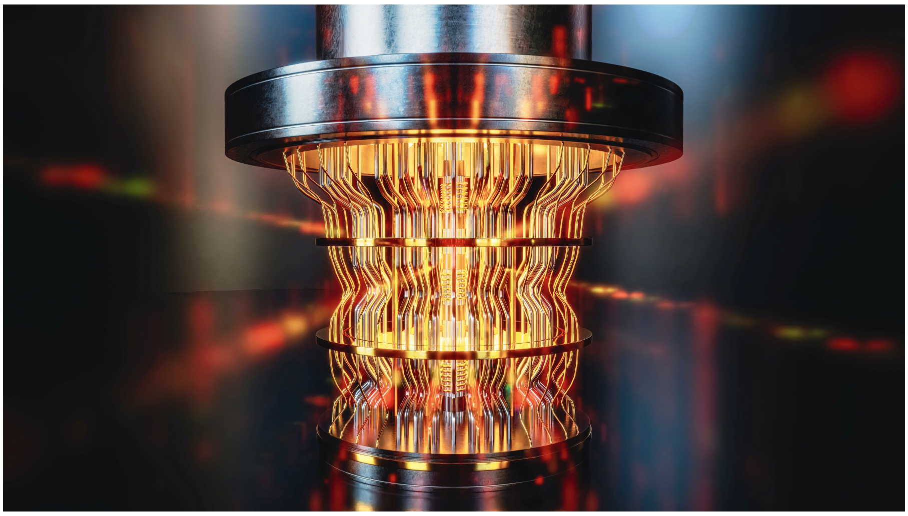
Computational Physics Projects

Quantum Machine Learning

Avinash Chaudhari
Roll No: 23B1064
Department of Computer Science and Engineering

Mentor: Aditya Saran



Contents

Chapter 1

Quantum Computing Basics

1.1 Linear Algebra

Revised linear algebra concepts that are used in quantum computing, such as vector spaces, bases and linear independence, linear operators and matrices. Also revised inner product, outer product, eigenvectors, eigenvalues, trace of a matrix.

1.1.1 Pauli Matrices

There are 4 important matrices in quantum computing, called the Pauli matrices. They are used to represent the I, X, Y, and Z gates.

$$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \quad Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{1.1}$$

1.1.2 Adjoints and Hermitian Operators

Suppose A is any linear operator acting on a vector space. The adjoint of A, denoted by $A^\dagger$, is the complex conjugate of the transpose of A. If $A = A^\dagger$, then A is called a Hermitian operator.

1.1.3 Tensor Product

The tensor product of two matrices A and B is denoted by $A \otimes B$. The tensor product of two matrices is a block matrix whose blocks are formed by multiplying each element of A by the matrix B.
The tensor product of two vectors, $|v\rangle \in V$ and $|w\rangle \in W$ is denoted by $|v\rangle \otimes |w\rangle$. The tensor product of two vectors is a vector in the tensor product space $V \otimes W$.

1.2 Quantum Mechanics Postulates

1.2.1 State Space

Postulate 1. *Associated to any isolated physical system is a complex vector space with inner product (Hilbert space) known as the **state space** of the system. The system is completely described by the **state vector**, which is a unit vector in the system's state space.*

1.2.2 Evolution

Postulate 2. *The evolution of a closed quantum system is described by a unitary transformation. That is, the state $|\psi\rangle$ of the system at time t_1 is related to the state $|\psi'\rangle$ of the system at time t_2 by a unitary operator U which depends only on the times t_1 and t_2,*

$$|\psi'\rangle = U|\psi\rangle \tag{1.2}$$

1.2.3 Quantum Measurement

Postulate 3. *Quantum measurements are described by a collection M_m of measurement operators. These operators acting on the state space of the system being measured. The index m refers to the measurement outcomes that may occur in the experiment. If the state of the quantum system is $|\psi\rangle$ immediately before the measurement than the probability that result m occurs is given by*

$$p(m) = \langle\psi| M_m^\dagger M_m |\psi\rangle \tag{1.3}$$

and the state of the system after the measurement is

$$\frac{M_m|\psi\rangle}{\sqrt{\langle\psi| M_m^\dagger M_m |\psi\rangle}} \tag{1.4}$$

The measurement operators satisfy the ***completeness equation****,*

$$\sum_m M_m^\dagger M_m = I \tag{1.5}$$

1.2.4 Projective Measurements

Definition 1. *A projective measurement is described by an observable, M, a Hermitian operator on the state space of the system being observed. The obsevable has a spectral decomposition*

$$M = \sum_m m P_m \tag{1.6}$$

where P_m is the projector onto the eigenspace of M with eigenvalue m. The possible outcomes of the measurement correspond to the eigenvalues, m, of the observable.

1.2.5 Composite Systems

Suppose, we are interested in a system consisting of two (or more) qubits. Then, such a system is called a *composite system.* The way of representing the state of the system which gives the whole information about the qubits in it is given by this next postulate.

Postulate 4. *The state space of a physical system is the tensor product of the state spaces of the component physical systems. Moreover, if we have systems numbered 1 through n, and system number i is prepared in the state $|\psi_i\rangle$, then the joint state of the total system is $|\psi_1\rangle \otimes |\psi_2\rangle \otimes \cdots \otimes |\psi_n\rangle$*

1.3 Quantum Circuits

1.3.1 Quantum Gates

Unitarity is the only condition on quantum gates. So, if U is a quantum gate, then

$$U^{\dagger}U = I \tag{1.7}$$

The following are some of the important quantum gates used in quantum computing.

1. **Identity Gate:** $I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

2. **Hadamard Gate:** $H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$

3. **Pauli Gates:**

   (a) $\mathbf{X} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$

   (b) $\mathbf{Y} = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$

   (c) $\mathbf{Z} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$

4. **CNOT** $= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$

1.3.2 Quantum Circuits

Quantum circuits are collections of *quantum gates* and *measurement operators* interconnected by quantum wires.

1.3.3 Quantum Entanglement

A state of a combined system is separable if it can be expressed as the tensor product of the states of the components. For example:

$$\frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle) = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \tag{1.8}$$

Not every 4-dimensional vector (system of two qubits) can be written as a tensor product of two 2-dimensional vectors (qubits). For example:

$$\frac{1}{2}(|01\rangle + |10\rangle); \qquad \frac{1}{2}(|00\rangle + |11\rangle) \tag{1.9}$$

Such systems are called *entangled*.

- **Separable state:**
  If a two qubit state can be written as $|\psi\rangle = |\phi_1\rangle \otimes |\phi_2\rangle$ for some qubit states $|\phi_1\rangle$ and $|\phi_2\rangle$, then the state $|\psi\rangle$ is said to be *separable*.

- **Entangled state:**
  If a two qubit state cannot be written as $|\psi\rangle = |\phi_1\rangle \otimes |\phi_2\rangle$ for any possible choice of $|\phi_1\rangle$ and $|\phi_2\rangle$, then the state $|\psi\rangle$ is said to be in entangled state.

Through entanglement some special states can be achieved:

$$
\begin{aligned}
00: |\psi_{00}\rangle &= \frac{|00\rangle + |11\rangle}{\sqrt{2}} \\
01: |\psi_{01}\rangle &= \frac{|00\rangle - |11\rangle}{\sqrt{2}} \\
10: |\psi_{10}\rangle &= \frac{|10\rangle + |01\rangle}{\sqrt{2}} \\
11: |\psi_{11}\rangle &= \frac{|01\rangle + |10\rangle}{\sqrt{2}}
\end{aligned} \tag{1.10}
$$

These four states are known as the *Bell basis*, *Bell states* or *EPR pairs*

1.3.4 Quantum Teleportation

Eve prepares an entangled pair of qubits. She transfers one qubit to Alice, while another to Bob. Alice wants to deliver a qubit $|\psi\rangle$ to Bob but she doesn't know his location. The only thing that Alice knows is that both of them share a pair of entangled qubits that Eve prepared. Alice doesn't even know the state $|\psi\rangle$. Even if she knew the state $|\psi\rangle$, describing it precisely takes an infinite amount of classical information since $|\psi\rangle$ takes values in a *continuous* space.
The state to be teleported is $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, where α and β are unknown amplitudes. The state input into the circuit $|\psi_0\rangle$ is

$$|\psi_0\rangle = \frac{1}{\sqrt{2}}\left[\alpha|0\rangle(|00\rangle + |11\rangle) + \beta|1\rangle(|00\rangle + |11\rangle)\right] \tag{1.11}$$

where the first two qubits belong to Alice and the last one belongs to Bob. Alice sends her qubit through a CNOT gate, and now the system is in the state:

$$|\psi_1\rangle = \frac{1}{\sqrt{2}}\left[\alpha|0\rangle(|00\rangle + |11\rangle) + \beta|1\rangle(|10\rangle + |01\rangle)\right] \tag{1.12}$$

She then sends her first qubit through Hadamard gate,

$$
\begin{aligned}
|\psi_2\rangle &= \frac{1}{2}\left[\alpha(|0\rangle + |1\rangle)(|00\rangle + |11\rangle) + \beta(|0\rangle - |1\rangle)(|10\rangle + |01\rangle)\right] \\
&= \frac{1}{2}[|00\rangle(\alpha|0\rangle + \beta|1\rangle) + |01\rangle(\alpha|1\rangle + \beta 0) \\
&+ |10\rangle(\alpha|0\rangle - \beta|1\rangle) + |11\rangle(\alpha|1\rangle - \beta|0\rangle)]
\end{aligned} \tag{1.13}
$$

This can be broken down into four terms. If alice performs a measurement and obtains 00, then Bob's qubit will be in the state $|\psi\rangle$.

$$
\begin{aligned}
00 &\to |\psi_3(00)\rangle \equiv [\alpha|0\rangle + \beta|1\rangle] \\
01 &\to |\psi_3(01)\rangle \equiv [\alpha|1\rangle + \beta|0\rangle] \\
10 &\to |\psi_3(10)\rangle \equiv [\alpha|0\rangle - \beta|1\rangle] \\
11 &\to |\psi_3(11)\rangle \equiv [\alpha|1\rangle - \beta|0\rangle]
\end{aligned} \tag{1.14}
$$

Depending on the output that Alice gets, Bob's qubit will end up in one of these four possible states. Now, Bob just needs to be told what the result of the measurement is and he can recover the state $|\psi\rangle$. The operation that Bob has to apply to his qubit is given by $Z^{M_1}X^{M_2}$. Thus the circuit is:

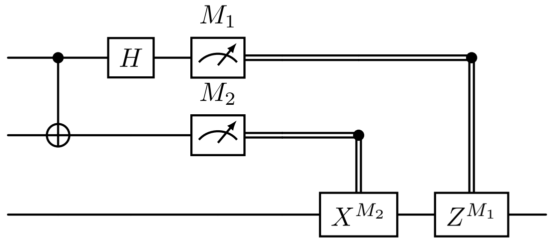


Figure 1.1: Quantum circuit for teleporting a qubit. The top two lines represent Alice's system, while the bottom line represents Bob's system.

By this protocol, a state $|\psi\rangle$ can be teleported from one place to another.

Chapter 2

Variational Quantum Algorithms

2.1 Cost Function

The cost function is a central component of Variational Quantum Algorithms (VQAs). It serves as a bridge between the quantum and classical parts of the algorithm, mapping the problem into a form that can be optimized using classical resources. The cost function evaluates the quality of a parameterized quantum state against a given problem, producing a scalar value that guides the optimization process. It is generally expressed as:

$$C(\theta) = f(\{\rho_k\}, \{O_k\}, \{U(\theta)\}) \tag{2.1}$$

Where:

- θ: Set of parameters to be optimized.
- ρ_k: Input states.
- O_k: Observables.
- $U(\theta)$: Parameterized quantum circuit.

A good cost function should be faithful, trainable, and efficiently estimable on quantum hardware. Its design ensures that the global minima correspond to optimal solutions for the problem at hand.

2.2 Ansatz

The ansatz refers to the structure of the parameterized quantum circuit used in a VQA. It is a trial wavefunction or quantum state that depends on a set of adjustable parameters. The ansatz defines the search space over which the optimization occurs.

2.2.1 Hardware Efficient Ansatz

The Hardware Efficient Ansatz is designed to minimize circuit depth by leveraging native gates and connectivity available on specific quantum hardware. It consists of layers of parameterized and unparameterized gates tailored to the physical qubit architecture. While this ansatz is highly versatile and adaptable to various quantum systems, it can suffer from trainability issues, such as barren plateaus, if not carefully initialized.

2.3 Gradient

Gradients are essential for optimizing the cost function in VQAs. They quantify how changes in the parameters affect the cost function's value and guide the optimization process towards the global minimum.

2.3.1 Parameter Shift Rule

The Parameter Shift Rule provides an efficient method to compute gradients on quantum hardware. For a parameter θ_l associated with a unitary gate, the gradient is given by:

$$\frac{\partial C}{\partial \theta_l} = \frac{1}{2\sin\alpha}\left(\mathrm{tr}[O_k U^\dagger(\theta_+)\rho_k U(\theta_+)] - \mathrm{tr}[O_k U^\dagger(\theta_-)\rho_k U(\theta_-)]\right) \tag{2.2}$$

$$\theta_\pm = \theta_l \pm e_l$$

Where:

- e_l is a unit vector corresponding to the parameter θ_l.

This rule avoids the need for finite difference approximations and is hardware-friendly, making it a cornerstone for gradient-based optimization in VQAs.

2.4 Optimizers

Optimizers are classical algorithms used to adjust the parameters of the quantum circuit to minimize the cost function. They play a crucial role in determining the efficiency and success of VQAs.

2.4.1 Gradient Descent Methods

Gradient Descent Methods are widely used in VQAs to iteratively minimize the cost function. These methods rely on gradient information to update parameters in the direction of the steepest descent. Key variants include:

- **Stochastic Gradient Descent (SGD)**: Efficient but noisy updates based on gradient estimates.
- **Adam Optimizer**: Adaptive step sizes for more efficient convergence.
- **Natural Gradient Descent**: Utilizes information geometry for more effective parameter updates.

Each method balances precision, computational cost, and resilience to noise in different ways.

2.5 Applications

VQAs have a broad range of applications, spanning fields like chemistry, optimization, and machine learning.

2.5.1 Variational Quantum Eigensolver (VQE)

The Variational Quantum Eigensolver (VQE) is one of the most prominent applications of VQAs, designed to approximate the ground state energy of a Hamiltonian. The cost function in VQE is expressed as:
Where $|\psi(\theta)\rangle$ is a parameterized quantum state and H is the Hamiltonian of the system.
VQE combines quantum state preparation with classical optimization to iteratively minimize the energy expectation value. It has been successfully applied in quantum chemistry, material science, and condensed matter physics, serving as a cornerstone algorithm for NISQ devices.

Chapter 3

Solving Nonlinear Differential Equations with Differential Quantum Circuits

In recent advancements in quantum computing, differentiable quantum circuits (DQCs) have emerged as a powerful tool for solving nonlinear differential equations. This chapter focuses on the mathematical formulations and key methodologies.

3.1 Boundary Handling

Boundary conditions are mathematically expressed as constraints applied to the differential equation's solution.

3.1.1 Pinned Boundary Handling

$$L_{\text{boundary}} = \eta \left(f_\theta(x_0) - u_0\right)^2 \tag{3.1}$$

Where:

- η is the pinning coefficient.
- x_0 represents the set of boundary points.
- μ_0 is the vector of boundary values.

3.1.2 Floating Boundary Handling

$$f(x) = f_b + \langle f_{\phi,\theta}(x)|\hat{C}|f_{\phi,\theta}(x)\rangle \tag{3.2}$$

$$f_b = u_0 - \langle f_{\phi,\theta}(x_0)|\hat{C}|f_{\phi,\theta}(x_0)\rangle \tag{3.3}$$

Where:

- f_b is iteratively adjusted to satisfy boundary conditions.

3.2 Optimization

Optimization focuses on minimizing boundary error:

$$L = L_{\text{diff}} + L_{\text{boundary}} \tag{3.4}$$

3.3 Cost Function Design

Cost functions are defined using Hermitian operators:

$$\hat{C} = \sum_j \alpha_j \hat{C}_j \tag{3.5}$$

Where:

- α_j are weights optimized during training.
- $\hat{C}_j$ are cost operators.

Examples:

- Single qubit magnetization: $\hat{C} = \hat{Z}_j$
- Total magnetization: $\hat{C} = \sum_j \hat{Z}_j$
- Ising Hamiltonian: $\hat{C} = \sum_j J_{j,j+1} \hat{Z}\hat{Z}_{j+1} + h_j^z \hat{Z}_j + h_j^x \hat{X}_j$
- Spin-glass Hamiltonian: $\hat{C} = \sum_{i<j} J_{i,j} \hat{Z}_i \hat{Z}_j + \sum_j h_j^z \hat{Z}_j$

3.4 Loss Functions

The loss function quantifies the deviation from the solution.

3.4.1 Mean Squared Error (MSE)

$$L_{\text{MSE}} = \frac{1}{M} \sum_{i=1}^{M} (f(x_i) - u(x_i))^2 \tag{3.6}$$

3.4.2 Mean Absolute Error (MAE)

$$L_{\text{MAE}} = \frac{1}{M} \sum_{i=1}^{M} |f(x_i) - u(x_i)| \tag{3.7}$$

3.5 Differential Equation Solving Framework

The framework minimizes the residual loss across grid points:

$$L_\theta = \sum_{i=1}^{M} L(F[d^m f/dx^m, f, x_i], 0) \tag{3.8}$$

Where:

- F represents the differential equation.
- m indicates the order of derivatives.

3.5.1 Variational Spectral Representation (VSR)

The trial solution is expressed as:

$$f(x) = \langle f_{\phi,\theta}(x) | \hat{C} | f_{\phi,\theta}(x) \rangle \tag{3.9}$$

3.6 Fluid Dynamics Applications

3.6.1 Navier-Stokes Equations

The Navier-Stokes system is solved using DQCs with:

$$\nabla \cdot \mathbf{u} = 0, \quad \frac{D\mathbf{u}}{Dt} = -\nabla p + \nu \nabla^2 \mathbf{u} \tag{3.10}$$

Where:

- $\mathbf{u}$ is the velocity vector.
- p is the pressure.
- ν is the kinematic viscosity.

3.7 Encoding Multiple Functions in Quantum Circuits

3.7.1 Parallel Encoding

Multiple functions are encoded either via:

1. Shared quantum register encoding:

$$\hat{C}_1, \hat{C}_2, \ldots, \hat{C}_n \tag{3.11}$$

2. Independent quantum register encoding with separate ansatze and cost functions.

Chapter 4

ETH Quantum Hackathon 2024 - Part I

4.1 The Problem

This week we are working on a problem from the ETH Quantum Hackathon 2024. The warmup round was done as an assignment for the 3rd week of this project. The problem in this week's assignment is making a DQC model which can approximate the following PDE in 2D:

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0 \tag{4.1}$$

which has an exact solution $u(x, y) = e^{-\pi x} \sin(\pi y)$ when solved with the Dirichlet boundary conditions in the domain $x, y \in [0, 1]$:

$$\begin{aligned} u(0, y) &= \sin(\pi y), \\ u(x, 0) &= 0, \\ u(X, y) &= e^{-\pi} \sin(\pi y), \\ u(x, Y) &= 0 \end{aligned} \tag{4.2}$$

The task is to make a DQC model that can approximate this PDE with the given boundary conditions using Qadence and evaluate the model's accuracy using Mean Squared Error (MSE).

4.2 Approach

This problem consists of 5 terms: 1 PDE and 4 boundary conditions. Each term contributes to the global loss function. We will be making a Quantum Neural Network (QNN) using the `QNN` class of the Qadence package. We will make a trainable DQC architecture and optimize the model using gradient-based methods to minimize the loss function.

4.3 Methodology

4.3.1 QNN Architecture

- **Feature Map:** The feature map uses the Chebyshev polynomials of the first kind for encoding along with a tower scaling reupload strategy.
- **Ansatz:** We will be using a Hardware Efficient Ansatz (HEA) of depth 8 for the model.

- **Observable:** The Z operator was chosen as the observable, scaled by 3 and shifted by -1.

4.3.2 Loss Function

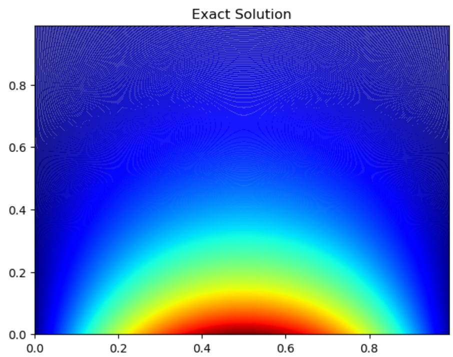
The loss function for this problem is the sum of the MSE loss for the PDE and the MSE loss for the boundary conditions.

4.3.3 Optimization

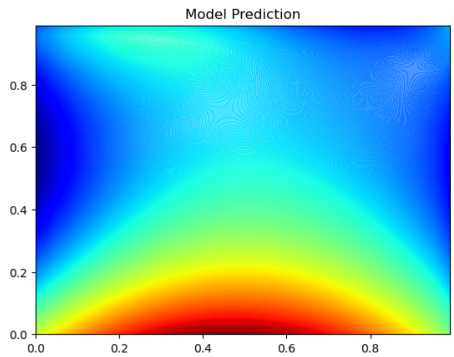
The Adam optimizer was used to optimize the model. The model was trained with a learning rate of 0.1 till the loss got to less than 0.15. A random seed of 42 was used for reproducibility.

4.4 Results

The model achieved consistent convergence with low MSE values, validating its ability to approximate the solution accurately. The model was trained for around 1000 epochs and the loss function was minimized to 0.15. The model was able to approximate the PDE and the boundary conditions with a good accuracy.



(a) Exact Solution



(b) Approximate Solution

The contour plots provide evidence of the model's ability to approximate the solution behaviour accurately. **Highlights:**

- The use of a physics-informed loss function effectively combined domain knowledge with quantum machine learning.
- The reproducibility and low MSE values validated the solution's reliability.

4.5 Conclusion

The implementation successfully demonstrated the capability of Differential Quantum Circuits for solving partial differential equations. The QNN model built using Qadence, provided an approximate solution for the 2D PDE.

Future Work:

- Investigate the impact of different feature maps and ansatz on the model's performance.
- Apply the model to other PDEs and compare the results.
- Experiment with Harmonic QNNs to improve convergence.