A VARIATIONAL QUANTUM ALGORITHM FOR SOLVING PARTIAL DIFFERENTIAL EQUATIONS

A Thesis

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Master of Science in Applied and Engineering Physics

by

Yilian Liu

May 2023

ABSTRACT

Variational Quantum Algorithms(VQAs) are a class of algorithm that uses a hybrid approach to solve optimization problems. VQAs use a quantum computer to compute expectation values of circuits that are encoded using classical data and then use a classical optimizer to adjust the parameters of these quantum circuits in order to find the optimal solution. The cost function is defined for any specific problem such that the set of parameters that yields the lowest cost function value corresponds to the solution of that problem. This method shows great potential in Near-term Intermediate Scale Quantum (NISQ) devices. This thesis investigates two approaches of constructing cost functions: the distance minimization approach and the energy minimization approach. Periodic, Dirichlet, and Neumann boundary conditions are studied. We present results for simulation of the heat equation solver, the Poisson equation solver, and the Navier-Stokes equation solver. We also present two ansatz designs, the ZGR-QFT ansatz and the Universal Layered Ansatz, as well as optimization strategies.

BIOGRAPHICAL SKETCH

Yilian Liu was born on May 30, 1999, in Nantong, China. He received a Bachelor of Arts in Physics from Reed College in Portland, Oregon in 2021 where he worked with Professor John Essick on optically detected magnetic resonance in diamond Nitrogen-Vacancy centers for his undergraduate thesis. While at Reed, he also found an interest in quantum algorithms and started working with Professor Peter McMahon on variational quantum algorithms which finally resulted in this thesis. He will be joining Applied Materials as a software engineer in June 2023, and he wishes to continue working on machine learning for quantum systems and quantum algorithms.

This document is dedicated to Lirui.

ACKNOWLEDGEMENTS

I am honored to have the opportunity to express my gratitude and acknowledge those who have supported me throughout my thesis journey.

First and foremost, I extend my deepest appreciation to my advisor, Professor Peter McMahon. He has been an invaluable mentor, who not only taught me the fundamentals of quantum algorithm research but also provided me with the guidance and resources necessary to complete this project. Without his encouragement and support, this thesis would not have been possible.

I would also like to extend my sincere thanks to my teammate Thomas Watts. His dedication and hard work, along with his contagious enthusiasm, helped me through countless late nights of debugging and algorithmic development. I am grateful for his constant support and encouragement.

Additionally, I would like to thank Abhi Sarma and all the other members of our research group, whose valuable insights and contributions helped me to refine my ideas and improve the quality of my work.

I would like to express my gratitude to my other advisor, Professor Karan Mehta, for his support, guidance, and opportunities in experimental research. And to all the members in the PQE group, thank you all for being so supportive during the last year of my program!

I am also deeply grateful to Hanrui Wang for inspiring me to explore machine learning for quantum systems and for providing me with valuable guidance and feedback. Many ideas in this thesis were inspired by our collaboration.

I want to thank my parents for their firm support and encouragement throughout my academic journey. Their love and support have been an endless source of strength and motivation.

Finally, I would like to thank Lirui, my love, for her constant love, support,

and encouragement. Our adventures and journeys together through the years are the most precious things I hold and I cannot wait to make many more memories with her in the years to come! I would not be where I am without her. Thank you, Lirui, for always being there for me.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Partial differential equations (PDEs) are an essential tool for understanding and modeling complex physical systems across a wide range of fields, including fluid dynamics, solid mechanics, electromagnetics, and quantum mechanics. These equations are used to describe the behavior of functions that depend on multiple independent variables, such as space and time. The study of PDEs has been instrumental in the advancement of scientific and engineering knowledge, and their solutions often provide valuable insights into the fundamental properties of the underlying systems.

Traditionally, numerical solutions of PDEs have been obtained by discretizing the function using finite difference methods, where the function is approximated at discrete points in the domain. The resulting system of equations is then solved using iterative algorithms, typically involving the inversion of a large matrix. However, the numerical solution of nonlinear PDEs can be particularly challenging due to the complex and chaotic nature of the underlying systems. In such cases, the finite difference method may require an extremely large number of grid points to accurately capture the dynamics of the system, which can be computationally expensive and time-consuming. Moreover, if the resolution of the grid is too low, valuable information about the system could be lost, leading to inaccurate or incorrect results.

To overcome these challenges, various advanced numerical methods have been developed for solving nonlinear PDEs, including finite element methods[3], spectral methods[14], and meshless methods[2]. These techniques offer improved accuracy, stability, and efficiency over traditional methods, and

are capable of handling complex geometries and boundary conditions. In this thesis, we will explore a new approach for solving these challenges using a quantum computer.

Quantum computing is a rapidly evolving field that promises to revolutionize computation by harnessing the principles of quantum mechanics to perform calculations in a fundamentally different way than classical computers. At the heart of quantum computing is the concept of qubits, which can exist in a superposition of states that correspond to the classical bits 0 and 1. When multiple qubits are combined in a quantum register, they allow for the representation of a vast number of basis states in a high-dimensional space.

In particular, an n-qubit quantum computer spans a 2^n dimensional space, and the amplitudes of the basis states can be used to encode information of a discretized function. This approach offers significant advantages over classical computing, as the number of data points grows exponentially with the number of qubits, allowing for a level of resolution intractable by classical computers.

This feature generated great interest in variational quantum algorithms (VQA) on current Near-term Intermediate Scale Quantum (NISQ) devices with tens of qubits. VQAs work by iteratively adjusting the parameters of a quantum circuit in order to find a circuit that performs well on a pre-defined cost function. This approach has been shown to be effective for a variety of problems, including machine learning, optimization, and chemistry. In theory, we need only to construct cost functions for specific problems to solve these problems on a quantum computer using VQA. In this thesis, we identify five crucial problems when fully implement VQAs to solve PDEs: (1) State preparation for initial conditions, (2) Ansatz design, (3) Implementation of boundary conditions,

(4) Optimization, and (5) Read-Out.

This thesis starts by investigating the variational quantum algorithm proposed by Lubasch et al., which uses a distance minimization approach. A modified approach using the energy minimization method will also be explored. We will discuss the theory behind variational quantum algorithm, circuit designs for implementing the algorithm on a quantum computer, Ansatz designs, optimization techniques, and results for numerical simulation.

CHAPTER 2

THEORY

2.1 Discretization

Consider a function $u(x,t)$ on the $[0,1]$ domain. We evenly discretize the domain into $N+1$ points with distance between each neighboring pair of points as $\Delta x = \frac{1}{N}$. The function $u(x,t)$ is then turned into a vector $U(t) = (u_0^t, u_1^t, ..., u_{N-1}^t) \in \mathbf{R}^N$ where each entry is the value of the function evaluated at the corresponding point, i.e.

$$u_i^t = u(i/N, t). \tag{2.1}$$

We will work with the discretized vector $U(t)$ instead of the function.

2.1.1 Finite Difference Method

A general 1D differential equation that's first order in time will have the form,

$$\frac{\partial u}{\partial t} = \hat{O}u, \tag{2.2}$$

where $\hat{O}$ is a linear combination of spatial derivative operators. The discretized first order spatial derivative using the forward difference method is given by

$$\frac{\partial u_i^t}{\partial x} = \frac{u_{i+1}^t - u_{i-1}^t}{\Delta x} + O(\Delta x). \tag{2.3}$$

The second order spatial derivative from the central difference method is given by

$$\frac{\partial^2 u_i^t}{\partial x^2} = \frac{u_{i+1}^t - 2u_i^t + u_{i-1}^t}{\Delta x^2} + O(\Delta x^2). \tag{2.4}$$

Higher order derivatives can be constructed similarly.

With the finite difference derivatives, we can construct the matrix O as the finite difference version of operator $\hat{O}$ and use a classical variational method to solve the differential equation for some initial condition $U(t_0)$.

2.2 The Quantum Approach

To harness the power of quantum computation, we will first convert the vector $U(t)$ into a quantum state through amplitude encoding,

$$|U(t)\rangle = \sum_{i}^{N-1} C_i^t |i\rangle \,, \tag{2.5}$$

where $|i\rangle$ is the computational basis state, such that $\lambda C_i^t = u_i^t$ for some $\lambda = \sum_i^{N-1} (u_i^t)^2$. The normalization constant λ is necessary because the quantum state has to have norm 1. An n-qubit system will have 2^n computational basis states, which can store vectors of length 2^n entries.

2.2.1 Derivative Operator

Based on the finite difference relation given by Eq. 2.3 with periodic boundary condition, the first order spatial derivative operator in matrix form is

$$\hat{\nabla} = \begin{pmatrix} 0 & 1 & 0 & 0 & & \dots & 0 & -1 \\ -1 & 0 & 1 & 0 & & \dots & & 0 \\ 0 & -1 & 0 & 1 & & \dots & & 0 \\ \vdots & & & \ddots & & & \vdots & \\ 1 & 0 & 0 & 0 & \dots & & -1 & 0 \end{pmatrix}. \tag{2.6}$$

We introduce the adder operator $\hat{A}$ which shifts the amplitude of a quantum state, and the adder dagger operator. More concretely the two operators are defined in Dirac notation as

$$\hat{A} = \sum_{i}^{N-1} |i+1\rangle \langle i| , \tag{2.7}$$

and

$$\hat{A}^{\dagger} = \sum_{i}^{N-1} |i-1\rangle \langle i| . \tag{2.8}$$

Observe that applied on a quantum state, the adder gives $\hat{A}\,|U(t)\rangle = \sum_i^N C_i^t\,|i+1\rangle$. And similarly the adder dagger gives $\hat{A}^{\dagger}\,|U(t)\rangle = \sum_i^{N-1} C_i^t\,|i-1\rangle$.

Using these two operators, we can write out the first spatial derivative operator as a quantum operator defined as

$$\hat{\nabla} = \hat{A} - \hat{A}^{\dagger}. \tag{2.9}$$

2.2.2 Second Order Derivative Operator

The second order derivative operator, $\hat{\Delta}$, based on Eq. 2.4 with periodic boundary condition in matrix form is given as,

$$\hat{\Delta} = \begin{pmatrix} -2 & 1 & 0 & 0 & \dots & 0 & 1 \\ 1 & -2 & 1 & 0 & \dots & & 0 \\ 0 & 1 & -2 & 1 & \dots & & 0 \\ \vdots & & & \ddots & & \vdots & \\ 1 & 0 & 0 & 0 & \dots & 1 & -2 \end{pmatrix}. \tag{2.10}$$

Observe that we can construct this matrix with

$$\hat{\Delta}^2 = -2I + \hat{A} + \hat{A}^{\dagger}. \tag{2.11}$$

An alternative approach for constructing the second derivative operator is given by Sato et al[12].

$$\hat{\Delta}^2 = I^{\otimes(n-1)} \otimes (I - X) + \hat{A}^\dagger (I^{\otimes(n-1)} \otimes (I - X))\hat{A} \tag{2.12}$$

2.2.3 Adder

The derivative operators can be constructed using the adder operator $\hat{A}$ which shifts the amplitude of a quantum state. More concretely it is defined as

$$\hat{A} = \sum_i^N |i+1\rangle \langle i| . \tag{2.13}$$

Observe that applied on a quantum state, it gives $\hat{A}\,|U(t)\rangle = \sum_i^N C_i^t\,|i+1\rangle$. Similarly, the adder dagger operator is defined as

$$\hat{A}^\dagger = \sum_i^N |i-1\rangle \langle i| . \tag{2.14}$$

2.3 Time Discretization

With the tools introduced above, we can construct any arbitrary spatial derivative operators on the right hand side of Eq. 2.2. We will now formulate time derivatives on the left hand side of the equation. For time dependent equations, we can discretize time into discrete time steps. This means that for any time step $|u(t)\rangle$ we can define relation for $|u(t+\delta t)\rangle$, using Forward Euler time discretization

$$|u(t+\delta t)\rangle = (1 + \delta t\hat{O})\,|u(t)\rangle , \tag{2.15}$$

or using Backward Euler time discretization

$$(1 - \delta t\hat{O})\,|u(t+\delta t)\rangle = |u(t)\rangle . \tag{2.16}$$

In Section. 2.4, we will discuss how to compute $|u(t+\delta t)\rangle$ using these two relations. Assume for now that we know how $|u(t+\delta t)\rangle$ can be computed based on $|u(t)\rangle$, we can simply plug in $|u(t_0)\rangle$ as an initial condition and solve for $|u(t_0+\delta t)\rangle$. We would then use the solution $|u(t_0+\delta t)\rangle$ as the new initial condition and solve for $|u(t_0+2\delta t)\rangle$, and so on. This way, we can solve for the time evolution of the function of interest up to any time point $T = t_0 + N\delta t$ for N time steps.

2.4 Variational Method - Distance Minimization

To demonstrate the variational method with all the tools that we just introduced, we will use the Forwards Euler's method which gives us the relation

$$u(x, t+\Delta t) = (\mathbf{1} + \Delta t\hat{O}')u(x,t) \tag{2.17}$$

for the general 1D differential equation given in Eq. 2.2. In the quantum approach, we turn the function $u(x,t)$, $u(x,t+\Delta t)$ into quantum states $\tilde{\lambda}\,|\tilde{\psi}\rangle$ and $\lambda\,|\psi\rangle$, and obtain,

$$\lambda\,|\psi\rangle = (I + \Delta t\hat{O}')\tilde{\lambda}\,|\tilde{\psi}\rangle\,. \tag{2.18}$$

Note that since the identity operator is simply the zero-th order spatial derivative, we can combine it into $\hat{O}'$ and get $\hat{O} = (\mathbf{1} + \Delta t\hat{O}')$,

$$\lambda\,|\psi\rangle = \hat{O}\tilde{\lambda}\,|\tilde{\psi}\rangle\,. \tag{2.19}$$

2.4.1 Cost Function

The most naive approach to solving $|\psi\rangle$ would be minimizing the distance between the two vectors. A forward Euler cost function for time discretization can then be constructed using Eq. 2.19,

$$C_{forward}(\lambda, |\psi\rangle) = \left\| \lambda |\psi\rangle - \tilde{\lambda}\hat{O} |\tilde{\psi}\rangle \right\|^2 + O(\Delta t^2) \tag{2.20}$$

$$= \lambda^2 + \tilde{\lambda}^2 \left\langle \tilde{\psi} \middle| \hat{O}^\dagger \hat{O} \middle| \tilde{\psi} \right\rangle - 2\lambda\tilde{\lambda} Re\{\langle\psi|\hat{O}|\tilde{\psi}\rangle\} + O(\Delta t^2). \tag{2.21}$$

Note here that terms not dependent on $\tilde{\lambda}$ and $|\tilde{\psi}\rangle$ are constant in the cost function, so we can reduce the forward Euler cost function to

$$C_{forward}(\lambda, |\psi\rangle) = \lambda^2 - 2\lambda\tilde{\lambda} Re\{\langle\tilde{\psi}|\hat{O}|\psi\rangle\} + const. + O(\Delta t^2). \tag{2.22}$$

We can also use the backward Euler for time discretization which gives

$$\hat{O}\lambda |\psi\rangle = \tilde{\lambda} |\tilde{\psi}\rangle , \tag{2.23}$$

with $\hat{O} = 1 - \Delta t \hat{O}'$, and write the cost function as

$$C_{backward}(\lambda, |\psi\rangle) = \tilde{\lambda}^2 + \lambda^2 \langle\psi|\hat{O}^\dagger\hat{O}|\psi\rangle - 2\lambda\tilde{\lambda} Re\{\langle\tilde{\psi}|\hat{O}|\psi\rangle\} + O(\Delta t^2) \tag{2.24}$$

$$= \lambda^2 \langle\psi|\hat{O}^\dagger\hat{O}|\psi\rangle - 2\lambda\tilde{\lambda} Re\{\langle\tilde{\psi}|\hat{O}|\psi\rangle\} + const. + O(\Delta t^2). \tag{2.25}$$

We will represent $|\psi\rangle$ as a parametrized quantum circuit with variational parameters λ, and $|\tilde{\psi}\rangle$ as a state preparation circuit. The expectation value will be computed on a quantum computer. The choice of λ and λ that gives the lowest cost function value corresponds to the beset approximation of the time evolution solution for this differential equation. We can then use a classical optimizer to find the optimized parameters.

2.5 Energy Minimization

A different approach, inspired by Sato et al. [12], is to use the energy minimization method for solving PDEs. The method was introduced for solving the time-independent Poisson equation, but can be extended to time-dependent equations when we discretize time and solve for the time evolution step by step.

We will demonstrate the energy minimization method for the Poisson equation. First, consider the 1D Poisson's Equation for some function $f(x)$,

$$-\nabla^2 u(x) = f(x). \tag{2.26}$$

This equation corresponds to the total potential energy,

$$E = \frac{1}{2}\int \nabla v \cdot \nabla v d\Omega - \int v f d\Omega, \tag{2.27}$$

with discretized version defined as,

$$E_h = \frac{1}{2} v^\perp \hat{\Delta} v - v^\perp f. \tag{2.28}$$

Now, suppose we have some state quantum $|f\rangle$ that is the discretized version of function $f(x)$, minimizing the expectation of the total potential energy

$$E_h(|\psi\rangle) = -\frac{1}{2}\frac{(\langle f, \psi | X \otimes I^{\otimes n} | f, \psi\rangle)^2}{\langle\psi|\hat{\Delta}|\psi\rangle} \tag{2.29}$$

would give the state that corresponds to the solution of the Poisson's equation. Note that the state $|f, \psi\rangle$ is defined as $(|0\rangle|f\rangle + |1\rangle|\psi\rangle)/\sqrt{2}$. The term on the numerator in Eq. 2.29 gives the expectation value $\langle f|\psi\rangle$ via a swap test.

This energy minimization method can be generalized to other equations by finding the Lagrangian $\mathcal{L}[u]$ for the equation $u(x,t)$ that satisfies the Euler-Lagrange equation

$$\frac{\partial}{\partial x}\frac{\partial \mathcal{L}}{\partial u'} + \frac{\partial}{\partial t}\frac{\partial \mathcal{L}}{\partial \dot{u}} - \frac{\partial \mathcal{L}}{\partial u} = 0, \tag{2.30}$$

and minimizing the action $S[u]$ defined as,

$$S[u] = \int \mathcal{L}[u] \mathrm{d}x\mathrm{d}t \tag{2.31}$$

using the variational method.

For the Poisson equation, the energy minimization method reduces the number of expectation circuits by about a half as shown in Appendix. A, making the cost function considerably simpler to compute. And it is likely that for more complicated, time-dependent equations this advantage holds true as well. The challenge, however, is task of finding the corresponding Lagrangian equation for each equation.

In this thesis, we introduce the energy minimization cost function for the Poisson equation and Heat equation by find the corresponding Lagrangian through inspection. We point out, however, that finding Lagrangian for PDEs in general is not a trivial task. We have no more insight to offer within the scope of this work, so we direct your attention to [13], which proposes a way to formulate variational problems for any arbitrary nonlinear problem.

2.6 Boundary Conditions

Due to the periodic nature of the Adder operator, which we use to construct the derivative operators, the most straight forward way to construct Eq. 2.6 and Eq. 2.10 is with periodic boundary conditions. To extend our method to problems with other boundary conditions, we would need to apply correction terms to the cost function. In this section, we will discuss how these correction circuits can be constructed.

2.6.1 Dirichlet Boundary Condition

The Dirichlet boundary condition is given by

$$u(x) = f(x) \quad \forall x \in \partial\Omega, \tag{2.32}$$

where $f(x)$ is the boundary condition, and $\partial\Omega$ denotes the boundary of the function. For a 1D equation $u(x)$ defined on $[0, 1]$, the Dirichlet boundary condition is $u(0) = u_0$ and $u(1) = u_1$. The periodic second derivative operator is given by 2.10, while the Dirichlet second derivative operator is given by

$$\hat{\Delta}_{Dirichlet} = \begin{pmatrix} -2 & 1 & 0 & 0 & \dots & 0 & 0 \\ 1 & -2 & 1 & 0 & \dots & & 0 \\ 0 & 1 & -2 & 1 & \dots & & 0 \\ \vdots & & & \ddots & & \vdots & \\ 0 & 0 & 0 & 0 & \dots & 1 & -2 \end{pmatrix}. \tag{2.33}$$

Note that the two matrices only differ by the two 1's on the top right and bottom left corners.

The Simple Case

We will first consider the simple case of $\langle\psi|\hat{\Delta}_{periodic}|\psi\rangle$ which is largely used in the energy minimization method, for some state $|\psi\rangle = \sum_i^N a_i |i\rangle$. The difference between $\langle\psi|\hat{\Delta}_{periodic}|\psi\rangle$ and $\langle\psi|\hat{\Delta}_{Dirichlet}|\psi\rangle$ is $2a_0a_N$. In other words,

$$\langle\psi|A_{Dirichlet}|\psi\rangle = \langle\psi|A_{periodic}|\psi\rangle - 2a_0a_N. \tag{2.34}$$

So all we need to do is to find a circuit that produces $2a_0a_N$. Sato et al. proposed a simple way for exactly this problem. We apply a Hadamard to the least

significant qubit after applying the (periodic) Adder to get

$$(I^{\otimes n-1}\otimes H)A\left|\psi\right\rangle = \frac{a_N + a_0}{\sqrt{2}}\left|\psi_0\right\rangle + \frac{a_N - a_0}{\sqrt{2}}\left|\psi_1\right\rangle + ... + \frac{a_{N-1} + a_{N-2}}{\sqrt{2}}\left|\psi_{N-1}\right\rangle + \frac{a_{N-2} - a_{N-1}}{\sqrt{2}}\left|\psi_N\right\rangle. \tag{2.35}$$

Note that

$$\left(\frac{a_0 + a_1}{\sqrt{2}}\right)^2 - \left(\frac{a_0 - a_1}{\sqrt{2}}\right)^2 = 2a_0a_1 \tag{2.36}$$

which is exactly what we want. So then we just measure the probability $\tilde{a}_0^2 = \left(\frac{a_0+a_1}{\sqrt{2}}\right)^2$ of state $\left|\psi_0\right\rangle$, and probability $\tilde{a}_1^2 = \left(\frac{a_0-a_1}{\sqrt{2}}\right)^2$ of state $\left|\psi_1\right\rangle$ to get

$$\tilde{a}_0^2 - \tilde{a}_1^2 = 2a_0a_1. \tag{2.37}$$

The General Case

Now for the more general case of $\langle\phi|\hat{\Delta}_{Dirichlet}|\psi\rangle$ for some states $\left|\psi\right\rangle = \sum_i^N a_i\left|i\right\rangle$ and $\left|\phi\right\rangle = \sum_i^N b_i\left|i\right\rangle$, we have

$$\langle\phi|\hat{\Delta}_{Dirichlet}|\psi\rangle = \langle\phi|\hat{\Delta}_{periodic}|\psi\rangle - a_0b_N - a_Nb_0. \tag{2.38}$$

To find the term $(a_0b_N + a_Nb_0)$, we can either sample $\left|\psi\right\rangle$ and $\left|\phi\right\rangle$ individually to get the terms a_0, a_N, b_0, b_N; Or we could use the state $(\left|0\right\rangle\left|\psi\right\rangle + \left|1\right\rangle\left|\phi\right\rangle)/\sqrt{2}$. We will use the first qubit as control and apply $CNOT$s on all other qubits to get the state

$$\frac{1}{\sqrt{2}}\left|0\right\rangle\left|\psi\right\rangle + \frac{1}{\sqrt{2}}\left|1\right\rangle(b_N\left|0\right\rangle + b_{N-1}\left|1\right\rangle + b_{N-2}\left|2\right\rangle + ... + b_0\left|N\right\rangle) \tag{2.39}$$

Then we will apply a Hadamard on the first qubit to get

$$\frac{1}{\sqrt{2}}\left((a_0 + b_N)\left|0\right\rangle + ... + (a_N + b_0)\left|\frac{N-1}{2}\right\rangle + (a_0 - b_N)\left|\frac{N+1}{2}\right\rangle + ... + (a_N - b_0)\left|N\right\rangle\right) \tag{2.40}$$

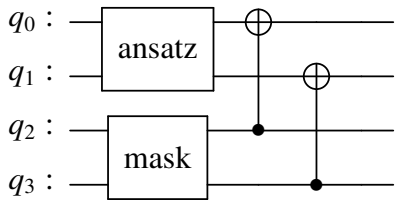
Now we just follow the steps described in the previous section and we can find the term $(a_0b_N + a_Nb_0)$.

2.6.2 Boundary Conditions in Higher Dimensions

For functions in more than one dimension, the problem becomes more complicated. We will focus on the 2D case in this section, but the method discussed here should generalize to higher dimensions.

A 2D function is encoded in an $N \times N$ grid for $N = 2^n$. The boundary condition is now applied to $4(N-1)$ points instead of 2 points in the 1D case. We can no longer apply the correction to each of the boundary point one by one, since the number boundary points is now dependent on the number of qubits. Otherwise the number of correction circuits will explode for large number of circuits.

Thus, for higher dimensions, we propose a general method of using a masking circuit of n ancilla qubits for n ansatz qubits. A simple example is shown in the circuit below.



The mask circuit turns the bottom two qubits into a binary state marking the amplitudes we want to pick out. The CNOT gates then act as point-wise multiplication to pick out only the amplitudes we marked from the ansatz. Note that this circuit only works when controlled by an ancilla qubit as in the case of the Hadamard test introduced in the next chapter. For higher dimensions, we can simply construct a mask that marks all the amplitudes that correspond to the boundary points and apply this circuit in the Hadamard test to find the

values we need to correct the cost function by.

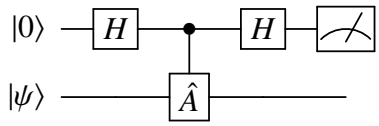
CHAPTER 3

CIRCUIT DESIGN

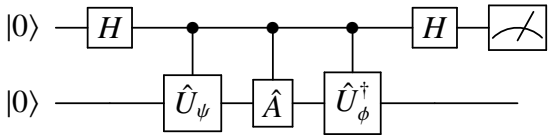
In this chapter we will discuss the circuit designs that make the algorithm possible.

3.1 Hadamard Test

First, we will introduce the Hadamard test. The Hadamard test computes the expectation value of an operator $\hat{A}$ with respect to some state $|\psi\rangle$, by sampling an ancilla qubit. It will give the value $Re(\langle\psi|\hat{A}|\psi\rangle)$. The circuit is given by



A more general Hadamard test can be used to calculate the expectation value $Re(\langle\phi|\hat{A}|\psi\rangle)$. The circuit is given by

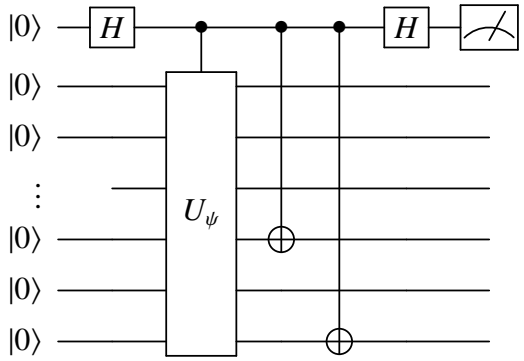


where $\hat{U}_\psi|0\rangle = |\psi\rangle$ and $\hat{U}_\phi|0\rangle = |\phi\rangle$.

3.1.1 Access of Individual Amplitude

For some state $\hat{U}_\psi|0\rangle = |\psi\rangle$, we can also use the Hadamard test to find the amplitudes of individual basis states. We first find the binary string representing the basis state. For example, the sixth state for n-qubits is the state $|0...0101\rangle$. To

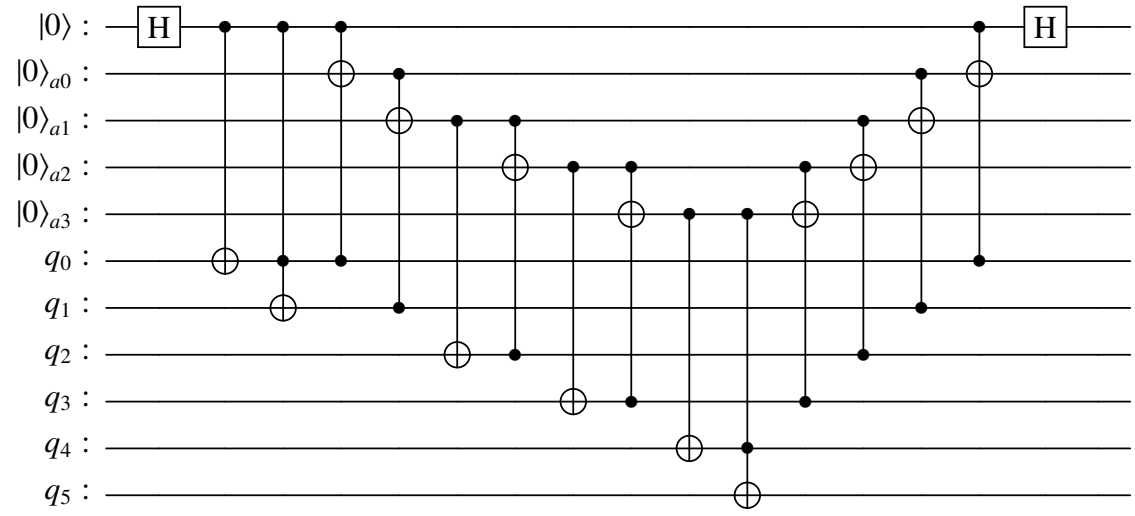
access the amplitude of that state we apply CNOT gates on the last and third to last qubits with the ancilla qubit as control,



The measurement will return the real part of the amplitude of state $|0...0101\rangle$.

3.2 Adder

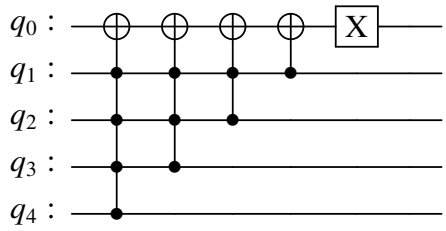
There are a few different ways of constructing the adder operator. The design proposed by Lubasch et al. [6] for 6 ansatz qubits, is given as



where the first qubit is the ancilla qubit for the Hadamard test, the qubits with subscript a are the ancilla qubits for the adder. This design in general uses $n - 2$

ancilla qubits and has $O(n)$ depth when decomposed to two qubit gates.

The design proposed by Sato et al[12] requires no ancilla qubits, but requires more multi-controlled CNOT gates, as shown in the circuit below. This design, when decomposed into two-qubit gates, will be $O(2^n)$ in depth.



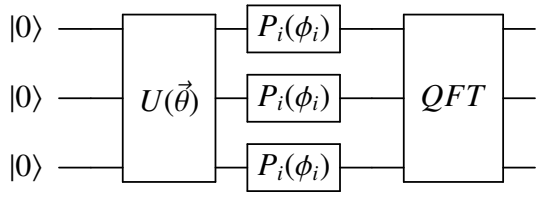
In Section 3.4, we will introduce the ZGR-QFT ansatz. This ansatz approximates the solution using a truncated Fourier series. It encodes the fourier coefficients in the qubits first, then a QFT is applied to bring the state from Fourier space to real space. Thus, we can use the fact that in Fourier space, the adder can be defined as

$$\hat{A} = \bigotimes_{i=0}^{n-1} P_i(\phi_i) \tag{3.1}$$

where

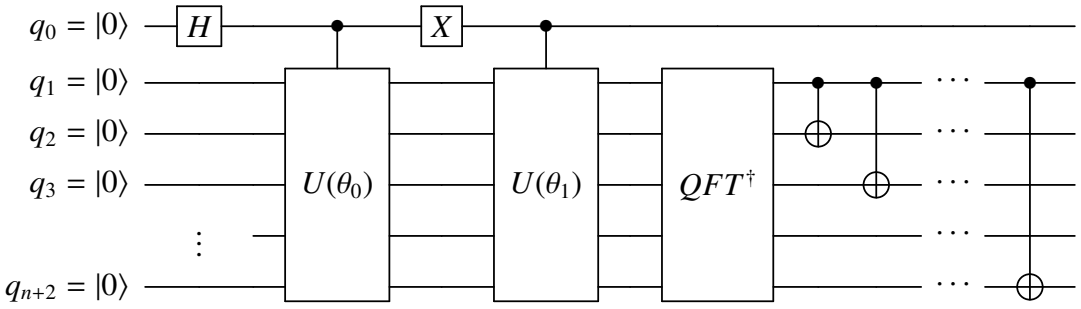
$$\phi_i = -\frac{2\pi}{N} 2^{n-1-i}. \tag{3.2}$$

We simply need to apply single qubit phase shift gates to each of the m qubits with Fourier coefficients encoded in between the *ZGR* and *QFT* circuit[8], as shown below.



3.2.1 Swap Test

As mentioned previously, we need to perform a swap test to compute the expectation value of $\langle f|u\rangle$. With the use of ZGR-QFT ansatz and the trick for representing non-periodic functions, we come up with the following circuit:



Our goal here is to prepare a state $(|0\rangle|\psi\rangle + |1\rangle|\phi\rangle)/\sqrt{2}$. Let CU be the controlled ZGR ansatz such that $QFT^\dagger(CU(\theta_0)|1\rangle|0\rangle_{n+1}) = |1\rangle|\tilde{\psi}\rangle_{n+1}$, where the refelcted state $|\tilde{\psi}\rangle_{n+1} = (|0\rangle|\psi(x)\rangle_0 + |1\rangle|\psi(1-x)\rangle)/\sqrt{2}$ and similarly $QFT^\dagger(CU(\theta_1)|1\rangle|0\rangle_{n+1}) = |1\rangle|\tilde{\phi}\rangle_{n+1}$. The circuit here uses 2 ancilla qubits, which is $n+2$ qubits to prepare 2 non-periodic n-qubit states.

3.3 Non-linearity

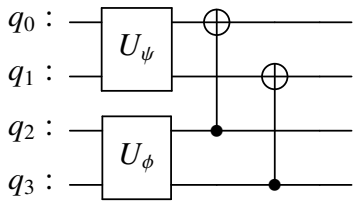
In this section, we will discuss how to introduce non-linearity to the cost functions. Consider a trivial nonlinear differential equation

$$\frac{\partial}{\partial x}\mathbf{u} = \mathbf{u}^2. \tag{3.3}$$

For some normalized quantum state $|u\rangle = \sum_i^N C_i|i\rangle$, we can write

$$|u^2\rangle = \sum_i^N C_i^2|i\rangle\,. \tag{3.4}$$

However, $|u^2\rangle$ is no longer a normalized state, which means there is no unitary operator that produces $|u^2\rangle$ from $|u\rangle$. The general solution of the problem requires us to find some way of performing point-wise multiplication in the computational basis for two arbitrary quantum states $|\psi\rangle$ and $|\phi\rangle$. Thus, we introduce the diagonal operator, implemented in the circuit below,



where $U_\psi |0\rangle = |\psi\rangle$ and $U_\phi |0\rangle = |\phi\rangle$.

3.4 Ansatz Design

Here we will discuss how to encode an arbitrary function $u(x)$ to a quantum state $\lambda |\psi\rangle$. The circuit that gives us arbitrary control over the amplitude of each state is called an ansatz circuit. A dilemma in designing an ansatz for a variational quantum algorithm lies in the very nature of the algorithm. One major advantage of using a quantum computer is in the number of grid points encoded in the quantum state which grows exponentially with the number of qubits. The type of precision intractable by classical computers can be achieved by a quantum computer with relatively small number of qubits. To perfectly access all the possible quantum states of a quantum computer however, we need $O(2^n)$ parameters for n-qubit machines, which defeats the advantage in data compression since the $O(2^n)$ parameters are stored classically. Furthermore, this would require exponentially deep circuits. However, if we were to use fewer than $O(2^n)$

parameters, we only have access to some subspace of the 2^n-dimensional Hilbert space of the quantum computer and we risk not being able to reach the state that corresponds to the solution.

3.4.1 ZGR-QFT Ansatz

One robust ansatz we will be using is the Zalka-Grover-Rudolph[9] Quantum Fourier Transform (ZGR-QFT) ansatz developed by Thomas Watts and Mudassir Moosa. This ansatz uses a combination of the ZGR ansatz and the Quantum Fourier Transform (QFT) circuit.

The ZGR circuit itself is used to generate arbitrary quantum states on n-qubits using $O(2^n)$ parameters. In the first part of the ZGR-QFT ansatz, we apply the ZGR circuit to on m qubits for some $m < n$ to encode the first 2^m fourier coefficients of the target function. After that, the QFT is applied on all n qubits, which turns the entire state into a truncated Fourier series of the target function with 2^m. This way, we only need a fix number of m qubits as the number of total qubits grow, for 2^m fourier terms. The number of parameters is thus $O(2^m)$.

While the ZGR-QFT ansatz performs state preparation with very high fidelity, limited only by the truncation error of the Fourier series, and the depth is only $O(2^m) + O(n^2)$ where the second term comes from the depth of perform QFT on n-qubits, it should be noted that the ansatz is classically simulatable. Since all the information is encoded in the 2^m fourier coefficients, a solution would just be equivalent to solving the coefficients individually, which is classically efficient using the spectral method[1]. In other words, increasing number of qubits will not increase the effective resolution of our solution. Thus, the ZGR-QFT ansatz

should be used for loading initial conditions, but not as ansatz for solving PDEs.

3.4.2 Gibbs Phenomenon

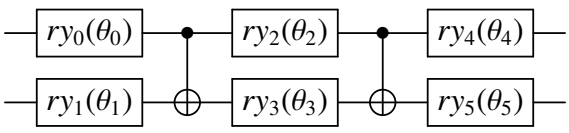
Due to the nature of the Fourier Transform, the ZGR-QFT ansatz is very good at representing periodic states, but the Gibbs Phenomenon will be present if the state is non-periodic. To solve this problem, we will first encode a periodic state for function $f(x)$ as

$$F(x) = \begin{cases} f(x) & 0 < x \leq 1 \\ f(1-x) & 1 < x \leq 2 \end{cases} \tag{3.5}$$

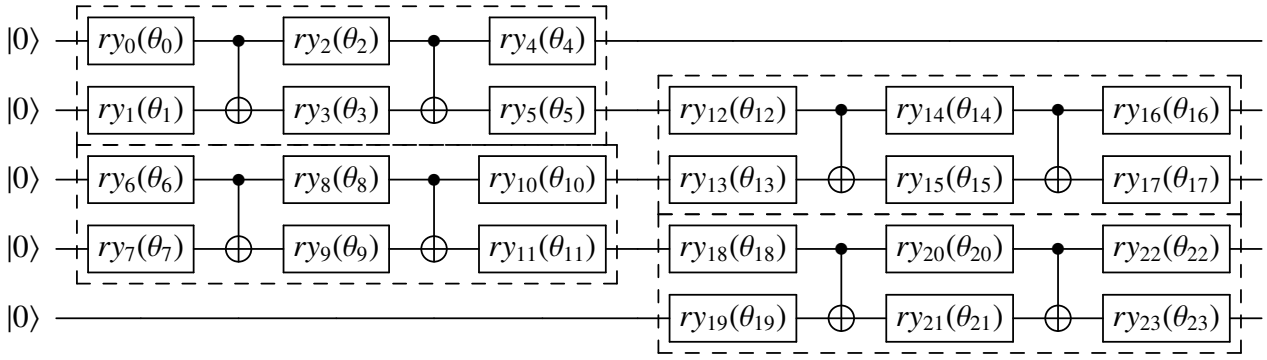
in $n+1$ qubits, which will give the state $|F(x)\rangle = (|0\rangle|f(x)\rangle + |1\rangle|f(1-x)\rangle)/\sqrt{2}$. We then disentangle the $|1\rangle$ state of the first qubit with the rest of the qubits using a series of $CNOT$ gates. We are left with $|0\rangle|f(x)\rangle$ with little error. It is now possible to exclude the first qubit, since it is not entangled with the rest of the circuit.

3.4.3 Universal Layered Ansatz

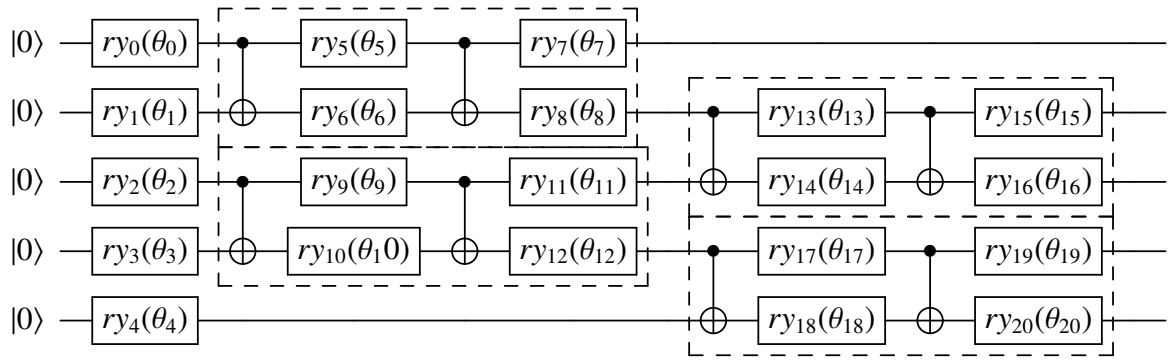
The Universal Layered Ansatz (ULA) is a robust ansatz we propose that is not classically simulatable. First, consider that for two qubits, one way to construct any arbitrary unitary gate for real valued states is to use circuit proposed by Parrish et al. [11]



So ULA applies alternating layers of the $SO(4)$ gate on all the qubits, giving the following circuit for 5 qubits with two layers,



where each block is an $SO(4)$. The number of layers can be adjusted according as needed during optimization. As shown in the circuit diagram above, we note that gates ry_5 and ry_{12} can be combine to a single ry which would remove one parameter. Similarly for the two sets of ry's below. Thus, we will use the following equivalent circuit for the purpose of optimization with fewer parameters.



Notice that the ULA is a Hardware-Efficient Ansatz, since it only requires two neighboring connections for each qubit which is a major advantage of this ansatz. Using the ULA, we would need $O(2n)$ number of parameters to be able to generate arbitrary real valued ansatz states. However, the hope lies in the fact that in optimizing the cost function, we need the ansatz to be able to reach a specific solution state instead of all arbitrary states. This means that if we can

identify and remove the redundancies in the ULA for a specific solution state, we can have a more sparse ansatz that is just enough.

3.5 Optimization Strategy

3.5.1 Genetic Algorithm-based Optimization

As mentioned in the previous chapter, the ULA, consisting of only RYs and CNOTs, should be able to reach any arbitrary real valued state. However, the previous effort in using the ULA shows that it is very difficult to optimize and runs into barren plateaus easily. If we simply add more layers in hope of a better result, the optimization will become exponentially harder to perform and we might be overparametrizing the circuit[5].

Therefore, we propose the genetic universal layered ansatz (gULA), which aims to solve this problem by parametrizing the gate configuration on top of the RY gate parameters. We call this kind of optimization: **Ansatz Structure Optimization**. The gate configuration is represented by a bit-string where '1' corresponds to an RY gate and '0' corresponds to an identity (in place of an RY). The configuration will be referred to as a *gene* in reference to the genetic algorithm.

There are now two layers of optimization. First, we generate a batch of genes with random bit-strings. We optimize over each gene to get the best cost and parameters. The cost is used to determine how likely a gene will be selected for the next round. We can also set hyper-parameters in the genetic algorithm for

randomly mutating a gene. We then optimize the parameters in the structure to find the new cost. Thus, we can find the optimal structure of ULA.

3.5.2 Modified Genetic Algorithm Optimization

In the previous optimization design, it should be noted that two layers of optimization will require $O(2^n)$ number of optimizations, significantly slowing down the speed of the algorithm. However, according to [15], the performance of a sub-circuit using parameters from an optimized larger circuit is correlated to the performance of an optimized sub-circuit. Thus, we propose a modified optimization scheme where a much larger circuit is trained first to obtain a set of parameters. Now the larger circuit likely won't be ablt to reach the solution state due to the presence of barren plateaus[7]. However, we can apply the parameters to sub-circuits of the larger circuit with some of the rotations removed. We identify the sub-circuit with the lowest cost using the same set of parameters and optimize the best performing sub-circuit.

3.5.3 Gradient-Free Optimization

Gradient-free optimizers have also been proposed for parameterized quantum circuit optimization such as Rotosolve[10]. These optimizers work by finding the parameter that produces the local minimum of each rotation gate while holding all the other parameters fixed. These optmizers gradient free optimizers can potentially work very well with the ULA since we can find the optimized parameters for each gate iteratively, and simply remove the gate with parameter

values under a certain threshold.

CHAPTER 4

POISSON EQUATION

4.1 Energy Minimization Poisson Equation Cost Function

Sato et al[12] introduced an energy minimization based cost function for solving the linear Poisson equation of the form

$$-\nabla^2 u(\mathbf{x}) = f(\mathbf{x}) \tag{4.1}$$

where $f(\mathbf{x})$ is some density function given as a source function. It can be shown (see Eq. 6 of [12]) that the critical points of the energy functional $E(v)$ are the solutions of the Poisson equation where

$$E[u] = \frac{1}{2}\int_\Omega \Delta v \cdot \Delta v \mathrm{d}\Omega - \int_\Omega v f \mathrm{d}\Omega. \tag{4.2}$$

Let $|f\rangle$ be the discretized source function $f(\mathbf{x})$ and assume $|f\rangle$ has squared norm 1. We can then let $|v\rangle = r|\psi\rangle$ be the discretized vector of the solution. Observe that $|v\rangle$ does not have squared norm 1 while $|\psi\rangle$ does. $|\psi\rangle$ is our ansatz state. The discretized energy functional is then

$$E = \frac{1}{2}\langle v|\hat{\Delta}|v\rangle - \langle v|f\rangle \tag{4.3}$$

$$= \frac{1}{2}r^2\langle\psi|\hat{\Delta}|\psi\rangle - r\langle\psi|f\rangle \tag{4.4}$$

Now we introduce a state

$$|f,\psi\rangle := \frac{1}{\sqrt{2}}(|0\rangle|f\rangle + |1\rangle|\psi\rangle). \tag{4.5}$$

We also introduce Hermitian operators $I_1 = |1\rangle\langle 1|$, $I_0 = |0\rangle\langle 0|$, $X = |0\rangle\langle 1| + |1\rangle\langle 0|$, such that

$$\langle f,\psi|X\otimes I^{\otimes n}|f,\psi\rangle = \frac{1}{\sqrt{2}}(\langle 0|\langle f| + \langle 1|\langle\psi|)(|0\rangle\langle 1| + |1\rangle\langle 0|)\frac{1}{\sqrt{2}}(|0\rangle|f\rangle + |1\rangle|\psi\rangle) \tag{4.6}$$

$$= \frac{1}{2}\langle f|\psi\rangle + \frac{1}{2}\langle\psi|f\rangle \tag{4.7}$$

$$= \langle f|\psi\rangle\,. \tag{4.8}$$

We use the fact that if $|f\rangle$ and $|\psi\rangle$ are states with real valued amplitudes only, then $\langle\psi|f\rangle = \langle f|\psi\rangle^* = \langle f|\psi\rangle$.

With the Hermitian operators, we can re-write the discretized energy functional as,

$$E = \frac{1}{2}r^2\langle\psi|\hat{\Delta}|\psi\rangle - r\langle\psi|f\rangle \tag{4.9}$$

$$= r^2\langle f,\psi|I_1\otimes\hat{\Delta}|f,\psi\rangle - r\langle f,\psi|X\otimes I^{\otimes n}|f,\psi\rangle \tag{4.10}$$

$$= \langle f,\psi|(r^2 I_1\otimes\hat{\Delta} - rX\otimes I^{\otimes n}|f,\psi\rangle \tag{4.11}$$

We take the partial derivative of the energy functional with respect to r, and set it equal to zero to find the expression for r that minimizes the energy functional

$$\frac{\partial E}{\partial r} = 0 = r\langle f,\psi|I_1\otimes\hat{\Delta}|f,\psi\rangle - \langle f,\psi|X\otimes I^{\otimes n}|f,\psi\rangle \tag{4.12}$$

$$r = \frac{\langle f,\psi|X\otimes I^{\otimes n}|f,\psi\rangle}{2\langle f,\psi|I_1\otimes\hat{\Delta}|f,\psi\rangle}. \tag{4.13}$$

We can now plug r in to find,

$$E = r^2 \langle f, \psi | (I_1 \otimes \hat{\Delta} | f, \psi \rangle - r \langle f, \psi | X \otimes I^{\otimes n} | f, \psi \rangle \tag{4.14}$$

$$= \frac{1}{4} \left(\frac{\langle f, \psi | X \otimes I^{\otimes n} | f, \psi \rangle}{\langle f, \psi | I_1 \otimes \hat{\Delta} | f, \psi \rangle} \right)^2 \langle f, \psi | (I_1 \otimes \hat{\Delta} | f, \psi \rangle \tag{4.15}$$

$$- \frac{1}{2} \frac{\langle f, \psi | X \otimes I^{\otimes n} | f, \psi \rangle}{\langle f, \psi | I_1 \otimes \hat{\Delta} | f, \psi \rangle} \langle f, \psi | X \otimes I^{\otimes n} | f, \psi \rangle \tag{4.16}$$

$$= - \frac{1}{4} \frac{\langle f, \psi | X \otimes I^{\otimes n} | f, \psi \rangle^2}{\langle f, \psi | I_1 \otimes \hat{\Delta} | f, \psi \rangle} \tag{4.17}$$

$$= - \frac{1}{4} \frac{\langle f, \psi | X \otimes I^{\otimes n} | f, \psi \rangle^2}{\frac{1}{2} \langle \psi | \hat{\Delta} | \psi \rangle} \tag{4.18}$$

$$= - \frac{1}{2} \frac{\langle f, \psi | X \otimes I^{\otimes n} | f, \psi \rangle^2}{\langle \psi | \hat{\Delta} | \psi \rangle} \tag{4.19}$$

Note that

$$\langle f, \psi | I_1 \otimes \hat{\Delta} | f, \psi \rangle = \frac{1}{2} \langle \psi | \hat{\Delta} | \psi \rangle . \tag{4.20}$$

Minimizing E with respect to $|\psi\rangle$ would give the solution to the Poisson equation.

$$E = - \frac{1}{2} \frac{\langle f, \psi | X \otimes I^{\otimes n} | f, \psi \rangle^2}{\langle \psi | \hat{\Delta} | \psi \rangle} \tag{4.21}$$

$$= - \frac{1}{2} \frac{\langle f | \psi \rangle^2}{(2 - \langle \psi | \hat{A} | \psi \rangle - \langle \psi | \hat{A}^\dagger | \psi \rangle)} \tag{4.22}$$

4.2 1D Poisson Equation

Using a step function source term given in [12] and the energy minimization cost function from Eq. 4.22, we have produced results for the 1D Poisson equation using a 6 qubit quantum simulator with periodic boundary condition and Dirichlet boundary condition. The full cost function derivation is given in the Appendix. The classical results are produced by inverting the matrix form of the discretized derivative operator on the left hand side of Eq. 4.1. As shown

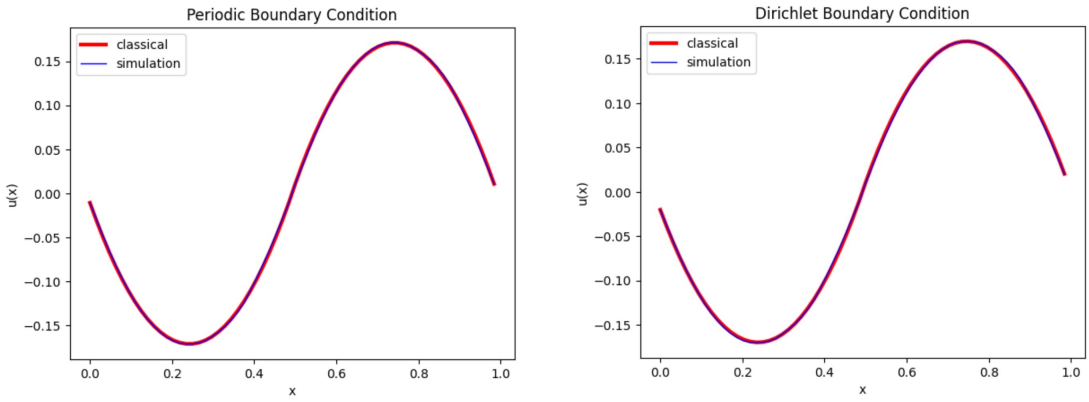


Figure 4.1: Results of the 1D poisson equation solver using a 6 qubit ULA with 38 parameters for the periodic boundary condition and Dirichlet boundary condition. The source term is a step function.

in Fig. 4.1, we optimized 6 qubit ULA with 38 parameters using the Adam optimizer with a step size of 0.01 and 1000 optimization steps for both boundary conditions. We calculate the infidelity as $1 - \langle\psi|u\rangle$ for normalized ansatz state $|\psi\rangle$ and solution state $|u\rangle$. The periodic solution had an infidelity of $4.1E-5$ and the Dirichlet solution had an infidelity of $6.0E-5$. The results can be reproduced here[1].

4.3 2D Poisson Equation

Extending the 1D cost function by simply converting the derivative operator to the 2D version, we get the cost function for the 2D Poisson Equation with a 2D Gaussian function as the source term as shown in Fig. 4.2. For the 2D case, we test out the the ZGR-QFT ansatz. The results can be reproduced here[2].

[1] https://github.com/mcmahon-lab/Quantum-Variational-PDE/blob/6c962f7226ffbb2e8bffe94a9bfdcbb5a659781c/yilian/vqapde/poisson/

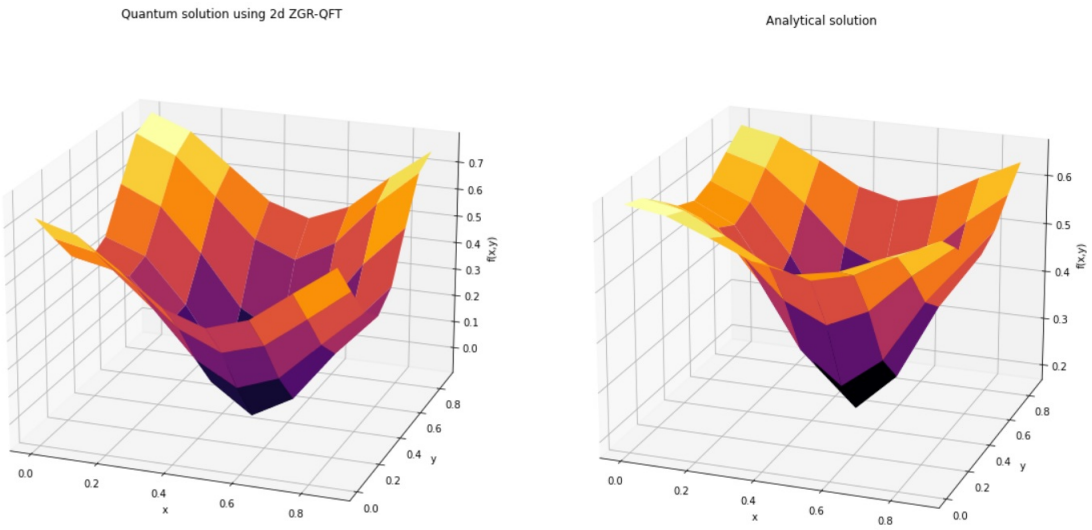


Figure 4.2: Comparison between the analytical results and the solution state using the 2D ZGR-QFT ansatz for the 2D Poisson equation.

boundary_dirichlet_periodic.ipynb

[2] https://github.com/mcmahon-lab/Quantum-Variational-PDE/blob/6c962f7226ffbb2e8bffe94a9bfdcbb5a659781c/yilian/poisson/2d_poisson/2d_real_valued_ZGRQFT-Poisson.ipynb

CHAPTER 5

HEAT EQUATION

In this chapter, we will explore how the energy-minimization method and the distance-minimization method can be applied to the heat equation. We will show the advantages of using the energy-minimization method in the presence of higher dimension equations.

The heat equation is of the form

$$\frac{\partial}{\partial t}u = \Delta u. \tag{5.1}$$

We will denote the previous time step (initial condition) as

$$u(0) = \tilde{u} = \tilde{\lambda_0} \left|\tilde{\psi}\right\rangle$$

and the variational ansatz state parameterized by a list of parameters $\vec{\lambda}$ and scaling factor λ_0,

$$u(\tau) = u = \lambda_0 \left|\psi(\vec{\lambda})\right\rangle = \lambda_0 \left|\psi\right\rangle.$$

We will be using the Backward Euler time discretization method, which gives relation

$$(1 - \tau\Delta)u = \tilde{u} \tag{5.2}$$

5.1 Energy Minimization

5.1.1 Euler-Lagrange Equation

First, we write out the Lagrangian for the heat equation

$$\mathcal{L} = \frac{1}{2}\tau(u')^2 + \frac{1}{2}u^2 - u\tilde{u} \tag{5.3}$$

such that it satisfies the Euler Lagrange equation,

$$-\frac{\mathrm{d}}{\mathrm{d}x}\frac{\partial \mathcal{L}}{\partial u'} + \frac{\partial \mathcal{L}}{\partial u} = 0. \tag{5.4}$$

We check the math,

$$\begin{aligned} -\frac{\mathrm{d}}{\mathrm{d}x}\frac{\partial \mathcal{L}}{\partial u'} + \frac{\partial \mathcal{L}}{\partial u} =& -\tau u'' + u - \tilde{u} \\ =& (1 - \tau\Delta)u - \tilde{u} \\ =& 0. \end{aligned}$$

This Lagrangian works for higher dimensions as well. In two dimensions, for $u(x, y, t)$, the Euler-Lagrange equation becomes

$$-\frac{\partial}{\partial x}\frac{\partial \mathcal{L}}{\partial u'_x} - \frac{\partial}{\partial y}\frac{\partial \mathcal{L}}{\partial u'_y} + \frac{\partial \mathcal{L}}{\partial u} = 0 \tag{5.5}$$

where $u'_x = \frac{\partial u}{\partial x}$ and $u'_y = \frac{\partial u}{\partial y}$. The Lagrangian is then

$$\mathcal{L} = \frac{1}{2}\tau({u'_x}^2 + {u'_y}^2) + \frac{1}{2}u^2 - u\tilde{u}. \tag{5.6}$$

5.1.2 Cost Function

We would like to find the stationary points of the action functional defined as

$$S = \int_\Omega \mathcal{L} d\Omega \tag{5.7}$$

where Ω is the domain of the function.

Observe that the integral can be easily turned into inner products and written in BraKet notation as

$$E = \frac{1}{2}\tau \left\langle u \middle| \hat{\Delta} \middle| u \right\rangle + \frac{1}{2}\langle u \,|\, u\rangle - \langle u \,|\, \tilde{u}\rangle \tag{5.8}$$

$$= \frac{1}{2}\tau\lambda_0^2 \left\langle \psi \middle| \hat{\Delta} \middle| \psi \right\rangle + \frac{1}{2}\lambda_0^2 - \lambda_0\tilde{\lambda}_0 \left\langle \psi \middle| \tilde{\psi} \right\rangle \tag{5.9}$$

We thus have energy functions for 1D, 2D, and 3D,

$$
\begin{array}{l|l}
\text{1D} & E = \frac{1}{2}\tau\lambda_0^2 4^n(-2 + 2\left\langle\psi\right|\hat{A}\left|\psi\right\rangle) + \frac{1}{2}\lambda_0^2 - \lambda_0\tilde{\lambda}_0\left\langle\psi\right|\left.\tilde{\psi}\right\rangle \\
\text{2D} & E = \frac{1}{2}\tau\lambda_0^2 4^n(-4 + 2\left\langle\psi\right|\hat{A}_x\left|\psi\right\rangle + 2\left\langle\psi\right|\hat{A}_y\left|\psi\right\rangle) + \frac{1}{2}\lambda_0^2 - \lambda_0\tilde{\lambda}_0\left\langle\psi\right|\left.\tilde{\psi}\right\rangle \\
\text{3D} & E = \frac{1}{2}\tau\lambda_0^2 4^n(-6 + 2\left\langle\psi\right|\hat{A}_x\left|\psi\right\rangle + 2\left\langle\psi\right|\hat{A}_y\left|\psi\right\rangle + 2\left\langle\psi\right|\hat{A}_z\left|\psi\right\rangle) + \frac{1}{2}\lambda_0^2 - \lambda_0\tilde{\lambda}_0\left\langle\psi\right|\left.\tilde{\psi}\right\rangle
\end{array}
$$

where $\hat{\Delta}$ in each dimension is given by [1]

$$
\begin{aligned}
&\text{1D} \quad \hat{\Delta} = 4^n(-2 + \hat{A} + \hat{A}^\dagger) \\
&\text{2D} \quad \hat{\Delta} = 4^n(-4 + \hat{A}_x + \hat{A}_x{}^\dagger + \hat{A}_y + \hat{A}_y{}^\dagger) \\
&\text{3D} \quad \hat{\Delta} = 4^n(-6 + \hat{A}_x + \hat{A}_x{}^\dagger + \hat{A}_y + \hat{A}_y{}^\dagger + \hat{A}_z + \hat{A}_z{}^\dagger)
\end{aligned}
$$

Note that $\left\langle\psi\right|\hat{A}_i\left|\psi\right\rangle = \left\langle\psi\right|\hat{A}_i^\dagger\left|\psi\right\rangle$ [2].

5.1.3 Results

The results for solving the heat equation using a 6 qubit ULA with 58 paramters is shown in Fig. 5.1. We choose an asymmetric triangle wave as initial condition to demonstrate the robustness of the ULA. We use the Adam optimizer with step size 0.01 and 1000 optimization steps. The results can be reproduced here[3].

[1] In 1D we omit the subscript for $\hat{A}$.

[2] However, in general $\left\langle\psi\right|\hat{A}_i\left|\phi\right\rangle \neq \left\langle\psi\right|\hat{A}_i^\dagger\left|\phi\right\rangle$

[3] `https://github.com/mcmahon-lab/Quantum-Variational-PDE/blob/6c962f7226ffbb2e8bffe94a9bfdcbb5a659781c/yilian/vqapde/heat/heat_solver.ipynb`

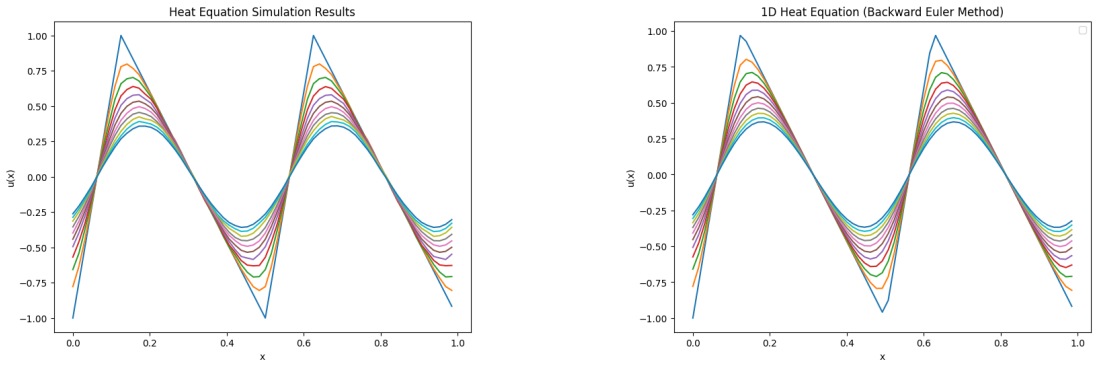


Figure 5.1: Comparison between the backward Euler classical results and the simulated quantum solution to the 1D heat equation using the asymmetric triangle wave as initial condition.

5.2 Distance Minimization

Using Eq. 5.2, we can also write cost function using the Euclidean distance between the two functions

$$C = ||\lambda_0 |\psi\rangle - \lambda_0 \tau \hat{\Delta} |\psi\rangle - \tilde{\lambda}_0 |\tilde{\psi}\rangle ||^2 \tag{5.10}$$

Without specifying dimensions, we can expand the cost function as

$$\begin{aligned}
&||\lambda_0 |\psi\rangle - \lambda_0 \tau \hat{\Delta} |\psi\rangle - \tilde{\lambda}_0 |\tilde{\psi}\rangle ||^2 \\
=&\lambda_0^2 \langle\psi|\psi\rangle + \lambda_0^2 \tau^2 \langle\psi|\hat{\Delta}^2|\psi\rangle + \tilde{\lambda}_0^2 \langle\tilde{\psi}|\tilde{\psi}\rangle + 2Re\left\{-\lambda_0^2 \tau \langle\psi|\hat{\Delta}|\psi\rangle - \lambda_0 \tilde{\lambda}_0 \langle\psi|\tilde{\psi}\rangle + \lambda_0 \tilde{\lambda}_0 \tau \langle\psi|\hat{\Delta}|\tilde{\psi}\rangle\right\} \\
=&\lambda_0^2 + \lambda_0^2 \tau^2 \langle\psi|\hat{\Delta}^2|\psi\rangle + \tilde{\lambda}_0^2 - 2\lambda_0^2 \tau \langle\psi|\hat{\Delta}|\psi\rangle - 2\lambda_0 \tilde{\lambda}_0 \langle\psi|\tilde{\psi}\rangle + 2\lambda_0 \tilde{\lambda}_0 \tau \langle\psi|\hat{\Delta}|\tilde{\psi}\rangle
\end{aligned}$$

5.2.1 Operators

We have defined $\hat{\Delta}$ above, we will now give $\hat{\Delta}^2$ in 1D, 2D, and 3D. Note that in general $[\hat{A}_i, \hat{A}_j] = 0$, thus $\hat{A}_i \hat{A}_j = \hat{A}_j \hat{A}_i$.

In 1D,

$$\hat{\Delta}^2 = 4^{2n}(6 - 4\hat{A} - 4\hat{A}^\dagger + \hat{A}\hat{A} + \hat{A}^\dagger\hat{A}^\dagger)$$

Since we only have $\langle\psi|\hat{\Delta}^2|\psi\rangle$, we can use symmetry to simplify the expression as

$$\hat{\Delta}^2 = 4^{2n}(6 - 8\hat{A} + 2\hat{A}\hat{A})$$

In 2D,

$$\begin{aligned}\hat{\Delta}^2 =& 4^{2n}(20 - 8\hat{A}_x - 8\hat{A}_x^\dagger - 8\hat{A}_y - 8\hat{A}_y^\dagger + 2\hat{A}_x\hat{A}_y + 2\hat{A}_x\hat{A}_y^\dagger \\ &+ 2\hat{A}_x^\dagger\hat{A}_y + 2\hat{A}_x^\dagger\hat{A}_y^\dagger + \hat{A}_x^2 + \hat{A}_x^{\dagger 2} + \hat{A}_y^2 + \hat{A}_y^{\dagger 2})\end{aligned}$$

The symmetry condition gives

$$\hat{\Delta}^2 = 4^{2n}(20 - 16\hat{A}_x - 16\hat{A}_y + 4\hat{A}_x\hat{A}_y + 4\hat{A}_x\hat{A}_y^\dagger + 2\hat{A}_x^2 + 2\hat{A}_y^2)$$

In 3D

$$\begin{aligned}\hat{\Delta}^2 =& 4^{2n}(42 - 12\hat{A}_x - 12\hat{A}_x^\dagger - 12\hat{A}_y - 12\hat{A}_y^\dagger - 12\hat{A}_z - 12\hat{A}_z^\dagger \\ &+ 2\hat{A}_x\hat{A}_y^\dagger + 2\hat{A}_y\hat{A}_x^\dagger + 2\hat{A}_x^\dagger\hat{A}_y^\dagger + 2\hat{A}_x\hat{A}_y + 2\hat{A}_x\hat{A}_z^\dagger + 2\hat{A}_z\hat{A}_x^\dagger \\ &+ 2\hat{A}_x^\dagger\hat{A}_z^\dagger + 2\hat{A}_x\hat{A}_z + 2\hat{A}_z\hat{A}_y^\dagger + 2\hat{A}_y\hat{A}_z^\dagger + 2\hat{A}_y^\dagger\hat{A}_z^\dagger + 2\hat{A}_y\hat{A}_z \\ &+ \hat{A}_x^{\dagger 2} + \hat{A}_x^2 + \hat{A}_y^{\dagger 2} + \hat{A}_y^2 + \hat{A}_z^{\dagger 2} + \hat{A}_z^2).\end{aligned}$$

Again, with the symmetry condition,

$$\begin{aligned}\hat{\Delta}^2 =& 4^{2n}(42 - 24\hat{A}_x - 24\hat{A}_y - 24\hat{A}_z + 4\hat{A}_x\hat{A}_y^\dagger + 4\hat{A}_x\hat{A}_y + 4\hat{A}_x\hat{A}_z^\dagger \\ &+ 4\hat{A}_x\hat{A}_z + 4\hat{A}_y\hat{A}_z^\dagger + 4\hat{A}_y\hat{A}_z + 2\hat{A}_x^2 + 2\hat{A}_y^2 + 2\hat{A}_z^2)\end{aligned}$$

We plug the operators into the cost function in each dimension.

5.2.2 Cost Functions

In 1D

$$\begin{aligned} C =&\lambda_0^2 + \lambda_0^2\tau^2 4^{2n}(6 - 8\langle\psi|\hat{A}|\psi\rangle + 2\langle\psi|\hat{A}\hat{A}|\psi\rangle) + \tilde{\lambda}_0^2 - 2\lambda_0^2\tau 4^n(-2 + 2\langle\psi|\hat{A}|\psi\rangle) \\ &- 2\lambda_0\tilde{\lambda}_0\langle\psi|\tilde{\psi}\rangle + 2\lambda_0\tilde{\lambda}_0\tau 4^n(-2\langle\psi|\tilde{\psi}\rangle + \langle\psi|\hat{A}|\tilde{\psi}\rangle + \langle\psi|\hat{A}^\dagger|\tilde{\psi}\rangle) \end{aligned}$$

In 2D

$$\begin{aligned} C =&\lambda_0^2 + \lambda_0^2\tau^2 4^{2n}(20 - 16\langle\psi|\hat{A}_x|\psi\rangle - 16\langle\psi|\hat{A}_y|\psi\rangle + 4\langle\psi|\hat{A}_x\hat{A}_y|\psi\rangle \\ &l + 4\langle\psi|\hat{A}_x\hat{A}_y^\dagger|\psi\rangle + 2\langle\psi|\hat{A}_x^2|\psi\rangle + 2\langle\psi|\hat{A}_y^2|\psi\rangle) \\ &+ \tilde{\lambda}_0^2 - 2\lambda_0^2\tau 4^n(-4 + 2\langle\psi|\hat{A}_x|\psi\rangle + 2\langle\psi|\hat{A}_y|\psi\rangle) \\ &- 2\lambda_0\tilde{\lambda}_0\langle\psi|\tilde{\psi}\rangle + 2\lambda_0\tilde{\lambda}_0\tau 4^n(-4\langle\psi|\tilde{\psi}\rangle + \langle\psi|\hat{A}_x|\tilde{\psi}\rangle + \langle\psi|\hat{A}_x^\dagger|\tilde{\psi}\rangle + \langle\psi|\hat{A}_y|\tilde{\psi}\rangle + \langle\psi|\hat{A}_y^\dagger|\tilde{\psi}\rangle) \end{aligned}$$

And finally in 3D

$$\begin{aligned} C =&\lambda_0^2 + \lambda_0^2\tau^2 4^{2n}(42 - 24\langle\psi|\hat{A}_x|\psi\rangle - 24\langle\psi|\hat{A}_y|\psi\rangle - 24\langle\psi|\hat{A}_z|\psi\rangle \\ &+ 4\langle\psi|\hat{A}_x\hat{A}_y^\dagger|\psi\rangle + 4\langle\psi|\hat{A}_x\hat{A}_y|\psi\rangle + 4\langle\psi|\hat{A}_x\hat{A}_z^\dagger|\psi\rangle + 4\langle\psi|\hat{A}_x\hat{A}_z|\psi\rangle + 4\langle\psi|\hat{A}_y\hat{A}_z^\dagger|\psi\rangle + 4\langle\psi|\hat{A}_y\hat{A}_z|\psi\rangle \\ &+ 2\langle\psi|\hat{A}_x^2|\psi\rangle + 2\langle\psi|\hat{A}_y^2|\psi\rangle + 2\langle\psi|\hat{A}_z^2|\psi\rangle) \\ &+ \tilde{\lambda}_0^2 - 2\lambda_0^2\tau 4^n(-6 + 2\langle\psi|\hat{A}_x|\psi\rangle + 2\langle\psi|\hat{A}_y|\psi\rangle + 2\langle\psi|\hat{A}_z|\psi\rangle) \\ &- 2\lambda_0\tilde{\lambda}_0\langle\psi|\tilde{\psi}\rangle + 2\lambda_0\tilde{\lambda}_0\tau 4^n(-6\langle\psi|\tilde{\psi}\rangle + \langle\psi|\hat{A}_x|\tilde{\psi}\rangle + \langle\psi|\hat{A}_x^\dagger|\tilde{\psi}\rangle \\ &+ \langle\psi|\hat{A}_y|\tilde{\psi}\rangle + \langle\psi|\hat{A}_y^\dagger|\tilde{\psi}\rangle + \langle\psi|\hat{A}_z|\tilde{\psi}\rangle + \langle\psi|\hat{A}_z^\dagger|\tilde{\psi}\rangle) \end{aligned}$$

5.3 Comparison

As shown in Table. 5.1, the number of expectation circuits is significantly fewer for the energy minimization method compared to the distance minimization method.

	Distance (Lubasch)	Energy (Sato)
1D	5	2
2D	11	3
3D	19	4

Table 5.1: Number of expectation circuits for two methods.

Having fewer expectations circuits is not just a constant factor difference in computation complexity in for the optimization process. It also provides a simpler optimization landscape. We show in Fig. 5.2 the difference in optimizing the two types of cost functions using the same ansatz and same optimizer for the 1D heat equation. The result can be reproduced here[4].

[4] https://github.com/mcmahon-lab/Quantum-Variational-PDE/blob/6c962f7226ffbb2e8bffe94a9bfdcbb5a659781c/yilian/vqapde/heat/heat_solver_lubasch.ipynb

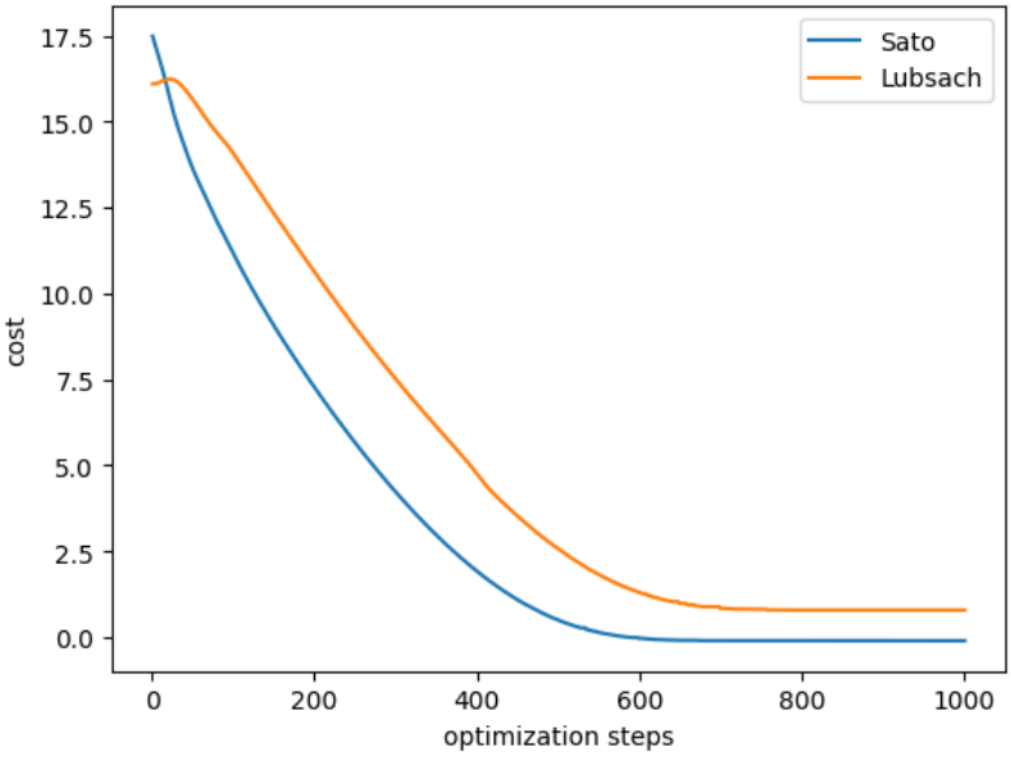


Figure 5.2: Comparison of cost over the optimization for the energy minimization cost function and the distance minimization cost function.

CHAPTER 6

NAVIER-STOKES EQUATION

6.1 1D Viscous Burger's Equation

In 1D, we will study the viscous Burger's equation which is identical to the Navier-Stokes equation with the incompressibility condition removed.

We will derive the full cost function for the 1D Navier Stokes equation as given by [4].

We have

$$C(\lambda_0, \boldsymbol{\lambda}) = \left\| \frac{\lambda_0}{\delta t} |\psi \boldsymbol{\lambda}\rangle - \left(\frac{1}{\delta t} - \tilde{\lambda}_0 \hat{D}_{\tilde{\psi}} \hat{\nabla} + \nu \hat{\nabla}^2 \right) \tilde{\lambda}_0 |\tilde{\psi}\rangle \right\|_2^2 \tag{6.1}$$

We find that up to a positive constant that is independent of λ_0 and $\boldsymbol{\lambda}$, we have

$$C(\lambda_0, \boldsymbol{\lambda}) = \frac{\lambda_0^2}{\delta t^2} - \frac{2\lambda_0 \tilde{\lambda}_0}{\delta t} \left\langle \psi(\boldsymbol{\lambda}) \left| \frac{1}{\delta t} - \tilde{\lambda}_0 \hat{D}_{\tilde{\psi}} \hat{\nabla} + \nu \hat{\nabla}^2 \right| \tilde{\psi} \right\rangle. \tag{6.2}$$

Expanding all terms and we get

$$C(\lambda_0, \boldsymbol{\lambda}) = \tilde{\lambda_0}^2 \left(\frac{1}{t^2} - \frac{4\nu 4^n}{\delta t} \left(\left\langle \tilde{\psi} \middle| A \middle| \tilde{\psi} \right\rangle - 1 \right) \right. \tag{6.3}$$

$$+ 2\tilde{\lambda_0}^2 4^{n-1} \left(\left\langle \tilde{\psi} \middle| A \hat{D}_{\tilde{\psi}} \hat{D}_{\tilde{\psi}} A \middle| \tilde{\psi} \right\rangle - \left\langle \tilde{\psi} \middle| A^\dagger \hat{D}_{\tilde{\psi}} \hat{D}_{\tilde{\psi}} A \middle| \tilde{\psi} \right\rangle \right) \tag{6.4}$$

$$\left. + \nu^2 4^{2n} \left(6 - 8 \left\langle \tilde{\psi} \middle| A \middle| \tilde{\psi} \right\rangle + 2 \left\langle \tilde{\psi} \middle| A^2 \middle| \tilde{\psi} \right\rangle \right) \right) \tag{6.5}$$

$$+ \frac{\boldsymbol{\lambda}^2}{\delta t^2} + 2\boldsymbol{\lambda} \frac{\tilde{\lambda_0}}{\delta t} * \left(\left\langle \psi \middle| \tilde{\psi} \right\rangle / \delta t \right. \tag{6.6}$$

$$+ 2^{n-1} \tilde{\lambda}_0 \left(\left\langle \psi \middle| A \hat{D}_{\tilde{\psi}} \middle| \tilde{\psi} \right\rangle - \left\langle \psi \middle| A^\dagger \hat{D}_{\tilde{\psi}} \middle| \tilde{\psi} \right\rangle \right. \tag{6.7}$$

$$\left. - \nu 4^n \left(-2 \left\langle \psi \middle| \tilde{\psi} \right\rangle + \left\langle \psi \middle| A \middle| \tilde{\psi} \right\rangle + \left\langle \psi \middle| A^\dagger \middle| \tilde{\psi} \right\rangle \right) \right) \tag{6.8}$$

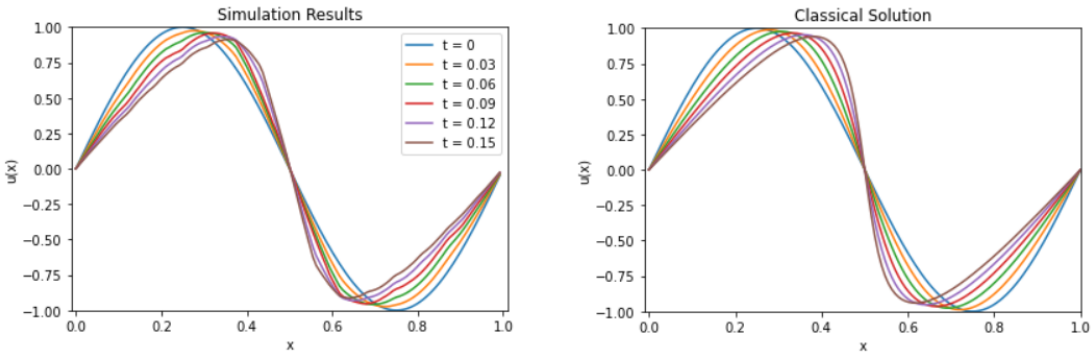


Figure 6.1: Comparison between the analytical results and the solution state for the 1D viscous Burger's equation using the Sine wave as initial condition.

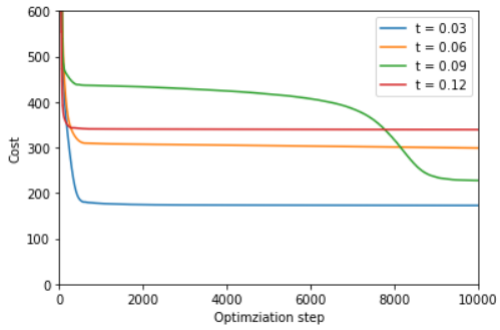


Figure 6.2: The convergence of costs during the optimizations for different time steps for the 1D viscous Burger's equation.

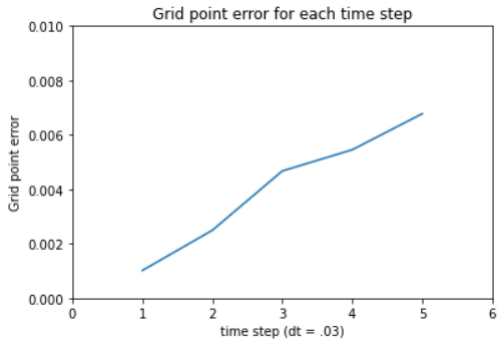


Figure 6.3: The errors of each time step for the 1D Viscous Burger's equation.

6.1.1 Results

We use a 7 qubit ULA to solve the 1d viscous Burgers equation with a backward Euler cost function. A sine wave is set as the initial condition and loaded into the ULA with 80 parameters. Then we optimize an 9 term cost function. 5 time steps were performed in 2.5 hours using the Adam optimizer and 10,000 optimization steps for each time step. The results are shown in Fig. 6.1. The results can be reproduced here[1].

The cost function should have a minimum at exactly 0, but the optimizer had a hard time finding that as shown in Fig. 6.2. It is most likely a barren plateau problem in the cost function as the exact solution at the last time step can be loaded into the ULA with no problem. However, barren plateaus are very much ansatz dependent, meaning changing the design of our ansatz could potentially alleviate the problem. Thus using the gULA could be helpful here. Also shown in the figure, for many time steps, only 2000 optimization steps were significant, so figuring out the hyperparameters needed for the optimization can greatly increase the efficiency of the solver. We need to find the optimal point in the trade off between optimization speed and step size. We also point out that the compounding error is observable, as shown in Fig. 6.3.

[1] https://github.com/mcmahon-lab/Quantum-Variational-PDE/blob/6c962f7226ffbb2e8bffe94a9bfdcbb5a659781c/yilian/burgers/viscous_burgers_1d.ipynb

6.2 2D Navier-Stokes Equation

We will derive the full cost function for the 2D Incompressible Navier Stokes equation as given by arXiv:2106.05782. The Incompressible Navier Stokes equation takes the form,

$$\nabla \cdot \mathbf{V} = 0 \tag{6.9}$$

$$\frac{\partial \mathbf{V}}{\partial t} + (\mathbf{V} \cdot \nabla)\mathbf{V} = -\nabla p + \nu \nabla^2 \mathbf{V}, \tag{6.10}$$

for velocity field $\mathbf{V}$, pressure p, where ν is the kinematic viscosity.

In 2D, the velocity field $\mathbf{V}$ has two components. We will write

$$\mathbf{V}(x, y) = u(x, y)\hat{\mathbf{x}} + v(x, y)\hat{\mathbf{y}}. \tag{6.11}$$

We can further write the discretized version with quantum states $|u\rangle$ and $|v\rangle$,

$$\mathbf{V}(x, y) = \lambda_u |u\rangle \hat{\mathbf{x}} + \lambda_v |v\rangle \hat{\mathbf{y}}. \tag{6.12}$$

6.2.1 Operators

We define quantum operators

$$\frac{\partial}{\partial x} = 2^{n-1}(\hat{A}_x - \hat{A}_x^\dagger) \tag{6.13}$$

$$\frac{\partial}{\partial y} = 2^{n-1}(\hat{A}_y - \hat{A}_y^\dagger) \tag{6.14}$$

$$\hat{A}_x = \hat{A} \otimes \mathbf{1} \tag{6.15}$$

$$\hat{A}_y = \mathbf{1} \otimes \hat{A} \tag{6.16}$$

$$\hat{\nabla}^2 = \frac{\partial}{\partial x}^2 + \frac{\partial}{\partial y}^2 = 4^n(-2 \cdot \mathbf{1} + \hat{A}_x + \hat{A}_x^\dagger) + 4^n(-2 \cdot \mathbf{1} + \hat{A}_y + \hat{A}_y^\dagger), \tag{6.17}$$

$$= 4^n(-4 \cdot \mathbf{1} + \hat{A}_x + \hat{A}_x^\dagger + \hat{A}_y + \hat{A}_y^\dagger) \tag{6.18}$$

where n is the number of qubits in one dimension, i.e. there would be $2n$ total ansatz qubits. Since $\hat{A}_x$ and $\hat{A}_y$ act on different qubits entirely, we can introduce a new operator $\hat{A}_{xy}$ and further simplify the operators

$$\hat{A}_{xy} = \frac{1}{2}\hat{A}_x + \hat{A}_y = \frac{1}{2}\hat{A} \otimes \hat{A} \tag{6.19}$$

$$\hat{\nabla}^2 = 4^n 2(-2 \cdot \mathbf{1} + \hat{A}_{xy} + \hat{A}_{xy}^\dagger) \tag{6.20}$$

$$\hat{\nabla}^4 = 4^{2n+1}(-2 \cdot \mathbf{1} + \hat{A}_{xy} + \hat{A}_{xy}^\dagger)(-2 \cdot \mathbf{1} + \hat{A}_{xy} + \hat{A}_{xy}^\dagger) \tag{6.21}$$

$$= 4^{2n+1}(6 - 4\hat{A}_{xy} - 4\hat{A}_{xy}^\dagger + \hat{A}_{xy}^2 + (\hat{A}_{xy}^\dagger)^2) \tag{6.22}$$

6.2.2 Cost Function

we can write backward Euler cost function

$$C(\mathbf{V}) = \left\| \left(\frac{1}{\delta t} + \mathbf{V} \cdot \overline{\nabla} - \nu \overline{\nabla}^2 \right) \mathbf{V} - \frac{1}{\delta t} \mathbf{V}^* \right\|_2^2 + \mu \| \overline{\nabla} \cdot \mathbf{V} \|_2^2 \tag{6.23}$$

where $\mathbf{V}^*$ is the velocity field at time t and $\mathbf{V}$ is the velocity field at time $t + \delta t$. According to Gourianov et al., by including the penalty term that enforces the imcompressibility condition, the pressure term p can be removed from the cost function.

We can further break the cost function down in terms of u and v.

$$C(u, v) = \left\| \left(\frac{1}{\delta t} + u\frac{\partial}{\partial x} + v\frac{\partial}{\partial y} - \nu\nabla^2 \right) u - \frac{1}{\delta t} u^* \right\|_2^2 + \left\| \left(\frac{1}{\delta t} + u\frac{\partial}{\partial x} + v\frac{\partial}{\partial y} - \nu\nabla^2 \right) v - \frac{1}{\delta t} v^* \right\|_2^2 \tag{6.24}$$

$$+ \mu \left\| \frac{\partial}{\partial x} u + \frac{\partial}{\partial y} v \right\|_2^2 . \tag{6.25}$$

We now turn it into the discretized version, denoting

$$C(|u\rangle, \lambda_u, |v\rangle, \lambda_v) = \left\| \left(\frac{1}{\delta t} + \hat{D}_u \frac{\partial}{\partial x} + \hat{D}_v \frac{\partial}{\partial y} - \nu \hat{\nabla}^2 \right) \lambda_u |u\rangle - \frac{\lambda_u^*}{\delta t} |u^*\rangle \right\|_2^2 \tag{6.26}$$

$$+ \left\| \left(\frac{1}{\delta t} + \hat{D}_u \frac{\partial}{\partial x} + \hat{D}_v \frac{\partial}{\partial y} - \nu \hat{\nabla}^2 \right) \lambda_v |v\rangle - \frac{\lambda_v^*}{\delta t} |v^*\rangle \right\|_2^2 \tag{6.27}$$

$$+ \mu \left\| \lambda_u \frac{\partial}{\partial x} |u\rangle + \lambda_v \frac{\partial}{\partial y} |v\rangle \right\|_2^2 . \tag{6.28}$$

We will treat each term individually

$$\left\| \left(\frac{1}{\delta t} + \lambda_u \hat{D}_u \frac{\partial}{\partial x} + \lambda_v \hat{D}_v \frac{\partial}{\partial y} - \nu \hat{\nabla}^2 \right) \lambda_u |u\rangle - \frac{\lambda_u^*}{\delta t} |u^*\rangle \right\|_2^2 \tag{6.29}$$

$$= \left(\frac{\lambda_u}{\delta t} \langle u| + \lambda_u \langle u| \frac{\partial}{\partial x}^\dagger \lambda_u \hat{D}_u + \lambda_u \langle u| \frac{\partial}{\partial y}^\dagger \lambda_v \hat{D}_v - \nu \lambda_u \langle u| \hat{\nabla}^2 - \frac{\lambda_u^*}{\delta t} \langle u^*| \right) \tag{6.30}$$

$$\left(\frac{\lambda_u}{\delta t} |u\rangle + \lambda_u \hat{D}_u \frac{\partial}{\partial x} \lambda_u |u\rangle + \lambda_v \hat{D}_v \frac{\partial}{\partial y} \lambda_u |u\rangle - \nu \hat{\nabla}^2 \lambda_u |u\rangle - \frac{\lambda_u^*}{\delta t} |u^*\rangle \right) \tag{6.31}$$

$$= \frac{\lambda_u^2}{\delta t^2} + \lambda_u^4 \left\langle u \left| \frac{\partial}{\partial x}^\dagger \hat{D}_u \hat{D}_u \frac{\partial}{\partial x} \right| u \right\rangle + \lambda_u^2 \lambda_v^2 \left\langle u \left| \frac{\partial}{\partial y}^\dagger \hat{D}_v \hat{D}_v \frac{\partial}{\partial y} \right| u \right\rangle + \nu^2 \lambda_u^2 \left\langle u \left| \hat{\nabla}^4 \right| u \right\rangle + \frac{(\lambda_u^*)^2}{\delta t^2} \tag{6.32}$$

$$- 2\nu \frac{\lambda_u^2}{\delta t} \left\langle u \left| \hat{\nabla}^2 \right| u \right\rangle - 2 \frac{\lambda_u \lambda_u^*}{\delta t^2} \langle u | u^* \rangle + 2\lambda_u^3 \lambda_v \left\langle u \left| \frac{\partial}{\partial x}^\dagger \hat{D}_u \hat{D}_v \frac{\partial}{\partial y} \right| u \right\rangle \tag{6.33}$$

$$+ 2\nu \frac{\lambda_u \lambda_u^*}{\delta t} \left\langle u \left| \hat{\nabla}^2 \right| u^* \right\rangle \tag{6.34}$$

$$= \frac{\lambda_u^2}{\delta t^2} + \lambda_u^4 4^n \left(\left\langle u \left| \hat{A}_x \hat{D}_u \hat{D}_u \hat{A}_x^\dagger \right| u \right\rangle - \left\langle u \left| \hat{A}_x \hat{D}_u \hat{D}_u \hat{A}_x \right| u \right\rangle \right) \tag{6.35}$$

$$+ \lambda_u^2 \lambda_v^2 4^n \left(\left\langle u \left| \hat{A}_y \hat{D}_v \hat{D}_v \hat{A}_y^\dagger \right| u \right\rangle - \left\langle u \left| \hat{A}_y \hat{D}_v \hat{D}_v \hat{A}_y \right| u \right\rangle \right) \tag{6.36}$$

$$+ \nu^2 \lambda_u^2 4^{2n+1} \left(6 - 8 \left\langle u \left| \hat{A}_{xy} \right| u \right\rangle + 2 \left\langle u \left| \hat{A}_{xy}^2 \right| u \right\rangle \right) + \frac{(\lambda_u^*)^2}{\delta t^2} \tag{6.37}$$

$$- 2\nu \frac{\lambda_u^2}{\delta t} 4^{n+1} \left(-2 + 2 \left\langle u \left| \hat{A}_{xy} \right| u \right\rangle \right) - 2 \frac{\lambda_u \lambda_u^*}{\delta t^2} \langle u | u^* \rangle \tag{6.38}$$

$$+ 2\lambda_u^3 \lambda_v \left(\left\langle u \left| \hat{A}_x^\dagger \hat{D}_u \hat{D}_v \hat{A}_y \right| u \right\rangle + \left\langle u \left| \hat{A}_x \hat{D}_u \hat{D}_v \hat{A}_y^\dagger \right| u \right\rangle - \left\langle u \left| \hat{A}_x \hat{D}_u \hat{D}_v \hat{A}_y \right| u \right\rangle - \left\langle u \left| \hat{A}_x^\dagger \hat{D}_u \hat{D}_v \hat{A}_y^\dagger \right| u \right\rangle \right) \tag{6.39}$$

$$+ \nu \frac{\lambda_u \lambda_u^*}{\delta t} 4^{n+1} (-2 \langle u | u^* \rangle + \left\langle u \left| \hat{A}_{xy} \right| u^* \right\rangle + \left\langle u \left| \hat{A}_{xy}^\dagger \right| u^* \right\rangle) \tag{6.40}$$

Similarly, we have

$$\left\|\left(\frac{1}{\delta t}+\lambda_u\hat{D}_u\frac{\partial}{\partial x}+\lambda_v\hat{D}_v\frac{\partial}{\partial y}-\nu\hat{\nabla}^2\right)\lambda_v\left|v\right\rangle-\frac{\lambda_v^*}{\delta t}\left|v^*\right\rangle\right\|_2^2 \tag{6.41}$$

$$=\frac{\lambda_v^2}{\delta t^2}+\lambda_v^2\lambda_u^2 4^n\left(\left\langle v\right|\hat{A}_x\hat{D}_u\hat{D}_u\hat{A}_x^\dagger\left|v\right\rangle-\left\langle v\right|\hat{A}_x\hat{D}_u\hat{D}_u\hat{A}_x\left|v\right\rangle\right) \tag{6.42}$$

$$+\lambda_v^4 4^n\left(\left\langle v\right|\hat{A}_y\hat{D}_v\hat{D}_v\hat{A}_y^\dagger\left|v\right\rangle-\left\langle v\right|\hat{A}_y\hat{D}_v\hat{D}_v\hat{A}_y\left|v\right\rangle\right) \tag{6.43}$$

$$+\nu^2\lambda_v^2 4^{2n+1}\left(6-8\left\langle v\right|\hat{A}_{xy}\left|v\right\rangle+2\left\langle v\right|\hat{A}_{xy}^2\left|v\right\rangle\right)+\frac{(\lambda_v^*)^2}{\delta t^2} \tag{6.44}$$

$$-2\nu\frac{\lambda_v^2}{\delta t}4^{n+1}\left(-2+2\left\langle v\right|\hat{A}_{xy}\left|v\right\rangle\right)-2\frac{\lambda_v\lambda_v^*}{\delta t^2}\left\langle v|v^*\right\rangle \tag{6.45}$$

$$+2\lambda_v^3\lambda_u\left(\left\langle v\right|\hat{A}_x^\dagger\hat{D}_u\hat{D}_v\hat{A}_y\left|v\right\rangle+\left\langle v\right|\hat{A}_x\hat{D}_u\hat{D}_v\hat{A}_y^\dagger\left|v\right\rangle-\left\langle v\right|\hat{A}_x\hat{D}_u\hat{D}_v\hat{A}_y\left|v\right\rangle-\left\langle v\right|\hat{A}_x^\dagger\hat{D}_u\hat{D}_v\hat{A}_y^\dagger\left|v\right\rangle\right) \tag{6.46}$$

$$+\nu\frac{\lambda_v\lambda_v^*}{\delta t}4^{n+1}(-2\left\langle v\right|v^*\rangle+\left\langle v\right|\hat{A}_{xy}\left|v^*\right\rangle+\left\langle v\right|\hat{A}_{xy}^\dagger\left|v^*\right\rangle) \tag{6.47}$$

Lastly, we have the penalty term

$$\left\|\lambda_u\frac{\partial}{\partial x}\left|u\right\rangle+\lambda_v\frac{\partial}{\partial y}\left|v\right\rangle\right\|_2^2 \tag{6.48}$$

$$=\left(\lambda_u\left\langle u\right|\frac{\partial}{\partial x}^\dagger+\lambda_v\left\langle v\right|\frac{\partial}{\partial y}^\dagger\right)\left(\lambda_u\frac{\partial}{\partial x}\left|x\right\rangle+\lambda_v\frac{\partial}{\partial y}\left|v\right\rangle\right) \tag{6.49}$$

$$=-\lambda_u^2\left\langle u\right|\frac{\partial}{\partial x}^2\left|u\right\rangle-\lambda_v^2\left\langle v\right|\frac{\partial}{\partial y}^2\left|v\right\rangle+2\lambda_v\lambda_u\left\langle v\right|\frac{\partial}{\partial y}^\dagger\frac{\partial}{\partial x}\left|u\right\rangle \tag{6.50}$$

$$=-\lambda_u^2 4^n(-2+2\left\langle u\right|\hat{A}_x\left|u\right\rangle)-\lambda_v^2 4^n(-2+2\left\langle v\right|\hat{A}_y\left|v\right\rangle) \tag{6.51}$$

$$+2\lambda_v\lambda_u 4^{n-1}\left(\left\langle v\right|\hat{A}_y^\dagger\hat{A}_x\left|u\right\rangle-\left\langle v\right|\hat{A}_y\hat{A}_x\left|u\right\rangle-\left\langle v\right|\hat{A}_y^\dagger\hat{A}_x^\dagger\left|u\right\rangle+\left\langle v\right|\hat{A}_y\hat{A}_x^\dagger\left|u\right\rangle\right) \tag{6.52}$$

CHAPTER 7

CONCLUSION AND FUTURE WORK

7.1 Conclusion

In this thesis, we explored the designs of variational quantum algorithms for solving linear and nonlinear partial differential equations. For cost function formulation, we compared the distance minimization approach and the energy minimization approach. Our derivation and results demonstrate that the energy minimization approach requires fewer expectations circuits and thus has a simpler optimization space for the optimizer. The advantage of the energy minimization approach is even more significant for higher dimensions.

We also present two ansatz designs, the ZGR-QFT ansatz and the ULA. For the ULA, we offer tailored optimization techniques, including genetic optimization and gradient-free optimization. We also discuss how different boundary conditions can be achieved using correction circuits. Using these tools, we present results for the 1D poisson equation in periodic boundary condition and Dirichlet boundary condition, the 2D poisson equation, the 1D heat equation, and the 1D viscous Burger's equation. We note that simulation results of higher dimension as well as nonlinear PDEs using variational quantum algorithms are not currently present in literature. Furthermore, using the energy minimization approach, we present a feasible cost function for the 3D heat equation, which has also not been presented by other literature.

7.2 Future Work

We started out this thesis by introducing five crucial problems in designing VQAs for solving PDEs: (1) State preparation for initial conditions, (2) Ansatz design, (3) Implementation of boundary conditions, (4) Optimization, and (5) Read-Out.

7.2.1 State Preparation and Boundary conditions

For (1) state preparations, and (3) boundary conditions, we presented efficient solutions. The ZGR-QFT ansatz for state preparation is $O(2^m) + O(n^2)$ in depth and the mask circuit for boundary condition requires only n ancilla qubits. However, these are general solutions for arbitrary state preparation and arbitrary boundary conditions. Problem-specific circuits can be designed to further reduce the depth of these circuits.

7.2.2 Ansatz and Optimization

The results of this thesis show that (2) ansatz design and (4) optimization remain two of the biggest challenges for fully implementing the VQA. In this thesis, we offer promising approaches, such as the ULA and the genetic algorithm optimization. However, during the numerical experiments for this thesis, we were unable to find a robust general optimization strategy that can efficiently optimize the ansatz structure space and the parameter space *at the same time*. We note that such a structure optimization strategy would also solve ansatz design

and optimization problems at the same time. We offer a code base that can be used to study ULA[1], and the genetic algorithm for ansatz structure optimization[2].

While in Section. 3.5 we mostly focused on genetic algorithm-based structure optimization, for future studies on ULA structure optimization strategies, we propose three general approaches: Pruning, Growing, and Mutation.

Pruning

In the pruning approach, a larger circuit is first optimized and rotation gates can be pruned based on the optimized parameter value for the rotation gate. The idea is that very small rotations have less significant effects on the ansatz state, and thus they are more likely to be redundant gates in the ansatz.

Growing

Alternatively, we can start with a smaller circuit and add rotation gates to the circuit as we optimize.

Mutation

The mutation approach is the general genetic algorithm approach where different ansatz structures of the same depth are tested to find the best structure.

[1] https://github.com/mcmahon-lab/Quantum-Variational-PDE/blob/01381766d34ad09941818cf211cd89bd3baef038/yilian/vqapde/vqapde/ula.py

[2] https://github.com/mcmahon-lab/Quantum-Variational-PDE/blob/01381766d34ad09941818cf211cd89bd3baef038/yilian/vqapde/vqapde/genetic.py

The above-mentioned three approaches can be applied individually or at the same time to achieve optimal results with the fewest amount of cost function evaluations.

7.2.3 Readout

The last problem, (5) read-out, was not explored in this thesis. Since we worked with simulators of quantum computers, we had full access to the quantum state easily. However, in real quantum computers, access to the full quantum state requires tomography which in turn requires exponential measurements. This would be intractable for large number of qubits. Therefore, for any meaningful implementation of the VQA, efficient read-out methods would likely be problem-specific, which extracts some properties of the optimized ansatz state instead of the entire state.

APPENDIX A

COST FUNCTION DERIVATIONS

A.1 Distance Minimization Poisson Equation Cost Function

For linear Poisson equations of the form

$$-\nabla^2 u(\mathbf{x}) = f(\mathbf{x}) \tag{A.1}$$

where $f(\mathbf{x})$ is some density function given as an initial condition, we can derive a cost function based on Lubasch's method. First, we write out the discretized version of the cost function

$$r\hat{\Delta}\,|\psi\rangle = |f\rangle\,, \tag{A.2}$$

where $r\,|\psi\rangle = |u\rangle$ such that $|\psi\rangle$ is a quantum state with squared norm 1. We note

$$\hat{\Delta} = 2 - \hat{A} - \hat{A}^\dagger, \tag{A.3}$$

and

$$\begin{aligned}
\hat{\Delta}^2 =& (2 - \hat{A} - \hat{A}^\dagger)(2 - \hat{A} - \hat{A}^\dagger) && \text{(A.4)}\\
=& 4 - 2\hat{A} - 2\hat{A}^\dagger - 2\hat{A} + \hat{A}\hat{A} + \hat{A}\hat{A}^\dagger - 2\hat{A}^\dagger + \hat{A}^\dagger\hat{A} + \hat{A}^\dagger\hat{A}^\dagger && \text{(A.5)}\\
=& 6 - 4\hat{A} - 4\hat{A}^\dagger + \hat{A}\hat{A} + \hat{A}^\dagger\hat{A}^\dagger && \text{(A.6)}
\end{aligned}$$

Using Lubasch's distance minimization method, we can write out a cost function

$$C[\psi] = |r\hat{\Delta}\,|\psi\rangle - |f\rangle\,|^2 \tag{A.7}$$

$$= (r\,\langle\psi|\,\hat{\Delta} - \langle f|)(r\hat{\Delta}\,|\psi\rangle - |f\rangle\,|^2) \tag{A.8}$$

$$= r^2\,\langle\psi|\hat{\Delta}^2|\psi\rangle - r\,\langle f|\hat{\Delta}|\psi\rangle - r\,\langle\psi|\hat{\Delta}|f\rangle + \langle f|f\rangle \tag{A.9}$$

$$= r^2\,\langle\psi|(6 - 4\hat{A} - 4\hat{A}^\dagger + \hat{A}\hat{A} + \hat{A}^\dagger\hat{A}^\dagger)|\psi\rangle - 2r\,\langle f|(6 - 4\hat{A} - 4\hat{A}^\dagger + \hat{A}\hat{A} + \hat{A}^\dagger\hat{A}^\dagger)|\psi\rangle + 1 \tag{A.10}$$

$$= r^2(6 - 4\,\langle\psi|\hat{A}|\psi\rangle - 4\,\langle\psi|\hat{A}^\dagger|\psi\rangle + \langle\psi|\hat{A}\hat{A}|\psi\rangle + \langle\psi|\hat{A}^\dagger\hat{A}^\dagger|\psi\rangle) \tag{A.11}$$

$$- 2r(6\,\langle f|\psi\rangle - 4\,\langle f|\hat{A}|\psi\rangle - 4\,\langle f|\hat{A}^\dagger|\psi\rangle + \langle f|\hat{A}\hat{A}|\psi\rangle + \langle f|\hat{A}^\dagger\hat{A}^\dagger|\psi\rangle) + 1 \tag{A.12}$$

$$= r^2(6 - 8\,\langle\psi|\hat{A}|\psi\rangle + 2\,\langle\psi|\hat{A}\hat{A}|\psi\rangle) \tag{A.13}$$

$$- 2r(6\,\langle f|\psi\rangle - 4\,\langle f|\hat{A}|\psi\rangle - 4\,\langle f|\hat{A}^\dagger|\psi\rangle + \langle f|\hat{A}\hat{A}|\psi\rangle + \langle f|\hat{A}^\dagger\hat{A}^\dagger|\psi\rangle) + 1 \tag{A.14}$$

Note that we assume both $|\psi\rangle$ and $|f\rangle$ are real valued so that

$$\langle\psi|f\rangle = \langle f|\psi\rangle^* = \langle f|\psi\rangle$$

$$\langle\psi|\hat{A}|\psi\rangle = \langle\psi|\hat{A}^\dagger|\psi\rangle$$

$$\langle f|\hat{A}|\psi\rangle = \langle\psi|\hat{A}^\dagger|f\rangle$$

APPENDIX B

VQAPDE LIBRARY

A companion Python library has been developed to expedite the process of studying VQAs named the VQAPDE library. To use, simply

```
import vqapde as vp
```

The Ansatz class takes a function that defines the ansatz circuit and the number of qubits for the ansatz to initialize. Some pre-defined parametrized child classes include `ULA`, `ZGR`, `ZGRQFT`.

```
ula_ansatz = vp.ULA(6)
```

Unparametrized ansatzes can be created by using the `Ansatz_no_param` class. For example, the step function initial condition is created here:

```
def step_ansatz(wires):
    qml.PauliX(wires[0])
    for i in wires:
        qml.Hadamard(wires=i)
step_function = vp.Ansatz_no_param(step_ansatz, 6, name='step')
```

A QNPU is defined by two `Ansatz` objects and any number of operators. Here, we define three `QNPU`s.

```
q0 = vp.QNPU(ula_ansatz, step_function)
q1 = vp.QNPU(ula_ansatz, ula_ansatz, vp.Adder(1))
q2 = vp.QNPU(ula_ansatz,ula_ansatz,vp.Adder(-1))
```

And finally, the `Cost_Function` class takes two ansatz objects, and a list of `QNPU`s to initialize

```
c = vp.Cost_Function(ula_ansatz, step_function, [q0,q1,q2])
```

If we inspect the `QNPUs` of the `Cost_Function` object we just created,

```
c.print_QNPUs()
```

we would get output

```
exps[0]=<ULA|step>

exps[1]=<ULA|A^1|ULA>

exps[2]=<ULA|A^-1|ULA>
```

Using Eq. 4.22, we can then define the cost function expression as

```
c.define_cost('-.5*exps[0]**2/(2-exps[1]-exps[2])')
```

The `Cost_Function` can then be optimized by

```
c.minimize_cost(initial_params, opt_steps=1000, opt = 'adam',
                                    step_size=0.01)
```

BIBLIOGRAPHY

[1] Bruno Costa. Spectral methods for partial differential equations. *CUBO, A Mathematical Journal*, 6(4):1–32, Dec. 2004.

[2] C Armando Duarte and JT Oden. *A review of some meshless methods to solve partial differential equations*. Texas Institute for Computational and Applied Mathematics Austin, TX, 1995.

[3] Gerhard Dziuk and Charles M. Elliott. Finite element methods for surface pdes. *Acta Numerica*, 22:289–396, 2013.

[4] Nikita Gourianov, Michael Lubasch, Sergey Dolgov, Quincy Y. van den Berg, Hessam Babaee, Peyman Givi, Martin Kiffner, and Dieter Jaksch. A quantum-inspired approach to exploit turbulence structures. *Nature Computational Science*, 2(1):30–37, jan 2022.

[5] Martin Larocca, Nathan Ju, Diego García-Martín, Patrick J. Coles, and M. Cerezo. Theory of overparametrization in quantum neural networks, 2021.

[6] Michael Lubasch, Jaewoo Joo, Pierre Moinier, Martin Kiffner, and Dieter Jaksch. Variational quantum algorithms for nonlinear problems. *Physical Review A*, 101(1):010301, January 2020. arXiv: 1907.09032.

[7] Jarrod R. McClean, Sergio Boixo, Vadim N. Smelyanskiy, Ryan Babbush, and Hartmut Neven. Barren plateaus in quantum neural network training landscapes. *Nature Communications*, 9(1), 2018.

[8] Mudassir Moosa. Quantum-Variational-PDE/Adder_circuit_with_qft.pdf at master · mcmahon-lab/Quantum-Variational-PDE.

[9] Mikko Mottonen, Juha J. Vartiainen, Ville Bergholm, and Martti M. Salomaa. Transformation of quantum states using uniformly controlled rotations. *arXiv:quant-ph/0407010*, July 2004. arXiv: quant-ph/0407010.

[10] Mateusz Ostaszewski, Edward Grant, and Marcello Benedetti. Structure optimization for parameterized quantum circuits. *Quantum*, 5:391, January 2021.

[11] Robert M. Parrish, Edward G. Hohenstein, Peter L. McMahon, and Todd J.

Martínez. Quantum computation of electronic transitions using a variational quantum eigensolver. *Phys. Rev. Lett.*, 122:230401, Jun 2019.

[12] Yuki Sato, Ruho Kondo, Satoshi Koide, Hideki Takamatsu, and Nobuyuki Imoto. A variational quantum algorithm based on the minimum potential energy for solving the Poisson equation. *Physical Review A*, 104(5):052409, November 2021. arXiv: 2106.09333.

[13] E. Tonti. Variational formulation for every nonlinear problem. *International Journal of Engineering Science*, 22(11):1343–1371, 1984.

[14] Lloyd Nicholas Trefethen. Finite difference and spectral methods for ordinary and partial differential equations. 1996.

[15] Hanrui Wang, Yongshan Ding, Jiaqi Gu, Zirui Li, Yujun Lin, David Z. Pan, Frederic T. Chong, and Song Han. Quantumnas: Noise-adaptive search for robust quantum circuits, 2021.