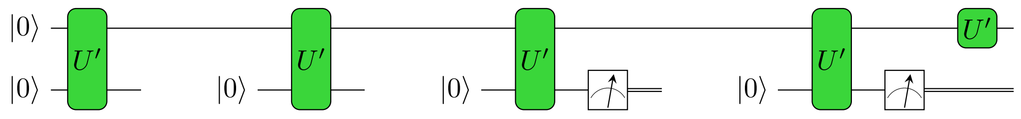
Finite Quantum MPS

Consider a finite quantum MPS circuit with a measurement on two of the physical legs



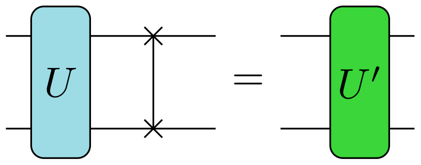
(4.1)

We refer to these as *space-like* quantum MPS, since distance between physical sites is mirrored by a distance between the qubits themselves on the device. The equivalent qubit efficient circuit is given by



(4.2)

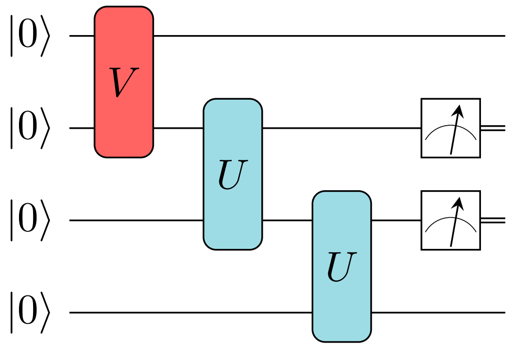
We refer to these as *time-like* quantum MPS, since distance between physical sites is encoded as a time between application of unitaries on a fixed pair of qubits. Time-like unitaries are related to their space-like counterparts by a SWAP gate applied to the output legs:



(4.3)

Translationally Invariant Qubit Efficient Circuits

Qubit-efficient circuits can represent translationally invariant quantum MPS states. For a translationally invariant quantum MPS state with an environment tensor V,



(4.4)

the qubit-efficient circuit is given by

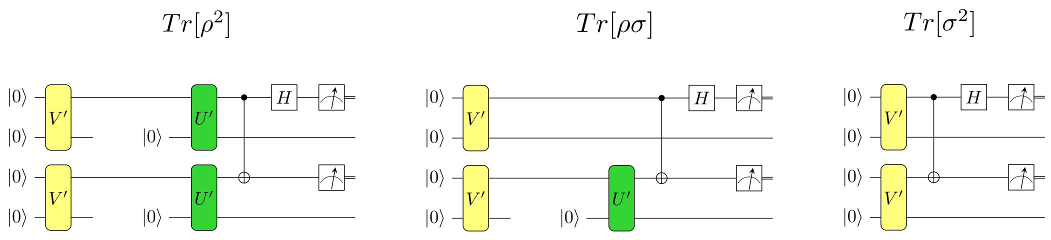
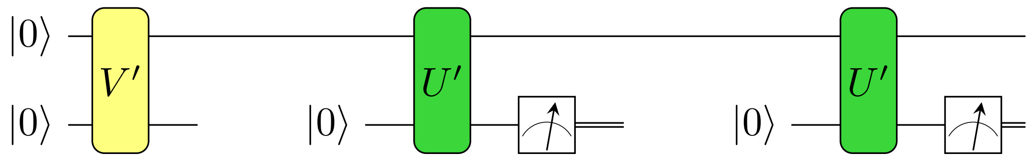
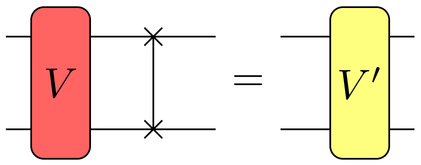


Figure 4.1: **Time-like Trace Distance Circuits:** The three circuits to calculate the trace distance to calculate the consistency between a U, V pair, using qubit efficient mid-circuit measurements.



(4.5)

where the time-like environment tensor is related to the space-like environment tensor by



(4.6)

Time-like SWAP tests

Given a time-like representation for translationally invariant quantum MPS, the next step is to use time-like quantum circuits to find the environment tensor. The SWAP test, as defined in Eqn. 2.1 can be adapted for use in time-like circuits. The circuits in Fig 4.1 are used to calculate the three terms in the trace distance formula 2.4. These three elements are sufficient to represent and optimize translationally invariant time-like MPS states on quantum hardware with mid-circuit measurements.

4.2 Qubit Efficient Time Evolution

The time evolution algorithms introduced in Chapter 3 can be translated to qubit efficient circuits when mid-circuit measurements are available. Ion-trap devices can execute these circuits as they permit mid-circuit measurements. Mid-circuit measurements offer a qubit-efficient way to estimate the environment of translationally invariant MPS circuits. Calculating the environments is one of the more computationally challenging parts of time-evolution algorithms using translationally invariant quantum MPS. Running multiple copies of the state tensors before introducing time evolution operators estimates the environment using the power method directly. This method trades qubit number for increased circuit depth. For qubit-poor devices with good gate fidelity, this may be a worthwhile trade-off.

Fig 4.2 shows how to map time evolution circuits to a qubit-efficient circuit. Fig 4.2a is an alternative way to measure the overlap between a time evolved state and a candidate quantum MPS. A multi-qubit SWAP test is used instead of applying the inverse of the candidate state. The Trotterized time evolution operators applied for half the time step on each half of the circuit. This symmetry makes it easier to see the mapping to qubit efficient circuits but is otherwise not necessary. Fig 4.2b shows the same circuit with mid-circuit measurements. The single multi-qubit SWAP test is replaced with multiple, smaller SWAP tests performed over time. The measurement statistics that build up over time are used to calculate the overlap between the two states. Future work will involve the execution of the time-like quantum TDVP algorithm on NISQ hardware to assess the trade-off between circuit depth and accuracy of overlap estimates. Deeper circuits are equivalent to higher order power method approximations, but are more exposed to noise.

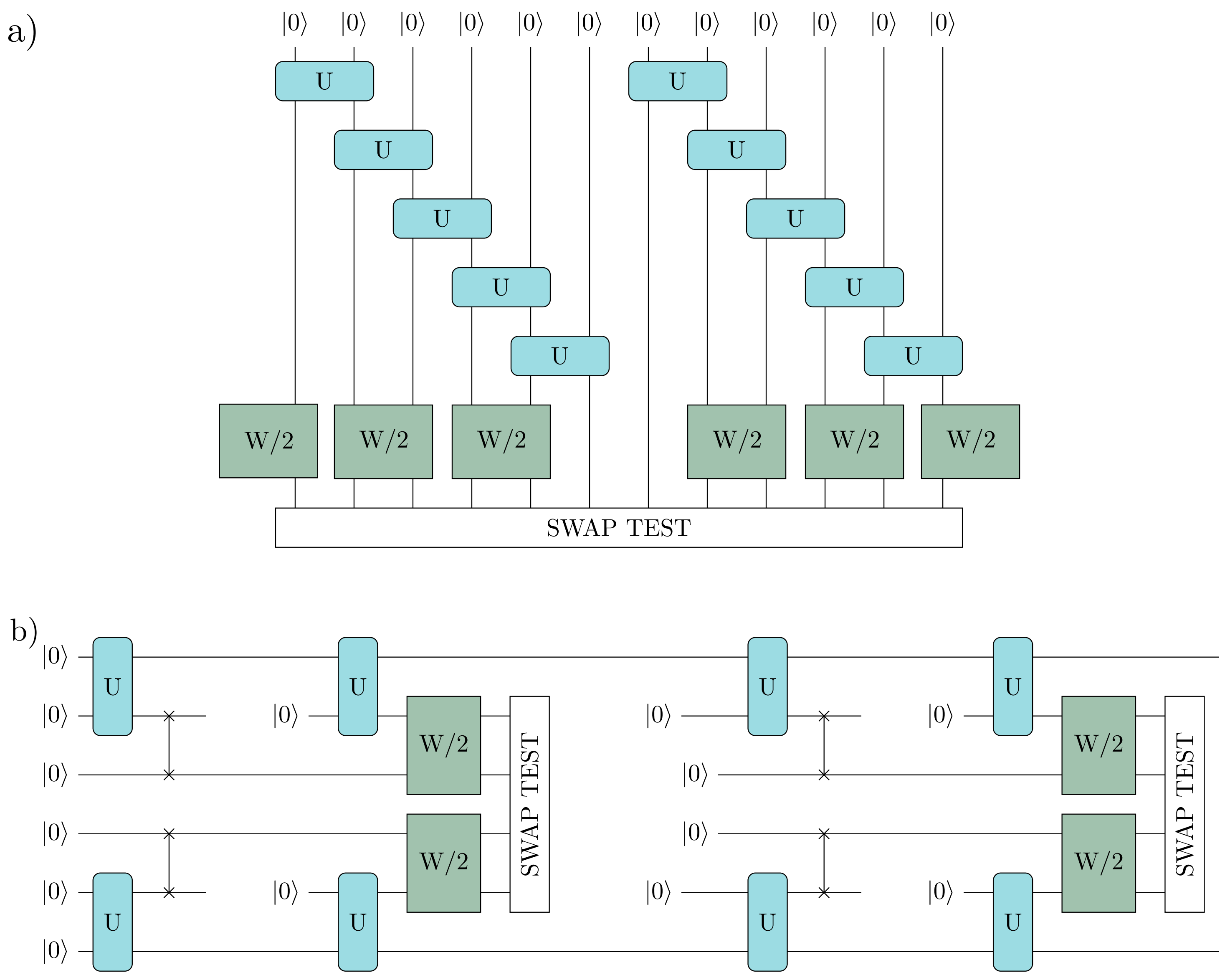


Figure 4.2: **Qubit efficient Time Evolution -** a) Circuit to calculate the overlap between a time evolved state and a candidate bond dimension 2 state. The circuit is split in half and the overlap is measured with a SWAP test. The time evolution operator, $W/2$, on each side is the same time evolution operator applied for half the time step. b) The same circuit, with mid-circuit measurements. The state tensor legs are permuted so the auxiliary leg remains on the same qubit on each side. The overlap is calculated using the measurement statistics that build up across time. The environment is implemented with repeated applications of the transfer matrix before the time evolution operator is introduced.

4.3 Generating Symmetric States

There exist situations where one may be interested in generating symmetric MPS states. Such states appear in the study of *thermofield MPS*. Thermofield MPS are used in the study of thermalization during time evolution[49], and in the study of black holes in holographic quantum systems[50]. This section introduces thermofield MPS states and a shallow factorization of thermofield unitaries that is NISQ friendly and permits the generation of symmetric thermofield MPS states.

4.3.1 Thermofield States

Consider a general quantum mixed state

$$\rho = \sum_{\alpha} \gamma_\alpha |\psi_\alpha\rangle\langle\psi_\alpha| \tag{4.7}$$

where γ_α represents the (classical) probability that the wavefunction is in the state $|\psi_\alpha\rangle$. Such states naturally appear in studying finite temperature quantum systems. At equilibrium, quantum states follow a Boltzmann distribution given by

$$\rho = \sum_{\alpha} e^{-\beta E_\alpha} |E_\alpha\rangle\langle E_\alpha| \tag{4.8}$$

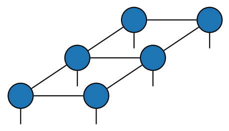
where β is the inverse temperature of the system at equilibrium. It is well known that such mixed states can always be *purified*, whereby a pure state is found such that the expectation values of all local operators on the mixed state and pure state are equal. A purified state acts on a larger Hilbert state to the mixed state. A symmetric purification of the mixed state ρ is given by

$$|\psi_{pure}\rangle = \sum_{\alpha} \sqrt{\gamma_\alpha} |\psi_\alpha\rangle \otimes |\psi_\alpha\rangle \tag{4.9}$$

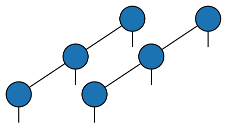
where now $\sqrt{\gamma_\alpha}$ is a probability amplitude. Any local observable on ρ is equal to the the corresponding expectation value on $|\psi_{pure}\rangle$ obtained by tracing out the other Hilbert space,

$$Tr[\hat{O}\rho] = \langle\psi_{pure}|\hat{O} \otimes \mathcal{I}|\psi_{pure}\rangle \tag{4.10}$$

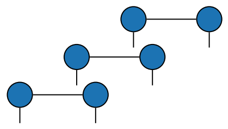
It is also possible to represent the state on each Hilbert space as an MPS, with bonds between the two Hilbert spaces generating the statistics of the mixed state when one state is traced out. This is known as a *thermofield MPS*[49]. Thermofield double states can be graphically represented as two copies of MPS with contracted bonds between the two Hilbert spaces

$$|\psi\rangle = \tag{4.11}$$


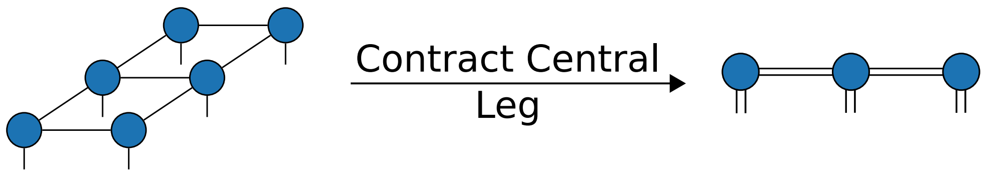
where non-zero entanglement between the two Hilbert spaces will be known as *admixture*. In the absence of admixture this represents two pure states,

$$|\psi_{pure}\rangle = \tag{4.12}$$


in which the traced out state remains pure. The infinite temperature state also has a particularly simple representation in this picture, given by the expression

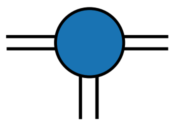
$$|\psi_{thermo}\rangle = \tag{4.13}$$


in which a pair of maximally entangled states are shared across each site. Thermofield states can be explicitly represented as MPS by contracting the central legs,



(4.14)

where the object on the right is an MPS with a squared bond dimension and a squared physical dimension. The central tensors are given by the expression



$$= A^{(\sigma,\sigma')}_{(i,i')(j,j')} = A^{\Sigma}_{I,J} \tag{4.15}$$

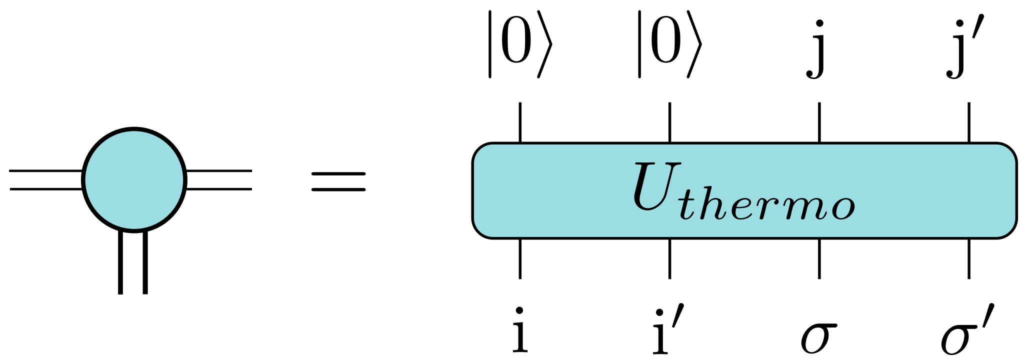
where the doubled legs are grouped together. The tensors satisfy a symmetry where the tensors should be equal under simultaneous permutation of the auxillary and physical legs,

$$A^{\Sigma}_{I,J} = A^{\tilde{\Sigma}}_{\tilde{I},\tilde{J}} = A^{(\sigma',\sigma)}_{(i',i)(j',j)} \tag{4.16}$$

which is necessary to ensure the states across the two Hilbert spaces are equivalent.

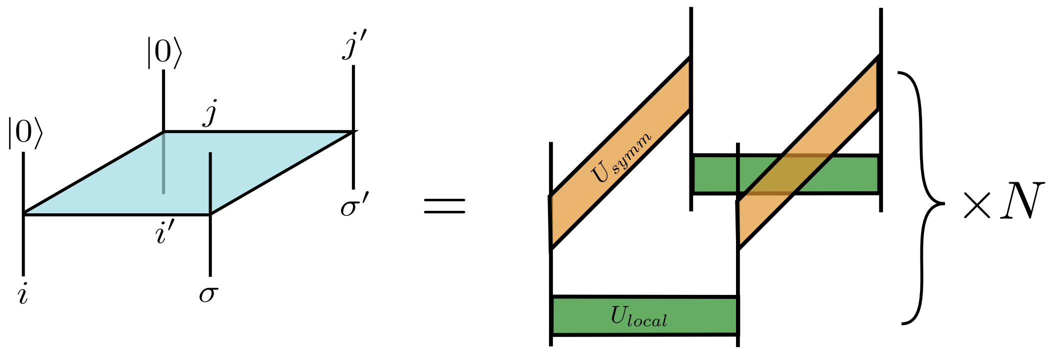
4.3.2 Quantum Thermofield MPS

The doubled tensor can be embedded in a unitary matrix with one qubit for each physical and auxiliary leg,



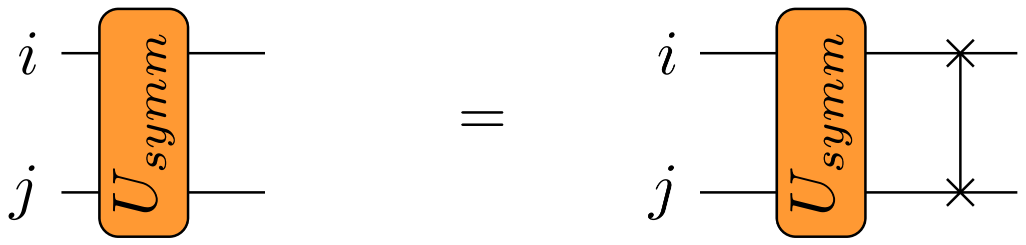
(4.17)

Arbitrary 4 qubit unitaries will not satisfy the symmetry properties in Eqn. 4.16. To be suitable for near-term simulation a shallow decomposition of the unitary needs to be found which satisfies the required symmetry conditions, Eqn. 4.16. One possible decomposition is the following,



(4.18)

where U_{local} is a fully general 2-qubit unitary which acts on a single Hilbert space, and U_{symm} is a symmetric unitary acting across Hilbert spaces which is invariant under a swapping of the output qubits. To generate the correct symmetries the U_{local} unitaries acting on the two physical legs are restricted to be the same unitary for each tensor. The symmetric gates satisfy the equality



(4.19)

which reflects invariance under permutations of the output legs. Given such a symmetric gate, it is possible to construct thermofield MPS states that satisfy the required symmetry properties. To be suitable for applications on NISQ devies this symmetric gate needs to be factorizable into shallow depth circuit.

Structure of $\mathbf{U_{symm}}$

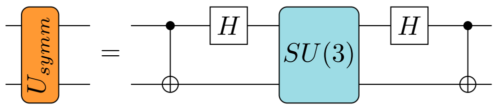
For dealing with states of fixed symmetry it is useful to work with the Bell basis. The four Bell basis states are

$$|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$
$$|\Phi^-\rangle = \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle)$$
$$|\Psi^+\rangle = \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle)$$
$$|\Psi^-\rangle = \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle)$$

where the 4 basis states are either symmetric under qubit swapping ($|\Phi^+\rangle, |\Phi^-\rangle, |\Psi^+\rangle$), or anti-symmetric, ($|\Psi^-\rangle$). Any symmetric two qubit state can be expressed as a linear combination of the three symmetric Bell states. This gives a clear way to generate arbitrary symmetric states in the Bell basis, by mixing between the symmetric terms, and excluding the antisymmetric term. Such an interaction will take the form of a block diagonal two-qubit gate

$$U_{symm}^{(Bell)} = \begin{bmatrix} \multicolumn{3}{c}{\boxed{SU(3)}} & \begin{matrix} 0 \\ 0 \\ 0 \end{matrix} \\ 0 \quad 0 & 0 & e^{i\alpha} \end{bmatrix} \tag{4.20}$$

where the unitary is given in the Bell basis. U_{symm} as constructed will map symmetric states to symmetric states. Any $U \in SU(3)$ can be embedded in the upper left block of the symmetric gate. Mapping this unitary from the Bell basis back to the computational basis gives the following general form for U_{symm}



(4.21)

where the gate labelled $SU(3)$ is the gate given Eqn. 4.20.

Examples of SU(3) Gates

There exist multiple examples of low-level $SU(3)$ gates that can be implemented on near term hardware. The Sycamore interactions are particularly suitable for this. The native interactions on the Sycamore architecture belong to the family of FSim gates[41]. The FSim interaction is given by

$$F(\theta, \psi) = e^{-i\theta(X\otimes X+Y\otimes Y)/2-i\psi Z\otimes Z/4} \tag{4.22}$$

which has the required form to generate symmetric interactions. Therefore, any future implementation of thermofield MPS on NISQ hardware will benefit from naturally low depth implementations of the entangling gates in the thermofield decomposition presented here.

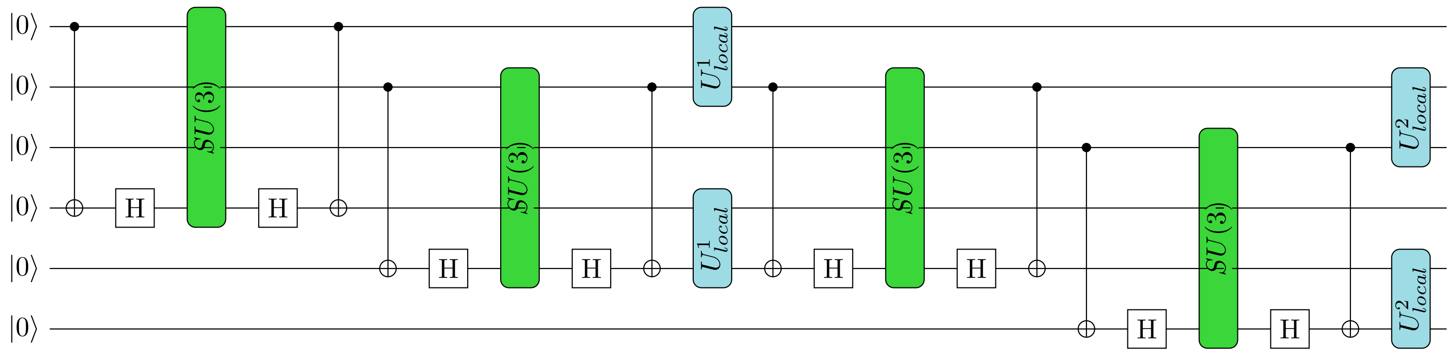


Figure 4.3: **Thermofield Circuit -** A circuit to create a thermofield MPS state that is invariant under simultaneous permutation of the auxillary and physical legs. The symmetric gates are decomposed into a Bell basis transformation and an $SU(3)$ gate as defined above. Non-local interactions in the 1D picture above are local on 2D nearest neighbour circuits, by picturing the two sets of qubits on which the local MPS are defined to lie on-top of eachother. Note two copies of an MPS state given by the blue tensors. These are connected with symmetric entangling gates.

Thermofield Circuit

Putting all this together a circuit can be constructed where the output state is a thermofield MPS guaranteed to be invariant under simultaneous permutation of the auxillary bond legs and the physical legs. Such a circuit is given in Fig 4.3.

4.4 Summary

In this chapter, I introduce extensions of the ideas introduced in the previous two chapters, which are avenues for future work. I have introduced qubit-efficient translationally invariant quantum matrix product states, which are equivalent to quantum matrix product states but can be expressed with fewer qubits. This is achieved by substituting qubit number for increased gate depth. This is possible on hardware supporting mid-circuit measurements, such as ion-trap devices. I also present an extension to the quantum matrix product framework to include quantum thermofield states. To create thermofield states, it is required to generate states symmetric across two sets of qubits. I demonstrate the existence of variational classes of such states, and argue that such states can be realized with low gate depth on near-term hardware.

Chapter 5

MPS Initialisation

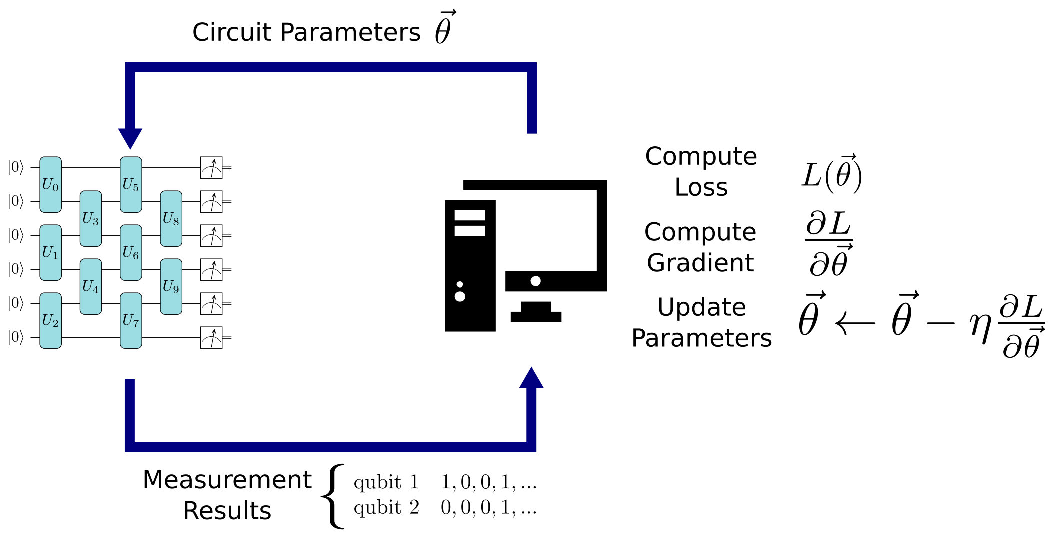
This chapter is based on the publication Matrix Product State Pre-Training for Quantum Machine[5]. In this work we develop an initialization scheme applicable to multiple near-term quantum machine learning tasks, utilising classical optimization of matrix product states. MPS are trained classically as solutions to classical machine learning tasks, using a combination of imaginary time evolution, and DMRG-inspired optimization methods for image recognition. The optimized MPS are then translated into quantum circuits, utilising in part the mapping between classical and quantum MPS outlined in [3]. We demonstrate numerically that this initialization is able to mitigate the impacts of vanishing gradients when optimizing variational circuits. We do this for finding ground states of Hamiltonians, combinatorial optimization problems, and for image classification. I contributed to this work with the initial development of how to use quantum MPS to initialize quantum circuits. I then wrote code to optimize classical MPS using the non-local iTEBD algorithm outlined in this chapter. Finally I conducted the quantum simulations for all the problems in the paper starting from the pre-trained classical MPS state and from randomly initialized and identity initialized-circuits.

Parametrised quantum circuits (PQCs) have been at the centre of attempts to achieve computational advantage on NISQ devices[51, 52, 53]. Many problems of scientific and commercial interests have been mapped to PQCs and shown to work on small problem instances that can run on current simulators and NISQ hardware. The most common approach is to parametrise a quantum circuit with a set of angles and define a loss function over these angles to be minimised by a classical optimiser. The quantum hardware evaluates difficult-to-simulate loss functions, while the classical hardware uses the outputs of the quantum device to calculate the subsequent inputs to reduce a loss function. These are known as Hybrid quantum-classical algorithms (HQCA). A general schematic of HQCAs is shown in Fig 5.1.

A significant hurdle in the success of HQCAs is the inherent difficulty in optimising these quantum circuits. It is well documented that the loss landscape of PQCs makes it challenging to optimise the parameters. One problem with the loss landscapes is the issue of vanishing gradients[54]. The gradients of the loss function in PQCs tend to fall to zero as the size of the circuits increase. It is not just the number of qubits and the depth of a circuit that results in vanishing gradients. Ansatz expressibility, entanglement between measured and un-measured qubits in the circuits, and circuit noise can all result in barren plateaus[55, 56, 57].

Several techniques have been proposed to make PQCs easier to optimise. A number of initialisation techniques have been proposed[58], new optimization techniques[59], and defining local cost functions as optimisation targets[60]. This chapter introduces a novel initialisation scheme based on tensor network algorithms. Tensor network-based methods are state of the art for numerical simulations of 1D and 2D systems. Tensor networks can also be used to solve optimisation problems, such as portfolio optimisation, and are competitive with commercial solvers[61]. Tensor networks are effective classifiers for several machine learning tasks, including image classification and anomaly detection[62, 63]. Tensor networks can be trained classically as solutions to machine learning problems and then mapped to quantum circuits which are precisely as good candidates for the solution as the classical tensor network. This method allows PQCs to be classically pre-trained, reflecting the methods used in state-of-the-art language and image modelling, where classical neural networks benefit from pre-training on large a corpus of data before being fine-tuned on their task of interest. Unlike in the classical machine learning setting, the pre-training of quantum tensor networks doesn't explicitly aim to improve the prediction quality but instead increases the trainability of the circuits by starting in a part of parameter space closer to the target state than randomly sampled parameters. In doing so, perhaps pre-training can mitigate issues of vanishing gradients and difficulties during learning.

The following sections outline the classical pre-training techniques before



Circuit Parameters $\vec{\theta}$
$|0\rangle$
U_0
U_1
U_2
U_3
U_4
U_5
U_6
U_7
U_8
U_9
Compute Loss
$L(\vec{\theta})$
Compute Gradient
$\frac{\partial L}{\partial \vec{\theta}}$
Update Parameters
$\vec{\theta} \leftarrow \vec{\theta} - \eta \frac{\partial L}{\partial \vec{\theta}}$
Measurement Results
qubit 1 $1, 0, 0, 1, \ldots$
qubit 2 $0, 0, 0, 1, \ldots$

Figure 5.1: **Hybrid Quantum-Classical Algorithms:** Outline of HQCAs. A classical computer is paired with quantum hardware which can implement gates parametrized by rotation angles, $\vec{\theta}$. The classical computer provides these angles, and gets back the results of a circuit which is hard to simulate classically. The classical computer then calculates a loss function, and updates the parameters following some optimization algorithm.

going through the results of classically pre-training quantum circuits. This chapter is based on work published in *Matrix Produe State Pre-Training for Quantum Machine Learning*[5].

5.1 MPS Optimization

Finding ground states of Hamiltonians is a standard task in many NISQ applications. This is the case when wanting to study the low energy states of a physical system, but it also appears in many other instances. Problems that are not clearly analogous to physical systems can often be solved by mapping the problem to a Hamiltonian and searching for its ground state. An example of this that will be outlined in more detail later is the well-studied MaxCut problem.

5.1.1 Imaginary Time Evolution

Imaginary time evolution is a technique to find the ground state of a Hamiltonian. Consider the Schrodinger equation

$$i\hbar\frac{\partial|\psi\rangle}{\partial t} = \hat{H}|\psi\rangle \tag{5.1}$$

which has solutions in terms of the energy eigenbases of $\hat{H}$, $|E_i\rangle$ with energy eigenvalues E_i

$$|\psi\rangle = \sum_i c_i|E_i\rangle \tag{5.2}$$

After evolving for a (real) time t the solutions are given by

$$|\psi(t)\rangle = \sum_i c_i(t=0)e^{-itE_i/\hbar}|E_i\rangle \tag{5.3}$$

with the phase of each term oscillating with a frequency given by the energy of the state. Reparametrising the time with imaginary time, $\tau = it$, this gives

$$|\psi(t)\rangle = \sum_i c_i(t=0)e^{-\tau E_i/\hbar}|E_i\rangle \tag{5.4}$$

The contribution of each term decays exponentially to zero, where the decay rate is the energy of the eigenstate. In the large τ limit, only the lowest energy state will remain, as this state decays the slowest.

Time Evolving Block Decimation

The Time Evolving Block Decimation (TEBD) algorithm is an MPS time evolution algorithm that uses a Trotter decomposition of a time evolution operator to give a low bond order MPS approximation to a time evolved state.

For this it is assumed that the Hamiltonian is given by a sum of nearest-neighbour two-qubit terms

$$\hat{H} = \sum_i \hat{h}_i \tag{5.5}$$

Then a Trotter decomposition is used to estimate the time evolution operator as sequentially applied local operators

$$e^{-i\hat{H}dt} \approx \prod_i e^{-ih_i dt} \tag{5.6}$$

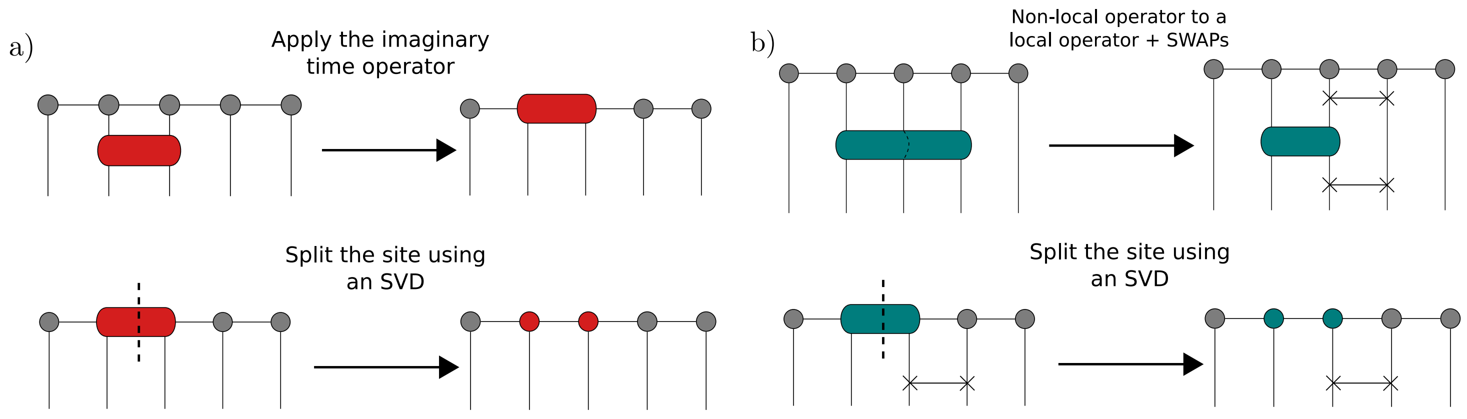

a)
Apply the imaginary time operator
Split the site using an SVD
b)
Non-local operator to a local operator + SWAPs
Split the site using an SVD

Figure 5.2: **TEBD Algorithm:** a) *Local TEBD* - The TEBD algorithm works by applying Trotterized time evolution operators to a MPS. The MPS is projected back down to a low bond order using an SVD after each application of the Trotterized time evolution operator. b) *Non-Local TEBD* - The non-local variant of the TEBD algorithm. Long range interactions are replaced by local interactions and SWAP operators, which can be applied and compressed using the local TEBD algorithm.

To perform the TEBD algorithm each local term is contracted with the MPS, and then this new tensor split using an SVD, keeping only the top k singular values, where k is the desired maximum bond dimension. This procedure is shown graphically in Fig 5.2a. When the time evolution operators in Eqn. 5.6 are replaced with imaginary time evolution operators, $e^{-\hat{h}_i d\tau}$, the TEBD algorithm evolves to the state to a large imaginary time, increasing the overlap with the low energy eigenstates of the Hamiltonian. This can be extended to non-local Hamiltonains, like those encountered in combinatorial optimization problems.

Non-Local Time Evolving Block Decimation

The TEBD algorithm outlined above can be extended to work for Hamiltonians with two-qubit terms that do not act on nearest neighbour sites.

Each long-range operator can be expressed as a local operator with cascades of SWAP operators acting on the physical legs of the lattice between the legs of the operator, Fig 5.2b. This new local operator + SWAP gates can be applied in the same way as the local TEBD algorithm.

5.1.2 DMRG-Inspired Optimisation

Until recently, MPS algorithms like TEBD, TDVP, and DMRG had been used to solve machine learning problems whose solution could be written as the ground state of a Hamiltonian. This formalism isn't very useful when considering the use case of machine learning using data. Instead, a new method was developed, taking inspiration from deep learning and kernel methods[62, 64].

Consider classical data, $\vec{x} \in \mathbb{R}^d$, sampled from a dataset $\mathcal{D}$ with $|\mathcal{D}| = N$ training samples. Each data vector has a label $y \in \{-1, 1\}$. The task is to find a function, f, such that $f(\vec{x}) \geq 0$ for all $\vec{x}$ with $y = 1$, and $f(\vec{x}) < 0$ otherwise, including generalising to unseen data. The proposed MPS method to solve this problem involves embedding the classical data into a higher dimensional vector space. This mapping, $\Phi(\vec{x})$, is defined as

$$\Phi(\vec{x}) = \bigotimes_{i=0}^{d} \phi(x_i) \tag{5.7}$$

where $\phi(x_i)$ is some non-linear function on the single data value. The non-linear mapping used in this work is the sinusoidal mapping,

$$\phi(x_i) = (\sin(\pi x_i/2), \cos(\pi x_i/2)) \tag{5.8}$$

although many others have been explored in the literature. This mapping can be interpreted as mapping each data point to a single qubit spin, where the value of the data point determines that angle of the spin. The total vector is a normalized wavefunction over d spins. The size of this vector is 2^d, exponentially large in the dimensionality of the classical data vector.

The goal is to train a linear classifier, $\mathbf{W}$, on the high-dimensional embedded data. This classification scheme draws inspiration from Support Vector Machines (SVMs), which use the fact that low dimensional data which is not linearly separable can become linearly separable in higher dimensions. Naively the matrix $\mathbf{W}$ would have to be exponentially large to act on the vector $\Phi(\vec{x})$. However one can use a matrix product state approximation to $\mathbf{W}$ so it can efficiently be contracted with the high dimension data. This gives a classifying function

$$f(\vec{x}) = \mathbf{W}_{\mathbf{MPS}}\Phi(\vec{x}) \tag{5.9}$$

where the sign of f is used to classify the binary data. This is easily extended to l possible labels by adding an output leg to the MPS classifier. A square-error loss function is calculated over the dataset

$$L = \frac{1}{N}\sum_{j=0}^{N}(y_j - \mathbf{W_{MPS}}\Phi(\vec{x}_j))^2 \tag{5.10}$$

There are several ways one could train the MPS classifier. In the most straightforward setting, one could calculate this loss function and use back-propagation to update the weights of the MPS tensors directly[65]. Directly updating the MPS tensors is particularly easy when using an auto-differentiation library, like Tensorflow, Pytorch, or Jax. Alternatively a DMRG-inspired optimisation scheme can be used, outlined in Fig 5.3b. Pairs of sites are combined and updated using gradient descent, while all other sites remain frozen. This has the benefit of automatically selecting the MPS classifier's bond dimension during training by removing all singular values below a certain size.

The following section considers how to use these methods to pre-train a quantum circuit, having trained MPS to solve both classical and quantum machine learning problems.

5.2 Initialising Circuits with Quantum MPS

Finding a better quantum circuit initialisation relies on initialising the circuit as a pre-trained classical MPS. Doing so gives an effective initialisation point for future variational quantum algorithms trying to solve the same problem. The ability to initialise a quantum circuit in this way is a unique benefit of the MPS approach to solving machine learning problems. Other classical methods, such as classical neural networks, cannot immediately be used to seed a quantum algorithm in the same way. From here onwards, this procedure will be known as *MPS Pre-training.*

5.2.1 MPS Pre-training

MPS Pre-training has three steps:

1. Train an MPS classically

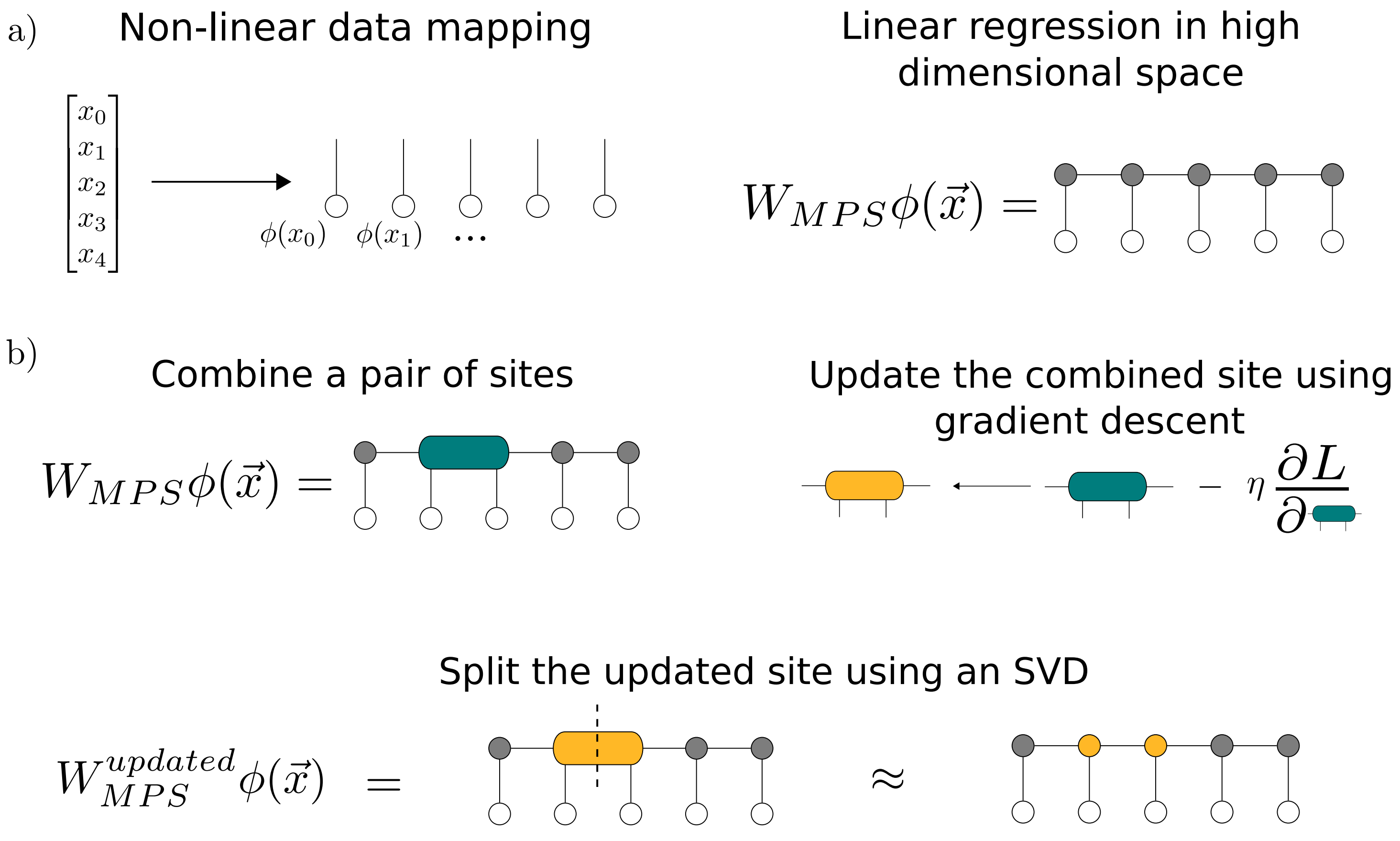

a)
Non-linear data mapping
x_0 x_1 x_2 x_3 x_4
$\phi(x_0)$ $\phi(x_1)$ $\cdots$
Linear regression in high dimensional space
$W_{MPS}\phi(\vec{x}) =$
b)
Combine a pair of sites
$W_{MPS}\phi(\vec{x}) =$
Update the combined site using gradient descent
$- \eta \frac{\partial L}{\partial}$
Split the updated site using an SVD
$W_{MPS}^{updated}\phi(\vec{x}) =$
$\approx$

Figure 5.3: **DMRG-Inspired Classification:** a) Classical data vectors are embedded in a wave function by encoding the data as the rotation angles of individual qubits. Linear classifiers can be trained efficiently by restricting optimization to MPS classifiers acting on this exponentially large dimensional space. b) Sites can be updated by combining pairs of sites and updating them using gradient descent, keeping all other tensors frozen. The updated tensor is then split using an SVD. This DMRG inspired approach is done instead of direct gradient descent on all of the tensor simultaneously.

2. Compile the MPS to an equivalent quantum circuit

3. Continue training of the quantum circuit from this starting point

Step 1 is covered in the previous section. Step 2 follows the procedure outlined in chapter 2. The trained MPS can be put into canonical form, where each tensor is an isometry, and these isometries can is embedded into a unitary matrix. The MPS will correspond to a diagonal staircase of unitaries in the circuit. Finally, one can view this initialized circuit as a PQC where the off-diagonal gates are initialized to the identity. The classical optimizer can now vary all the parameters, including those in the identity off-diagonal gates. This procedure is outlined in Fig 5.4.

The MPS unitaries are converted into rotation angles using Cartan's decomposition, commonly referred to as the KAK decomposition[66]. MPS pretraining proceeds by initialising a brick wall circuit, also known as a quantum neural network, using the longest diagonal of the circuit to initialise with the MPS. MPS Pretraining is evaluated on three tasks.

1. Finding ground states of Hamiltonians

2. Combinatorial optimisation

3. Image classification

5.2.2 Finding Ground States

MPS pre-training can accelerate VQE and improve the task of finding the ground states of Hamiltonians. Particularly for condensed matter Hamiltonians, like the TFIM, this would seem a natural application of the technique, where MPS can capture the ground states of these systems effectively. *Electronic Hamiltonian* MPS pre-training works to accelerate finding the ground states of electronic Hamiltonians. For the electronic ground state of the H_2 molecule, it is demonstrated that on the order of 10^5 fewer function evaluations are needed to find the ground state to within 10^{-8} of the true ground state when using pre-training compared to random initialisation, Fig 5.5b. For Ising models, the benefits of pre-training are also apparent. MPS pretraining was tested on the transverse fielding Ising model at criticality,

$$\hat{H} = \sum_{i=0}^{L-1} J Z_i Z_{i+1} + \sum_{i=0}^{L} g X_i \tag{5.11}$$

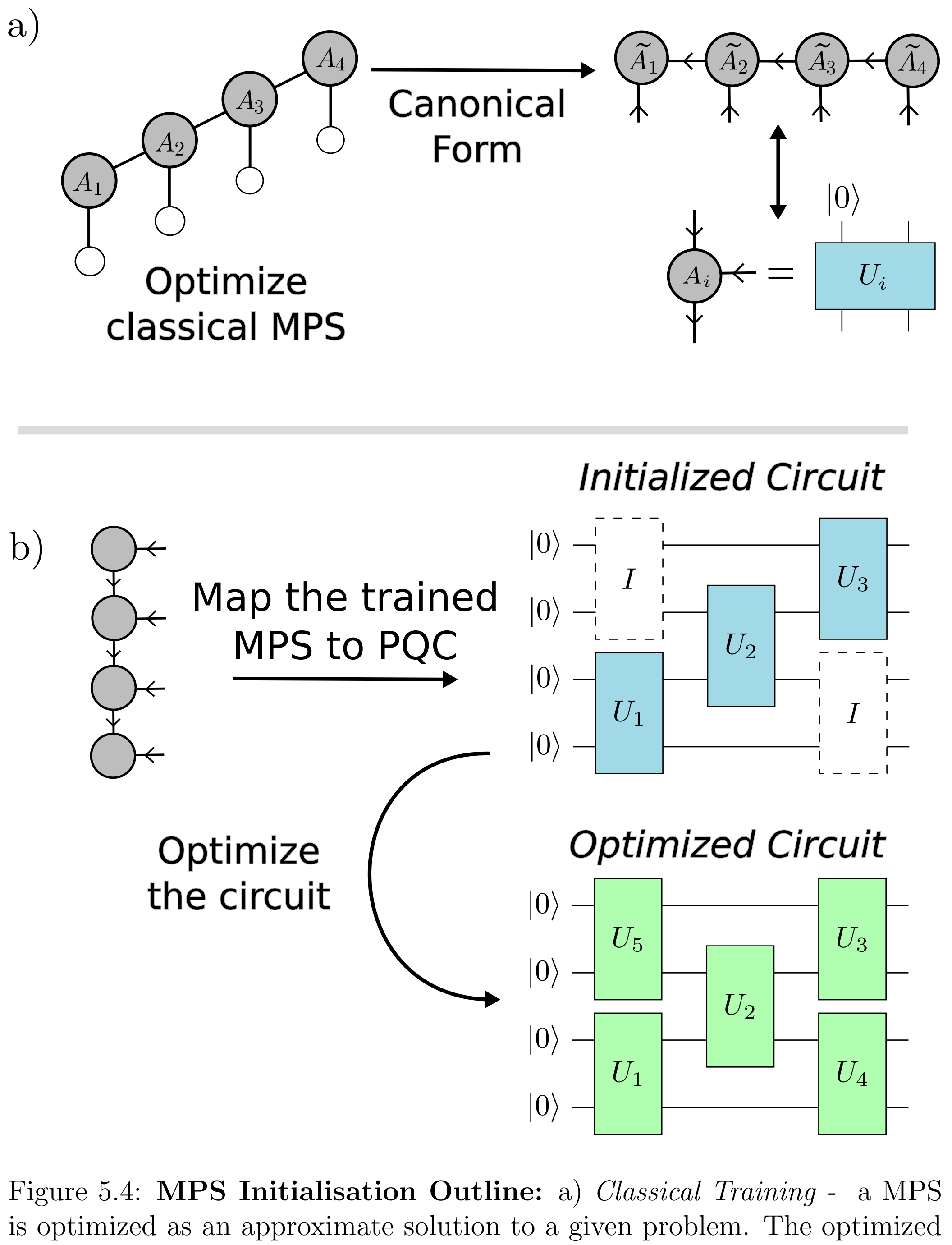


Figure 5.4: **MPS Initialisation Outline:** a) *Classical Training* - a MPS is optimized as an approximate solution to a given problem. The optimized MPS is put into canonical form, where tensors are isometries. In canonical form the tensors are equivalent to unitary matrices acting on a reference state. b) *Initialisation* - The trained MPS is mapped to a quantum circuit. Off diagonal gates are implicitly initialized to the identity. This circuit is as good an approximation to the problem solution as the classical MPS was. The optimizer is then able to vary all the parameters, including those in the gates initialised to the identity.

where $\frac{J}{g} = 1$. At the critical point, long-range correlations make low bond order MPS a worse candidate for finding the ground state; training begins at a more favourable energy than randomly initialised circuits, and proceeds more rapidly, Fig 5.5d. Note that circuits initialised to the identity in this case often failed to optimise, with the identity representing quite a severe and difficult to overcome local minima. The Tilted Field Ising Model was used to test the effectiveness of this initialisation in mitigating the barren plateau. The barren plateau is the exponential decrease in the variance of gradients as the depth and width of a parametrised quantum circuit grows. As shown in Fig 5.5c, pre-trained circuits did not have exponentially decreasing gradient variance. The efficiently trainable MPS initialised circuit contrasts randomly initialised circuits and circuits initialised to the identity.

5.2.3 Combinatorial Optimization

Another set of problems that can be expressed in the language of finding ground states is that of combinatorial optimization. The following section focuses on MaxCut, one of the most well studied combinatorial problems.

The Max Cut optimisation problem begins with a weighted graph, $G(E, V)$ with weights $w_{i,j}$ along each edge. The goal is to find a set of vertices, S, such that the total weight of the edges connecting S to it's complement is maximized. Finding this set of edges is equivalent to finding the ground state of the Hamiltonian

$$\hat{H} = -\sum_{<i,j>} w_{ij}(1 - Z_i Z_j) \tag{5.12}$$

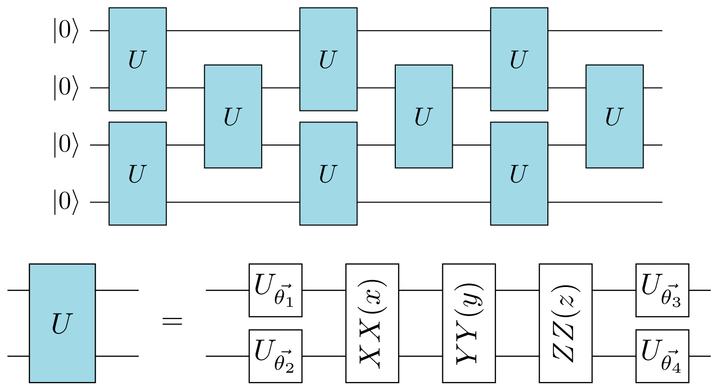
Another set of problems expressible in the language of finding ground states is that of combinatorial optimisation. The following section focuses on MaxCut, one of the most well-studied combinatorial problems.

The MaxCut optimisation problem begins with a weighted graph, $G(E, V)$ with weights $w_{i,j}$ along each edge. The goal is to find a set of vertices, S, that maximises the total weight of the edges connecting S to its complement. Finding this set of edges is equivalent to finding the ground state of the Hamiltonian

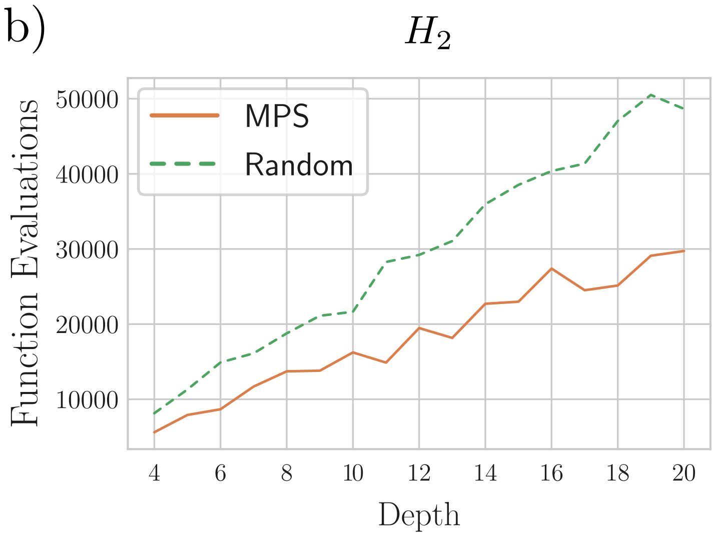
$$\hat{H} = -\sum_{<i,j>} w_{ij}(1 - Z_i Z_j) \tag{5.13}$$

The Hamiltonian contributes 0 whenever nodes i and j are in the same set, and -2 otherwise. The lowest energy state will be a superposition of

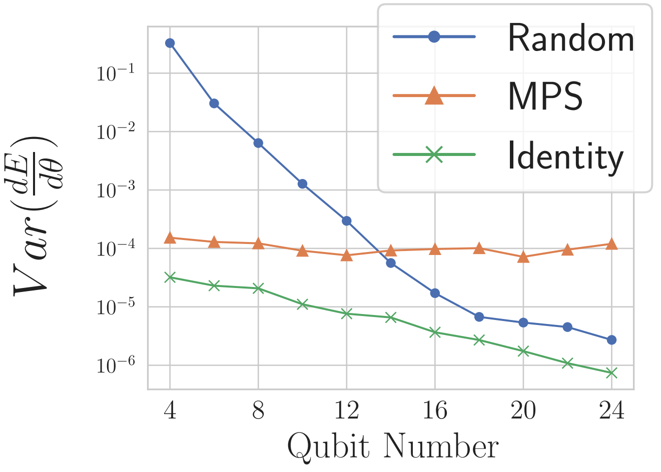
a)



b)



c) $H = \sum_i Z_i Z_{i+1} + \lambda \sum_i X_i + \delta \sum_i Z_i$



d) $H = \sum_i Z_i Z_{i+1} + \sum_i X_i$

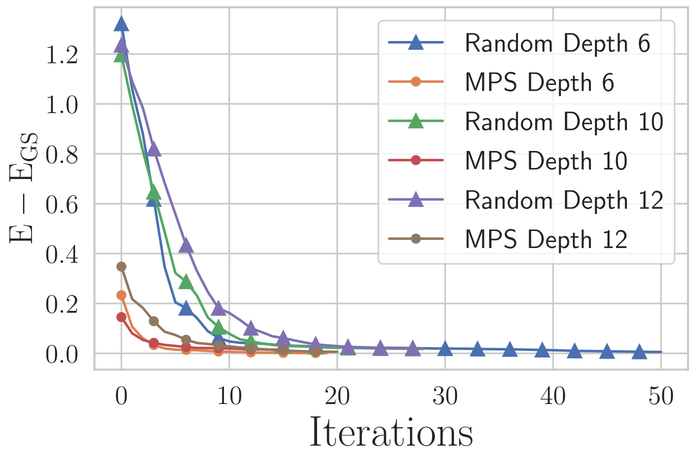


Figure 5.5: **Initialising VQE with MPS:** a) *Quantum Circuit Ansatz* - The ansatz used in these experiments. Shown is a depth 6 circuit acting on 4 qubits. Each 2 qubit unitary, U, is decomposed into 15 parametrized gates using Cartan's KAK Decomposition. b) H_2 *Optimisation* - The H_2 electronic Hamiltonian was optimized, until a good estimate of the ground state energy was found, $E - E_{min} < 1 \times 10^{-8}$. The MPS initialised circuits require thousands of fewer function evaluations at each depth. c) *Non-vanishing gradients* - For the tilted field Ising model we find the variance of the MPS initialised gradients, at initialisation, do not decrease exponentially with circuti depth. This is in contrast to both random and identity optimization. The gradient was calcualted with respect to parameters in the first gate in blocks initialsied to the identity. The tilted field parameters are $\lambda = 1$ and $\delta = 0.5$. d) *TFIM Optimisation* - At all depths tested the MPS initialisation converged with fewer iterations.

product states in which spins are in different sets if there is a large weight connecting them. There is no restriction on the graphs used for MaxCut problems, so this Hamiltonian will generally have long-range interactions. Hence non-local imaginary time TEBD was used to optimise a MPS as the solution to the MaxCut problem given by the graph in Fig 5.6a. Pre-training accelerated optimisation by beginning initialisation in a lower energy state, and subsequent training reached convergence in many fewer gradient updates.

One might assume that matrix product states would be poor initial guesses to solve MaxCut problems. Low bond order MPS will not effectively capture the long-range interactions in the Max Cut Hamiltonian. For the small problem instances tested here, MPS could express the ground state of the Hamiltonian effectively. To mimic the scenario in which the ground state has long-range correlations, so the bond order 2 MPS is unable to capture it, the training of the MPS was restricted to a short imaginary time. Hence the initialised MPS is a poor approximation to the ground state. Even in this case, it is clear that the MPS still provides an effective initialisation point. This may suggest that even for larger problem instances with more entanglement, MPS pre-training could be an effective initialisation scheme.

5.2.4 Image Classification

During training, optimisation schemes can treat PQCs as a differentiable black box, similar to neural network models in classical machine learning. Hence PQCs are a drop-in replacement for neural networks in many machine-learning tasks. One such task is image classification. It is not entirely clear that a so-called *quantum neural network* should provide a benefit over classical methods for such tasks, and it remains to be seen if any relevant classical dataset exists in which quantum processing offers an advantage. As a side note, hand-crafted data sets exist such that quantum classifiers are more potent than their classical counterparts[67]. However, these do not appear relevant to commercial or scientific use cases. This work demonstrates that the Fashion MNIST classification problem, a common image classification problem in the classical machine learning literature, can be accelerated using MPS pre-training. The classifier was trained to distinguish between t-shirts and trousers. Binary classification was considered for simplicity.

A quantum-inspired machine learning model can be trained on data producing an MPS classifier and an embedding map. The MPS classifier has an equivalent quantum circuit as long as the embedding map can be easily

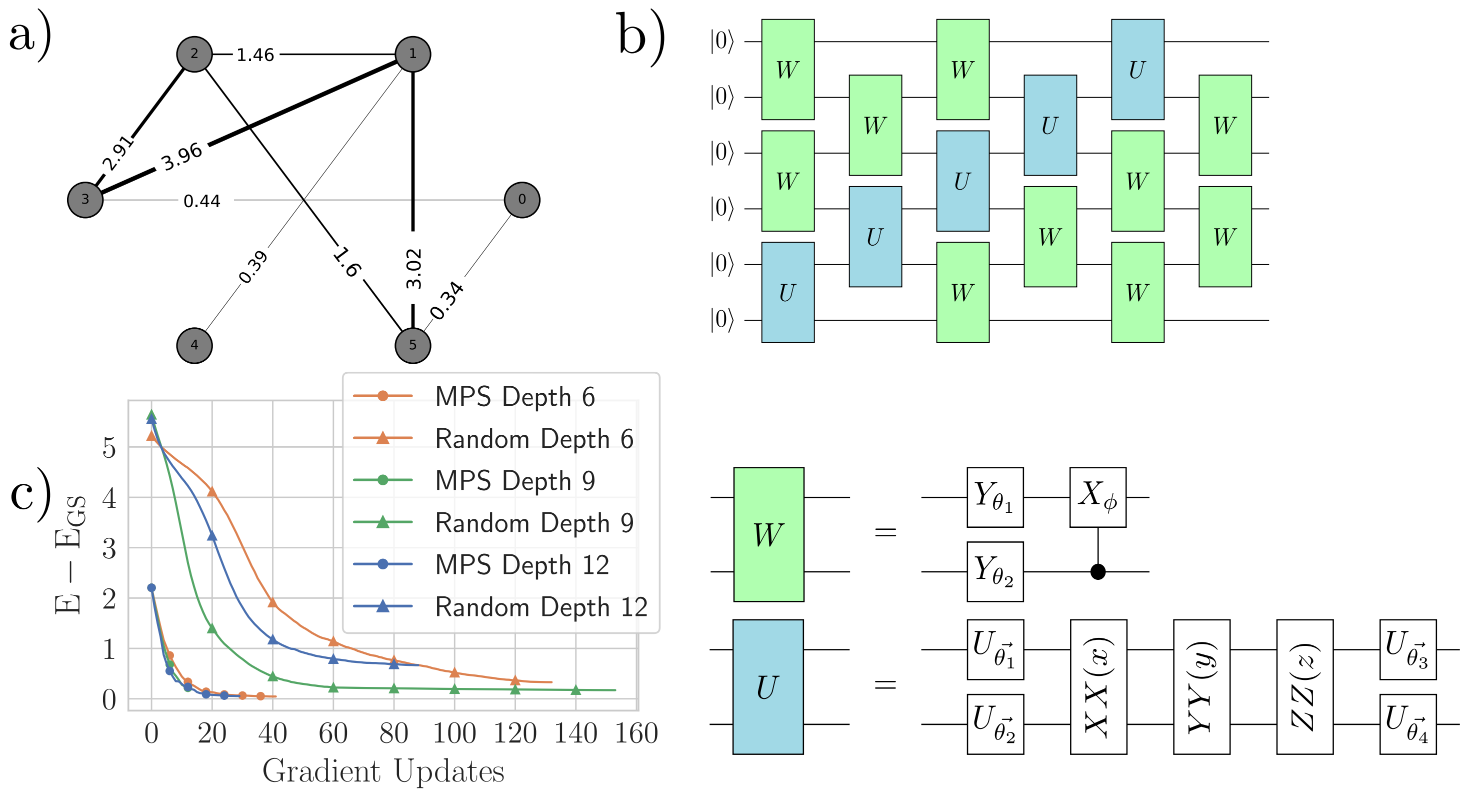


Figure 5.6: **Max Cut Optimization:** a) *Max Cut Graph* - The 6 node Max Cut graph with weighted edges. Each node is encoded as a single qubit in the imaginary time TEBD algorithm. b) *Quantum Circuit Ansatz* - For all experiments the diagonal unitaries are given exactly by the KAK decomposition, and off diagonal unitaries are composed of Pauli Y rotations and controlled X rotations. c) *MPS Pre-training* - MPS initialized circuits converge after fewer gradient steps and converge to a better ground state than randomly initialised counterparts.

realisable as a quantum circuit. The sinusoidal embedding is equivalent to a Pauli Y rotation, where the brightness of the pixel defines the rotation angle. Naively one would use a single qubit per pixel. However, that would require over 400 qubits, far beyond any classical simulators or any currently available hardware. Principle component analysis is used to compress the images. The images were projected onto the principal components calculated on the training set.

The training images were flattened and collected into a matrix $\mathbf{X}$. Then the covariance matrix was computed

$$\mathbf{\Sigma} = \mathbf{X}^T\mathbf{X} \tag{5.14}$$

then an SVD is performed on the matrix $\mathbf{\Sigma}$.

$$\mathbf{\Sigma} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\dagger \tag{5.15}$$

The principal components are the columns of the matrix $\mathbf{U}$,

$$\{\vec{u_1}, \vec{u_2}, \dots, \vec{u_N}\} \tag{5.16}$$

. Taking just the top N principle components of U and finding the inner product of an image with each principal component produces a compressed feature vector that will fit on an N qubit device. The input to the i^{th} qubit from the j^{th} is then given by

$$\tilde{x}_{i,j} = \langle \vec{x_j}, \vec{u_i} \rangle \tag{5.17}$$

The values $\tilde{x}_{i,j}$ are the inputs to the MPS and the quantum classifier. The test images were projected onto the principal components found for the training data set.

Fig 5.8c compares the MPS pre-trained circuits to randomly initialised circuits and circuits initialised to the identity by setting all rotation angles to zero. The MPS pre-trained circuits once again started at lower energy and optimised faster than the other initialisation schemes, Fig 5.8.

5.2.5 Discussion

Comparison to other methods

There have been multiple methods proposed to initialise quantum circuits to improve trainability. The *warm-start QAOA* algorithm is very similar in

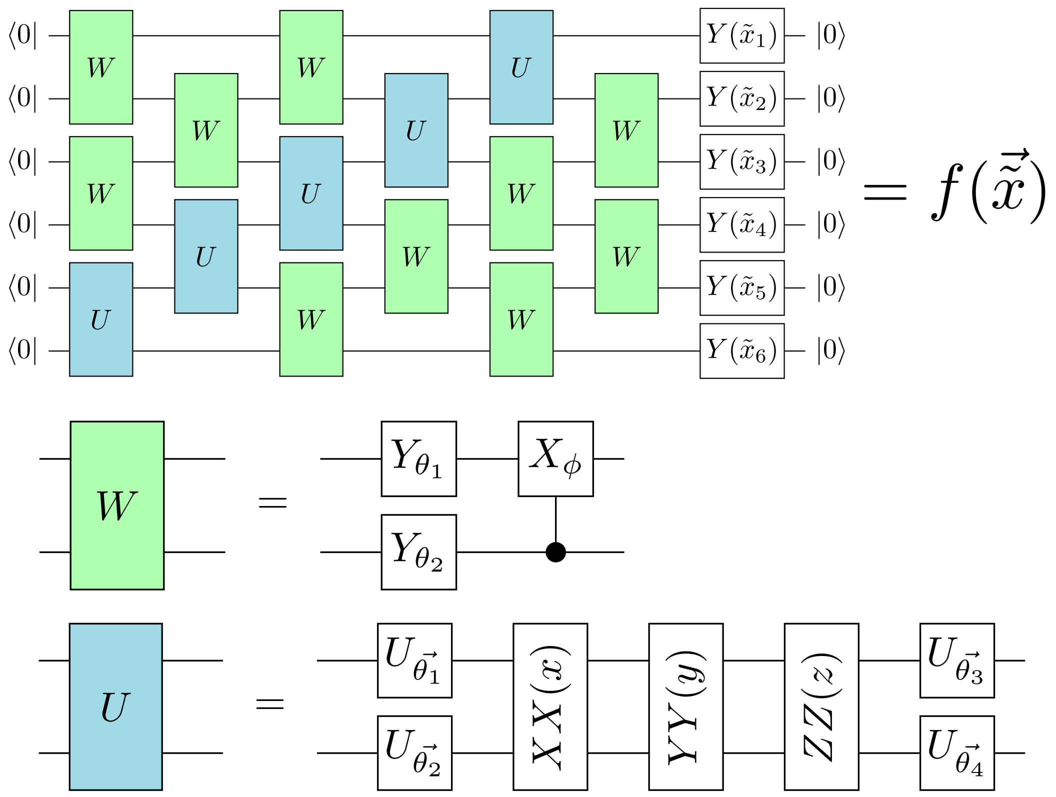


Figure 5.7: **Fashion MNIST Circuits:** *Quantum Circuit Ansatz* - The circuits used in all these experiments. Diagonal unitaries are decomposed using the KAK decomposition whereas off diagonal unitaries are composed of Pauli Y gates and controlled X rotations. The linear classifier is given by a brickwall circuit, and the data uploading is done using Pauli Y rotations parametrised by the projections of the images onto the training set singular values, $\vec{x}$. A binary decision function is calculated as the probability of measuring the all-zero bit string. Binary cross-entropy loss is used, An image is classified as label 0 if $f(\vec{x}) = P(|00\ldots\rangle) < 0.5$ and 1 otherwise

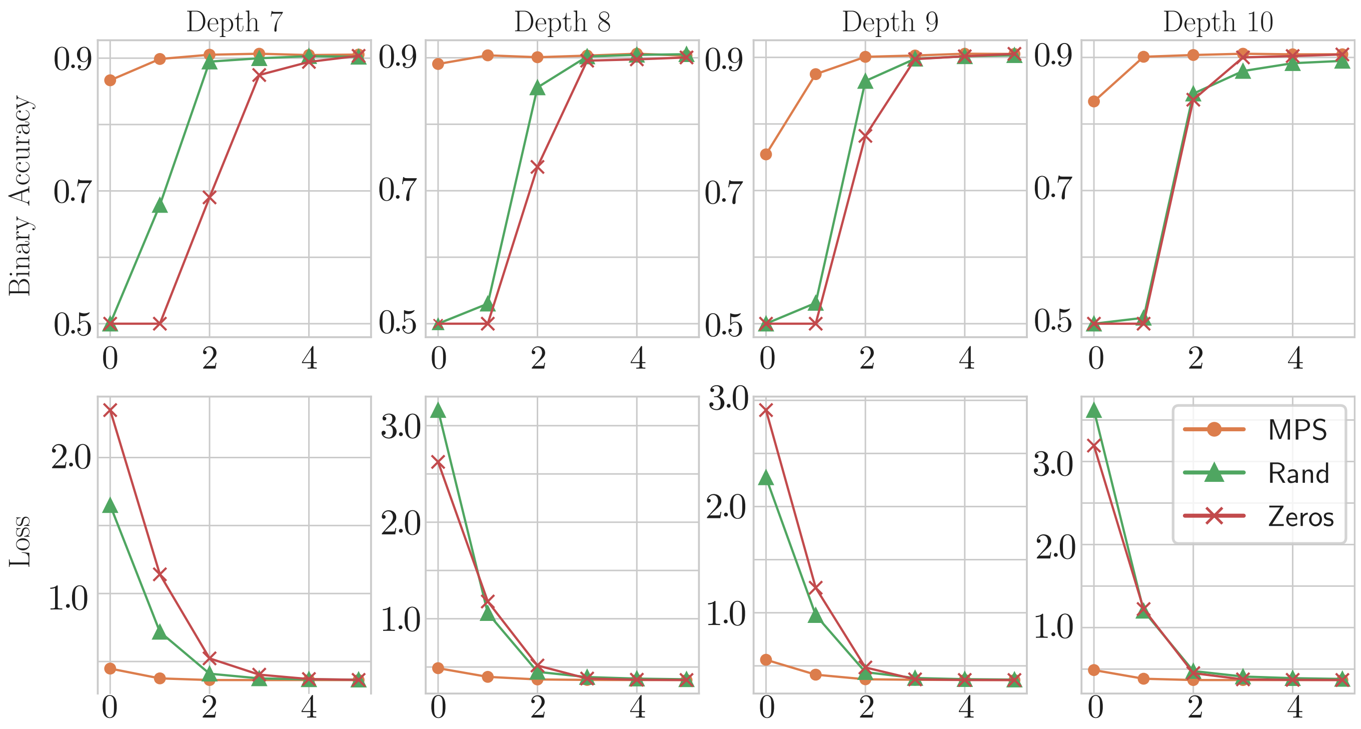


Figure 5.8: **Fashion MNIST Results:** *MPS Initialisation Results* - As well as starting at a higher binary accuracy, the accuracy of the MPS initialised circuits increased faster during the early epochs, indicating that the local loss landscape of the MPS initialised circuits may be easier to traverse for the optimizer. The *Adam* optimizer is used to update parameters in the circuit. The classical MPS were optimized for 5 epochs before being translated to a circuit and training continued. The MPS initialised circuits took no more than 2 epochs to converg==e, whereas all others required 4 or more epochs. As well as starting at a higher binary accuracy, the accuracy of the MPS initialised circuits increased faster during the early epochs, indicating that the local loss landscape of the MPS initialised circuits may be easier to traverse for the optimizer.

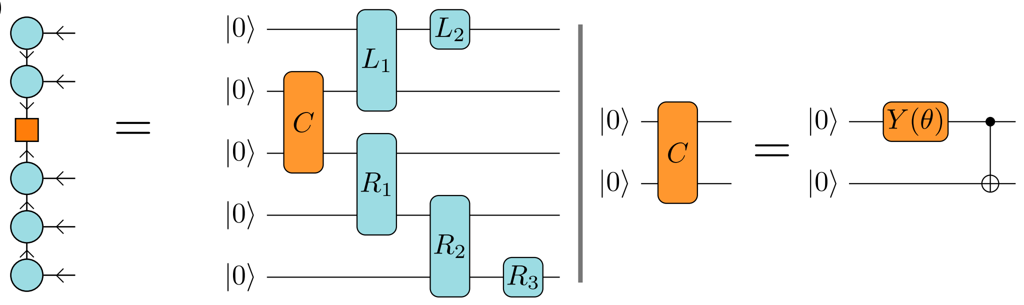
spirit to the pre-training method proposed here. The QAOA algorithm solves combinatorial optimisation problems, like the MaxCut problem. QAOA works by first preparing the ground state of a Hamiltonian that is easy to prepare, such as Pauli X acting on all sites. Then gates are applied, which act to interpolate between a Hamiltonian with an easy-to-prepare ground state and the target Hamiltonian, such as the MaxCut Hamiltonian. The adiabatic principle ensures that, if interpolated sufficiently slowly, the system will always remain in its ground state. At the end of the process, the QAOA algorithm will produce the ground state of the target Hamiltonian. Any product state would suffice as a good initial state, and the parent Hamiltonian would be able to be implemented to perform QAOA. [68] proposes to use classically generated solutions to combinatorial problems as the initial state. Relaxations of a discrete problem generate a continuous optimisation problem, solved via gradient descent or similar. Then after no interpolation at all, the state will already be as good an approximation to the answer as can be generated by the classical solver. Classical relaxation techniques are often very effective and hence provide a good initialisation for QAOA.

Layer-wise learning is another quantum circuit initialisation scheme. This technique proposes optimising only the first 1-2 layers of a quantum circuit and freezing all others to the identity. These low-depth circuits are less likely to suffer from barren plateaus and might provide an effective starting point for the rest of the circuit to train. To compare to MPS pre-training, note that a depth 1 or 2 brick wall circuit is strictly less expressive than bond dimension 2 MPS and can is equivalent to a finite correlation length bond dimension 2 MPS. It is natural to assume that a method which initialises a circuit from a general bond dimension 2 MPS will start from a better approximation than one initialised by a 1-2 layer pre-trained brick wall circuit.

Other Tensor Network Structures

Quantum circuits were initialised exclusively with MPS in left (or right) canonical form. More general tensor network structures can seed a quantum circuit, with circuits that no longer correspond to the diagonals of a brick wall circuit. Centre gauge, or mixed canonical, MPS are suitable circuit initialisers. The equivalent quantum circuit is less deep as several gates are applied in parallel. Mixed canonical MPS could be beneficial for quantum hardware with significant errors when qubits are idle, so minimising time between operations on a qubit is helpful to reduce the impact of noise.

a)



b)

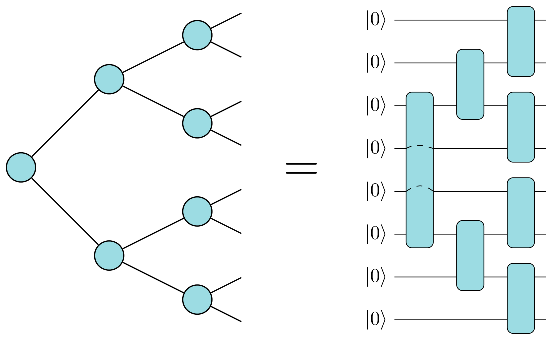


Figure 5.9: **Alternative Tensor Network Circuits:** a) *Mixed Canonical Form* - Mixed canonical form MPS can also be used to initialise a circuit, where the centre of orthogonality is parametrized using a Pauli Y rotation and a CNOT gate. This circuit is shallower than the equivalent left canonical circuit, and may help mitigate the effect of noise. b) *Tree Tensor Network* - Tree tensor networks can be exactly expressed as quantum circuits with longer range interactions. Tree tensor networks may be useful for pre-training on data which has longer range correlations.

Similarly, more exotic tensor network structures can be trained classically and embedded as a quantum circuit. Tree tensor networks, Fig 5.9b, may be more effective classifiers for problems which benefit from forming hierarchical representations at different scales, as is the case with image processing.

5.2.6 Summary

This chapter introduced MPS pre-training, a novel initialisation scheme for quantum neural networks. MPS pre-training involves classically optimising an MPS to solve some machine learning task such as finding the ground state of a Hamiltonian, discrete optimisation problems, or image classification. The optimised MPS can be translated without loss to a quantum circuit, and then compiled into a set of rotation angles. The circuit is initialised with these angles, guaranteeing that the circuit starts optimisation as good a solution to the target problem as the MPS. Optimisation proceeds from this starting point. The hope is that by starting at a good approximation to the target solution, the impacts of the barren plateau can be mitigated.

This method was tested with numerical evidence on a wide range of problems commonly used to test NISQ devices. Testing with the tilted field Ising model suggests that such an initialisation doesn't show vanishing gradients. Further tests optimising the transverse field Ising model and the H_2 electronic Hamiltonian demonstrate that the ground state energies can be identified in far fewer iterations, even when the MPS is poorly optimised. This suggests that this scheme may continue to work even for more highly entangled ground states, where low bond order MPS are no longer good solutions. This method demonstrated effectiveness in initialising quantum neural networks to solve combinatorial optimisation problems like MaxCut. MPS pre-training greatly reduced the number of iterations needed to find the ground state, as well as helping avoid local minima. Finally MPS pre-training is shown to be effective for initialisation of image classification models, as demonstrated on the Fashion MNIST dataset. The MPS is pre-trained using DMRG-inspired methods for optimising MPS as classifiers.

It remains to be seen under what scenarios MPS pre-training provides a significant benefit in a large-scale setting. By under-optimising the classical MPS, studying ground states at criticality, and studying scaling up to 24 qubits we hope to get a better understanding of how this method will scale to larger, more complex problems. One might suggest that because low bond-order MPS can only capture low entangled states that this method is unlikely to work for cases where the target state is highly entangled. There might exist low entangled states which are close to the highly entangled solutions in parameter-space, and those states are what is being targeted with this method. Larger scale experiments are needed, as well as experiments on NISQ devices to study the impact of noise on the initialisation quality.

Chapter 6

Conclusion

This thesis has explored connections between classical tensor networks and quantum computing. Many tensor network states and algorithms can be mapped onto quantum hardware, thereby benefiting from the increased expressive power of quantum devices. The contributions of each chapter are summarised below, and future directions for research are discussed.

Representing QMPS - This chapter introduced a novel way to simulate translationally invariant systems on quantum devices by mapping concepts from the classical simulation of infinite MPS to NISQ hardware. It's shown how to represent a translationally invariant MPS state on quantum hardware and use this representation to calculate the ground states of infinite translationally invariant systems. Moreover, such representations are easily extendible to capture large bond dimension states, albeit subsets of those states restricted to shallow depth circuits. Simulations of these optimisation algorithms demonstrate that shallow decompositions of quantum MPS faithfully capture ground state properties of low bond order MPS, and circuits are provided which may be able to extend to larger bond dimensions. The ability of shallow-depth quantum tensor network states to remain accurate as bond dimensions exceed classical simulation is an active research area. It is easy to write quantum MPS states which are difficult to simulate classically; however, the ability of these states to accurately represent a given physical system of interest is still unknown. Efforts towards answering this question, such as [12, 13] have been discussed, which identify the possibility of quantum advantage within the class of quantum tensor network states. This work introduces the *singular value* gate, which captures high bond dimension environments by matching the most significant n singular values of a $2^n \times 2^n$ environment tensor. Similar analysis needs to be done in the future to determine if the error scaling properties of this and other environment decompositions can achieve a quantum advantage. Circuits are outlined that are suitable for execution on hardware, allowing mid-circuit measurements, providing a way to run translationally invariant QMPS with fewer qubits at the cost of greater circuit depth.

This chapter demonstrated the ability to optimise shallow factorisations of translationally invariant bond dimension 2 MPS on NISQ hardware. Experiments on Google's Rainbow device achieve accurate estimates of ground state energy at the critical point of the TFIM by employing multiple error mitigation strategies used simultaneously.

Time Evolving QMPS - This work outlines a time evolution procedure for translationally invariant quantum MPS states based on the TDVP algorithm. A variational time evolution procedure is introduced in which quantum MPS states can be efficiently time evolved and then projected back onto a fixed bond dimension state which maximises the overlap with the time evolved state. The difficulty of measuring the overlap between two infinite states required a more involved procedure than is usually required for variational time evolution over finite states. The chapter proposes two solutions, one utilising the power method for estimating the overlap, the other requiring a nested optimisation loop to find the right and left eigenvalues of the transfer matrix. The latter effectively captured the dynamics of multiple systems, including scarred states in the PXP model and dynamical phase transitions in the TFIM model. Furthermore, it is demonstrated on the Google Rainbow device that the power method algorithm can potentially capture dynamical phase transitions in the TFIM model. Time evolution cost functions are identified that faithfully track the overlap during optimisations of a single time step. These experiments are the first steps toward the long-time evolution of critical systems on near-term hardware directly in the thermodynamic limit.

MPS Pre-Training - This chapter introduced a novel initialisation scheme for quantum neural networks, using classical MPS to seed a variational quantum algorithm. This classical pre-training mitigates the impacts of vanishing gradients, as well as removing the barren plateau up to 24 qubit simulations of the tilted field Ising model. This initialisation scheme is effective for quantum problems, such as finding the ground states of chemical and condensed matter Hamiltonians, and classical problems, like MaxCut optimisation and image classification on the Fashion MNIST dataset.

Future Directions - Extending the algorithms and techniques presented here to a higher bonder dimension is a promising avenue for future research. Higher bond dimension quantum MPS begin to accumulate errors originating from deeper circuits and shallow factorisations of tensors. Investigating the impact of these errors and methods to mitigate them could enable the simulation of tensor network states that are not classically simulatable. Ongoing works towards this include time evolution algorithms for the thermofield state introduced in Chapter 2, and the implementation of qubit-efficient time evolution algorithms to investigate the trade-off between circuit depth and accumulated noise in trapped ion qubit devices.

Bibliography

[1] Y. Cao, J. Romero, and A. Aspuru-Guzik. “Potential of quantum computing for drug discovery”. In: *IBM Journal of Research and Development* 62.6 (2018), 6:1–6:20. DOI: `10.1147/JRD.2018.2888987`.

[2] Isaac H. Kim et al. “Fault-tolerant resource estimate for quantum chemical simulations: Case study on Li-ion battery electrolyte molecules”. In: *Phys. Rev. Research* 4 (2 2022), p. 023019. DOI: `10.1103/PhysRevResearch.4.023019`. URL: `https://link.aps.org/doi/10.1103/PhysRevResearch.4.023019`.

[3] Fergus Barratt et al. “Parallel quantum simulation of large systems on small NISQ computers”. In: *npj Quantum Information* 7.1 (2021), pp. 1–7.

[4] James Dborin et al. “Simulating groundstate and dynamical quantum phase transitions on a superconducting quantum computer”. In: *Nature Communications* 13, 5977 (Oct. 2022), p. 5977. DOI: `10.1038/s41467-022-33737-4`.

[5] James Dborin et al. “Matrix product state pre-training for quantum machine learning”. In: *Quantum Science and Technology* 7.3 (2022), p. 035014.

[6] David Perez-Garcia et al. “Matrix Product State Representations”. In: *Quantum Information and Computation* 7 (July 2007), pp. 401–430. DOI: `10.26421/QIC7.5-6-1`.

[7] Guifré Vidal. “Class of quantum many-body states that can be efficiently simulated”. In: *Physical review letters* 101.11 (2008), p. 110501.

[8] Brian Swingle. “Entanglement renormalization and holography”. In: *Physical Review D* 86.6 (2012), p. 065007.

[9] Yudong Cao et al. “Quantum chemistry in the age of quantum computing”. In: *Chemical reviews* 119.19 (2019), pp. 10856–10915.

[10] Feng Pan and Pan Zhang. “Simulating the Sycamore quantum supremacy circuits”. In: *arXiv preprint arXiv:2103.03074* (2021).

[11] Frank Arute et al. “Quantum supremacy using a programmable superconducting processor”. In: *Nature* 574.7779 (2019), pp. 505–510.

[12] Sheng-Hsuan Lin et al. “Real- and Imaginary-Time Evolution with Compressed Quantum Circuits”. In: *PRX Quantum* 2 (1 2021), p. 010342. DOI: `10.1103/PRXQuantum.2.010342`. URL: `https://link.aps.org/doi/10.1103/PRXQuantum.2.010342`.

[13] Reza Haghshenas et al. “Variational power of quantum circuit tensor networks”. In: *Physical Review X* 12.1 (2022), p. 011047.

[14] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, 2010. DOI: `10.1017/CBO9780511976667`.

[15] Guang Hao Low and Isaac L Chuang. “Optimal Hamiltonian simulation by quantum signal processing”. In: *Physical review letters* 118.1 (2017), p. 010501.

[16] Peter W Shor. “Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer”. In: *SIAM review* 41.2 (1999), pp. 303–332.

[17] Lov K Grover. “A fast quantum mechanical algorithm for database search”. In: *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*. 1996, pp. 212–219.

[18] Dorit Aharonov et al. “Quantum walks on graphs”. In: *Proceedings of the thirty-third annual ACM symposium on Theory of computing*. 2001, pp. 50–59.

[19] András Gilyén et al. “Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics”. In: *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*. 2019, pp. 193–204.

[20] Eric Dennis et al. “Topological quantum memory”. In: *Journal of Mathematical Physics* 43.9 (2002), pp. 4452–4505.

[21] Craig Gidney and Martin Ekerå. “How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits”. In: *Quantum* 5 (2021), p. 433.

[22] John Preskill. “Quantum computing in the NISQ era and beyond”. In: *Quantum* 2 (2018), p. 79.

[23] Jerry Chow, Oliver Dial, and Jay Gambetta. “IBM Quantum breaks the 100-qubit processor barrier”. In: (2021).

[24] Yiqing Zhou, E. Miles Stoudenmire, and Xavier Waintal. “What Limits the Simulation of Quantum Computers?” In: *Phys. Rev. X* 10 (4 2020), p. 041038. DOI: `10.1103/PhysRevX.10.041038`. URL: `https://link.aps.org/doi/10.1103/PhysRevX.10.041038`.

[25] Jules Tilly et al. “The variational quantum eigensolver: a review of methods and best practices”. In: *arXiv preprint arXiv:2111.05176* (2021).

[26] Jarrod R McClean et al. “OpenFermion: the electronic structure package for quantum computers”. In: *Quantum Science and Technology* 5.3 (2020), p. 034014.

[27] Román Orús. “A practical introduction to tensor networks: Matrix product states and projected entangled pair states”. In: *Annals of physics* 349 (2014), pp. 117–158.

[28] Ulrich Schollwöck. “The density-matrix renormalization group in the age of matrix product states”. In: *Annals of Physics* 326.1 (2011). January 2011 Special Issue, pp. 96–192. ISSN: 0003-4916. DOI: `https://doi.org/10.1016/j.aop.2010.09.012`. URL: `https://www.sciencedirect.com/science/article/pii/S0003491610001752`.

[29] Jacob Biamonte and Ville Bergholm. “Tensor networks in a nutshell”. In: *arXiv preprint arXiv:1708.00006* (2017).

[30] Roger Penrose. “Applications of negative dimensional tensors”. In: *Combinatorial Mathematics and its Applications* (1971).

[31] Subir Sachdev. *Quantum phase transitions.* Cambridge university press, 2011.

[32] Subir Sachdev. “Quantum Criticality: Competing Ground States in Low Dimensions”. In: *Science* 288.5465 (2000), pp. 475–480. DOI: `10.1126/science.288.5465.475`. eprint: `https://www.science.org/doi/pdf/10.1126/science.288.5465.475`. URL: `https://www.science.org/doi/abs/10.1126/science.288.5465.475`.

[33] Piers Coleman and Andrew J Schofield. "Quantum criticality". In: *Nature* 433.7023 (2005), pp. 226–229.

[34] Adam Smith et al. "Crossing a topological phase transition with a quantum computer". In: *Phys. Rev. Research* 4 (2 2022), p. L022020. DOI: `10.1103/PhysRevResearch.4.L022020`. URL: `https://link.aps.org/doi/10.1103/PhysRevResearch.4.L022020`.

[35] B Pirvu, Frank Verstraete, and G Vidal. "Exploiting translational invariance in matrix product state simulations of spin chains with periodic boundary conditions". In: *Physical Review B* 83.12 (2011), p. 125104.

[36] John Cardy. *Scaling and Renormalization in Statistical Physics*. Cambridge Lecture Notes in Physics. Cambridge University Press, 1996. DOI: `10.1017/CBO9781316036440`.

[37] Juan Carlos Garcia-Escartin and Pedro Chamorro-Posada. "Swap test and Hong-Ou-Mandel effect are equivalent". In: *Physical Review A* 87.5 (2013), p. 052330.

[38] Xia Liu et al. "Mitigating barren plateaus of variational quantum eigensolvers". In: *arXiv preprint arXiv:2205.13539* (2022).

[39] C Neill et al. "Accurately computing the electronic properties of a quantum ring". In: *Nature* 594.7864 (2021), pp. 508–512.

[40] Yuchen Guo and Shuo Yang. "Quantum error mitigation via matrix product operators". In: *arXiv preprint arXiv:2201.00752* (2022).

[41] Frank Arute et al. "Observation of separated dynamics of charge and spin in the fermi-hubbard model". In: *arXiv preprint arXiv:2010.07965* (2020).

[42] Jin-Guo Liu et al. "Variational quantum eigensolver with fewer qubits". In: *Physical Review Research* 1.2 (2019), p. 023025.

[43] Joseph Vovrosh et al. "Simple mitigation of global depolarizing errors in quantum simulations". In: *Physical Review E* 104.3 (2021), p. 035309.

[44] Guifré Vidal. "Efficient Simulation of One-Dimensional Quantum Many-Body Systems". In: *Phys. Rev. Lett.* 93 (4 2004), p. 040502. DOI: `10.1103/PhysRevLett.93.040502`. URL: `https://link.aps.org/doi/10.1103/PhysRevLett.93.040502`.

[45] Jutho Haegeman et al. "Time-Dependent Variational Principle for Quantum Lattices". In: *Phys. Rev. Lett.* 107 (7 2011), p. 070601. DOI: 10.1103/PhysRevLett.107.070601. URL: https://link.aps.org/doi/10.1103/PhysRevLett.107.070601.

[46] Stefano Barison, Filippo Vicentini, and Giuseppe Carleo. "An efficient quantum algorithm for the time evolution of parameterized circuits". In: *Quantum* 5 (2021), p. 512.

[47] M. Heyl, A. Polkovnikov, and S. Kehrein. "Dynamical Quantum Phase Transitions in the Transverse-Field Ising Model". In: *Phys. Rev. Lett.* 110 (13 2013), p. 135704. DOI: 10.1103/PhysRevLett.110.135704. URL: https://link.aps.org/doi/10.1103/PhysRevLett.110.135704.

[48] A. A. Michailidis et al. "Slow Quantum Thermalization and Many-Body Revivals from Mixed Phase Space". In: *Phys. Rev. X* 10 (1 2020), p. 011055. DOI: 10.1103/PhysRevX.10.011055. URL: https://link.aps.org/doi/10.1103/PhysRevX.10.011055.

[49] Andrew Hallam, JG Morley, and Andrew G Green. "The Lyapunov spectra of quantum thermalisation". In: *Nature communications* 10.1 (2019), pp. 1–8.

[50] Juan Maldacena. "Eternal black holes in anti-de Sitter". In: *Journal of High Energy Physics* 2003.04 (2003), p. 021.

[51] Kishor Bharti et al. "Noisy intermediate-scale quantum (NISQ) algorithms". In: *arXiv preprint arXiv:2101.08448* (2021).

[52] Marco Cerezo et al. "Variational quantum algorithms". In: *Nature Reviews Physics* 3.9 (2021), pp. 625–644.

[53] Jules Tilly et al. "The variational quantum eigensolver: a review of methods and best practices". In: *arXiv preprint arXiv:2111.05176* (2021).

[54] Jarrod R McClean et al. "Barren plateaus in quantum neural network training landscapes". In: *Nature communications* 9.1 (2018), pp. 1–6.

[55] Zoë Holmes et al. "Connecting ansatz expressibility to gradient magnitudes and barren plateaus". In: *PRX Quantum* 3.1 (2022), p. 010313.

[56] Carlos Ortiz Marrero, Mária Kieferová, and Nathan Wiebe. "Entanglement-induced barren plateaus". In: *PRX Quantum* 2.4 (2021), p. 040316.

[57] Samson Wang et al. “Noise-induced barren plateaus in variational quantum algorithms”. In: *Nature communications* 12.1 (2021), pp. 1–11.

[58] Edward Grant et al. “An initialization strategy for addressing barren plateaus in parametrized quantum circuits”. In: *Quantum* 3 (2019), p. 214.

[59] Mateusz Ostaszewski, Edward Grant, and Marcello Benedetti. “Structure optimization for parameterized quantum circuits”. In: *Quantum* 5 (2021), p. 391.

[60] Marco Cerezo et al. “Cost function dependent barren plateaus in shallow parametrized quantum circuits”. In: *Nature communications* 12.1 (2021), pp. 1–12.

[61] Samuel Mugel et al. “Dynamic portfolio optimization with real datasets using quantum processors and quantum-inspired tensor networks”. In: *Physical Review Research* 4.1 (2022), p. 013006.

[62] Edwin Stoudenmire and David J Schwab. “Supervised learning with tensor networks”. In: *Advances in Neural Information Processing Systems* 29 (2016).

[63] Jinhui Wang et al. “Anomaly detection with tensor networks”. In: *arXiv preprint arXiv:2006.02516* (2020).

[64] Alexander Novikov, Mikhail Trofimov, and Ivan Oseledets. “Exponential machines”. In: *arXiv preprint arXiv:1605.03795* (2016).

[65] Fergus Barratt, James Dborin, and Lewis Wright. “Improvements to Gradient Descent Methods for Quantum Tensor Network Machine Learning”. In: *arXiv preprint arXiv:2203.03366* (2022).

[66] Robert R Tucci. “An introduction to Cartan’s KAK decomposition for QC programmers”. In: *arXiv preprint quant-ph/0507171* (2005).

[67] Yunchao Liu, Srinivasan Arunachalam, and Kristan Temme. “A rigorous and robust quantum speed-up in supervised machine learning”. In: *Nature Physics* 17.9 (2021), pp. 1013–1017.

[68] Daniel J Egger, Jakub Mareček, and Stefan Woerner. “Warm-starting quantum optimization”. In: *Quantum* 5 (2021), p. 479.