University of the Witwatersrand
Johannesburg

September 13, 2025

School of Physics

RESEARCH PROJECT

Course: PHYS4018A - Research Project: Physics-2024-SM2
Student Name: Kgothatso Ntumbe
Student Number: 2445026
Email Address: 2445026@students.wits.ac.za

Supervisor Name: Isaac Nape
Supervisor Email: isaac.nape@wits.ac.za

Contents

A Quantum Neural Network for Process Tomography

Abstract

This research investigates the application of Quantum Neural Networks (QNNs) to Quantum Process Tomography (QPT), a vital technique for characterizing quantum processes. While conventional Artificial Neural Networks (ANNs) excel in classical machine learning by approximating complex functions, we explore how their structure can be adapted to reconstruct quantum processes from measurement data. Given the exponential resource requirements for QPT as system size increases, we treat quantum tomography as a learning problem where a QNN maps input quantum states to corresponding measurement outcomes. Utilizing Variational Quantum Circuits (VQCs) with tunable quantum gates, we achieve quantum parallelism and enhanced computational efficiency in QPT. Our methodology involves training the QNN using a simulator and minimizing a cost function to quantify reconstruction errors, aiming to create a model that generalizes well across various quantum channels. This work bridges ANNs and QPT, highlighting the potential of QNNs to advance quantum technology and improve quantum process reconstruction efficiency.

Keywords: Quantum Neural Network (QNN), Quantum Process Tomography(QPT), Variational Quantum Algorithm (VQA), Variational Quantum Circuit(VQC).

1 Introduction

Artificial Neural Networks (ANNs), the foundation of classical machine learning, have demonstrated remarkable success in tasks like image recognition, natural language processing, and pattern recognition by approximating complex functions [1]. ANNs are composed of interconnected layers of artificial neurons that use weighted inputs and activation functions to learn relationships in data through a process of optimization, usually via backpropagation[2].

Inspired by this success, **QNNs** aim to perform similar learning tasks, but within the quantum realm. This project explores how the structure of ANNs can be adapted to the quantum domain to tackle the specific problem of reconstructing quantum processes from measurement data in QPT [3].It is essential for understanding how quantum devices and systems behave. By collecting measurement data from a series of quantum states and analyzing the results, QPT allows us to infer the complete behavior of a quantum channel. However, the number of measurements and the computational resources required to reconstruct the channel increase exponentially with the size of the system. [4]. This leads to the central research question: *Can we leverage QNNs to efficiently reconstruct unknown quantum processes through process tomography?*

The concept of QNNs builds on the success of **Variational Quantum Circuits (VQCs)**. In this context, we treat the process of quantum tomography as a learning problem, where the task of the QNN is to reconstruct the unknown quantum operator based on a series of tomographic measurements[5]. By training the QNN to map input states to measurement outcomes, the network can ”learn” the underlying quantum process, providing a scalable alternative to classical reconstruction algorithms.One key advantage of QNNs over traditional approaches is the potential for **quantum parallelism**. Through the principles of quantum superposition and entanglement, QNNs can process multiple quantum states simultaneously, leading to potential computational speedups [6]. This intrinsic parallelism is particularly appealing for QPT, where the task involves reconstructing a process from multiple quantum measurements. By leveraging the power of quantum gates as tunable parameters, the QNN can model a wide variety of quantum operations, offering a flexible and powerful framework for quantum process reconstruction[7].

A central aspect of the proposed QNN approach is the optimization of its parameters through **quantum-classical hybrid algorithms**, which combine quantum computing with classical optimization techniques[8]. In a QNN, quantum gates act as tunable parameters, analogous to the weights in an ANN. The goal is to optimize these parameters such that the QNN produces output measurements that match the true quantum process. This optimization process is akin to training a classical neural network and can be achieved using techniques like **quantum gradient descent**[9] .Moreover, the use of **quantum feature space mapping** is another critical advantage that QNNs offer[10]. Just as classical neural networks rely on mapping data into higher-dimensional feature spaces to capture complex relationships, QNNs can exploit quantum feature maps that leverage entanglement and superposition to map quantum states into a space where they can be more easily distinguished [11]. This allows the QNN to capture intricate relationships between different quantum states and their measurements, resulting in more accurate reconstructions of the quantum process.

Recent research in quantum machine learning has shown that QNNs, particularly those based on **Variational Quantum Algorithms (VQAs)**, can be effectively trained to solve a variety of quantum problems [12]. VQAs rely on the iterative optimization of a quantum circuit's parameters to minimize a predefined cost function, which in our case corresponds to the error in the reconstructed quantum process. By minimizing this cost function, we aim to create a QNN that generalizes well across different quantum channels, enabling it to perform accurate process tomography across a range of quantum systems [13].

To further align with the principles of classical neural networks, the proposed QNN architecture can be designed with multiple layers of quantum gates, each layer acting as a quantum analog of classical neural network layers [14]. Each layer of the QNN applies a sequence of parameterized quantum operations to the input states, gradually ”learning” the process through quantum measurements. The quantum gates in these layers can be adjusted to simulate different quantum operations, and through an optimization process, the QNN can refine its parameters to achieve accurate process tomography[15]. One of the main challenges in this project is to ensure that the QNN can generalize across different quantum processes rather than being tailored to a specific set of measurements[16]. Similar to how classical ANNs generalize from training data to unseen data, our QNN must be trained to learn a wide variety of quantum channels, ensuring that it can reconstruct any arbitrary quantum process based on a set of tomographic measurements[17].

To achieve the proposed goal, we will implement a QNN using a **Variational Quantum Circuit (VQC)** framework. The VQC will be parameterized by quantum gates such as rotation and entangling gates, which will be adjusted during the training process[18, 19]. Drawing inspiration from the **Variational Quantum Eigensolver (VQE)** and other quantum variational algorithms, we will optimize the parameters of the QNN using classical machine learning techniques [19]. The training process will involve a hybrid quantum-classical loop, where the quantum circuit is updated iteratively based on feedback from the cost function, which measures the accuracy of the quantum process reconstruction.

In the upcoming sections, we will explore the details of how this research bridges the world of artificial neural networks (ANNs) and quantum process tomography (QPT). First, we will provide an overview of ANNs, exploring their structure and how they have been traditionally applied to solve complex problems in classical computing. Following this, we will offer a thorough introduction to QPT, explaining its role in characterizing quantum processes and the challenges it presents as quantum systems grow larger. Next, we will establish the connection between ANNs and QPT, demonstrating how the principles of neural networks can be explored in the natural space of quantum realm through the use of Quantum Neural Networks (QNNs). This will lead us to the methodology, where we will unpack the tools and techniques employed, including the design and training of a QNN to efficiently reconstruct quantum processes. Finally, we will present and analyze the results, providing a deeper insight into the performance and potential of QNNs in solving quantum process tomography. Through this exploration, we aim to spark curiosity about the overlap of quantum computing and machine learning, offering a glimpse into the future of quantum technology.

2 Brief Overview

2.1 Artificial Neural Network

Artificial Neural Networks (ANNs) are models designed to mimic the structure and function of the human brain[20], enabling them to identify patterns and make decisions. They are made up of interconnected units known as neurons, which are arranged in layers. The standard ANN architecture includes three main types of layers: the input layer, one or more hidden layers, and the output layer, as illustrated in Figure 1.

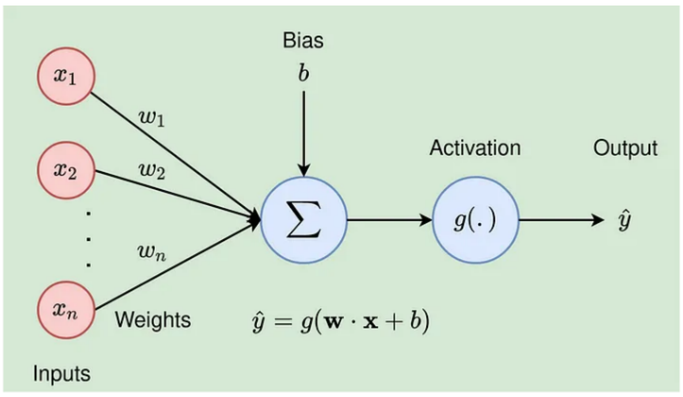


Figure 1: A neuron, a basic unit of artificial neural network[21]

By adjusting neuron weights during training, ANNs can learn patterns and relationships within data, making them effective for tasks like image recognition, natural language processing, and time series forecasting. The input layer takes in the data, while the hidden layers, which make up most of the network, handle the bulk of the processing by learning data features and representations[1]. The output layer provides the final result or prediction. The depth (number of layers) and width (neurons per layer) of the network—collectively called its architecture—play a key role in how well it captures data complexity..

Training an artificial neural network (ANN) involves an optimization process[1] where the network adjusts its weights to reduce a loss function, which reflects the difference between its predictions and the actual outcomes. This is typically done using gradient-based optimization techniques, such as stochastic gradient descent (SGD)[9] and its variations. In regression tasks, a common choice for the loss function is the Mean Squared Error (MSE)[22], which quantifies the average squared difference between predicted and true values.

$$L(y,\hat{y}) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{1}$$

where N is the number of data points,y_i is the true output, $\hat{y}_i$ is the predicted output by the network.The network's weights $\mathbf{w}$ are updated iteratively to minimize the loss function by using a gradient-based optimization algorithm such as Stochastic Gradient Descent (SGD).The update rule in SGD is[9]:

$$\mathbf{w} \leftarrow \mathbf{w} - \eta \nabla_{\mathbf{w}} L(y,\hat{y}) \tag{2}$$

where, $\mathbf{w}$ represents the weights (parameters) of the neural network, η is the learning rate(a small constant that controls the step size), $\nabla_{\mathbf{w}} L(y,\hat{y})$ is the gradient of the loss function with respect to the weights. In order to compute the gradient ,we use the backpropagation algorithm[2].It works by applying the chain rule of calculus to propagate the error backwards from the output layer to the input layer. For a simple neural network with one hidden layer, the output $\hat{y}$ is calculated as:

$$\hat{y} = f(\mathbf{w}^{(2)}\sigma(\mathbf{w}^{(2)}\mathbf{x} + b^{(2)}) + b^{(2)}) \tag{3}$$

where $\mathbf{x}$ is the input, $\mathbf{w}^{(2)}$ and $\mathbf{w}^{(2)}$ are the weights for the first and second layers, $\mathbf{b}^{(2)}$ are the biases,σ is the activation function, and f is the output activation function. The gradient of the loss function L with respect to each weight is computed using the chain rule for a weight in the second layer $\mathbf{w}^{(2)}$, the update rule:

$$\frac{\partial L}{\partial \mathbf{w}^{(2)}} = \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial \mathbf{w}^{(2)}} \tag{4}$$

Similarly, for the weights in the first layer $w^{(2)}$

$$\frac{\partial L}{\partial \mathbf{w}^{(2)}} = \frac{\partial L}{\partial \hat{y}} \cdot \frac{\hat{y}}{\partial \sigma} \cdot \frac{\partial \sigma}{\partial \mathbf{w}^{(2)}} \tag{5}$$

Once the gradients are computed, the weights are updated using the gradient descent rule. This process is repeated over many iterations (epochs) until the loss function converges to a minimum, allowing the network to learn the optimal set of weights for accurate predictions. The training process, therefore, relies on a combination of the loss function, gradient descent, and backpropagation to gradually adjust the weights, reduce error, and improve the network's overall performance.

Artificial Neural Networks (ANNs) have transformed numerous fields thanks to their ability to model complex, non-linear functions. They excel at generalizing from large datasets and have outperformed traditional algorithms in many areas. Recently, deep neural networks—ANNs with multiple hidden layers—have achieved significant advancements, especially in tasks like image and speech recognition. Their success is largely due to their hierarchical structure[20], where each layer learns progressively more abstract features from the input data..

Despite their success in classical machine learning, ANNs have limitations. As tasks become more complex and datasets grow larger, training ANNs can become computationally demanding, requiring vast amounts of data and computing resources. Additionally, traditional ANNs are based on classical computation, preventing them from leveraging the potential speed-ups that quantum computing offers. This is where Quantum Neural Networks (QNNs) come in, aiming to use quantum mechanics to address more complex challenges, such as those encountered in Quantum Process Tomography (QPT).

In the next section, we will dive into Quantum Process Tomography, a technique used to characterize quantum systems, and then explore how we can connect the principles of ANNs to QPT using QNNs.

2.2 Quantum Process Tomography

Quantum Process Tomography (QPT) is a method used to fully characterize quantum processes or channels, showing how an unknown quantum operation transforms input states[23]. It's crucial for analyzing and diagnosing the behavior of quantum systems like gates, circuits, and noisy channels. The main objective of QPT is to reconstruct the mathematical model of a quantum process using experimental data, offering insights into how it affects any given quantum state. To build a solid understanding of QPT, it's important to first explore the key concepts and mathematical structure behind it.

In quantum mechanics, a quantum state is typically represented by a density matrix (denoted as ρ), which provides a statistical description of the state of a quantum system. The density matrix can describe both pure states (where the system is in a definite state) and mixed states (where the system exists in a statistical mixture of different states)[24].A quantum process, or quantum channel, describes the evolution or transformation of these states. The mathematical framework used to describe such a process is called a Completely Positive Trace-Preserving (CPTP) map[25]. The completely positive property ensures that the channel preserves the positivity of the density matrix, which guarantees that the probabilities remain valid (i.e., non-negative) even when extended to a larger system. The trace-preserving condition ensures that the sum of probabilities across all possible outcomes remains 1, reflecting the conservation of total probability.

$$\rho_{out} = \varepsilon(\rho_{in}) \tag{6}$$

The aim of QPT is to reconstruct ε, which can be expressed in a variety of forms, such as the Kraus representation, the Choi matrix[26], or the process matrix (χ-matrix) representation. Each of these forms captures how a process acts on quantum states, and the goal of QPT is to estimate this matrix using measured data from the system.One of the most common mathematical representations of quantum processes is the Kraus representation. In this formulation, a quantum process ε can be described using a set of Kraus operators[27] K_i , where the process acts on a density matrix

$$\varepsilon(\rho) = \sum_i K_i \rho K_i^\dagger \tag{7}$$

where K_i represents the action of the process, and they must satisfy the condition:

$$\sum_i K_i^\dagger K_i = I \tag{8}$$

This condition ensures that the process is trace-preserving. The Kraus representation is flexible and can describe both unitary and non-unitary processes, making it a powerful tool for modeling noise and errors in quantum systems[27].Another widely used representation in QPT is the process matrix or χ-matrix formalism ,which offers a convenient way to express a quantum process in terms of a fixed operator basis. Given a set of basis operators E_i (e.g., Pauli matrices or Gell-Mann operators[26]), the process ε can be written as:

$$\varepsilon(\rho) = \sum_{i,j} \chi_{ij} E_i \rho E_j^\dagger \tag{9}$$

In this formulation, the matrix χ encodes all the information about the quantum process.The matrix χ is Hermitian and positive semi-definite, ensuring that the process is completely positive. Furthermore, the trace-preserving condition is enforced through constraints on χ.

The Choi matrix is another important mathematical tool used to describe quantum processes. The

Choi matrix. $J(\varepsilon)$ is obtained by applying the quantum process ε to one half of a maximally entangled state $|\Phi\rangle$[26]

$$J(\varepsilon) = (I \otimes \varepsilon)|\Phi\rangle\langle\Phi| \tag{10}$$

where I is the identity operation, and $|\Phi$ is the maximally entangled state,

$$|\Phi\rangle = \frac{1}{\sqrt{d}} \sum_i |i\rangle \otimes \langle i| \tag{11}$$

The Choi matrix provides a one-to-one correspondence between quantum processes and positive semi-definite matrices, making it useful in the analysis of quantum channels. For a process ε to be completely positive, its corresponding Choi matrix must be positive semi-definite. Additionally, for the process to be trace-preserving, the partial trace of the Choi matrix must satisfy:

$$Tr_{out}(J(\varepsilon)) = I_{in} \tag{12}$$

To reconstruct a quantum process through QPT, one must perform measurements on the system after the quantum process has been applied. These measurements are governed by the **Born rule**, which states that the probability of obtaining a particular outcome is related to the trace of the measured observable and the density matrix of the system[28]. By collecting these measurement probabilities, one can gather enough data to reconstruct the underlying quantum process.

According to the Born rule, the probability **p** of obtaining a particular measurement outcome associated with an observable **M** is given by:

$$p = Tr(M\rho) \tag{13}$$

For a quantum process, the measured probabilities are given by:

$$p = Tr(M\varepsilon(\rho)) \tag{14}$$

These measurement probabilities provide the data needed to reconstruct the process matrix χ (or the Kraus operators, depending on the representation). The goal of QPT is to collect enough measurement data to infer the complete action of the quantum process.

Limitations

As quantum systems grow in size and complexity, the challenge of characterizing quantum processes becomes significantly harder. In small systems with just a few qubits, it is relatively straightforward to apply QPT and determine the action of a quantum process. However, as the number of qubits in the system increases, the amount of data required for accurate process tomography grows[29] rapidly.For a system with $n-$qubits, the number of measurements required to fully characterize the quantum process scales exponentially, since the dimension of the process matrix grows as 4^n. This means that for every additional qubit, the size of the process matrix quadruples, making QPT extremely resource-intensive in larger quantum systems. Furthermore, the process of reconstructing the quantum channel involves solving large sets of equations based on these measurements, adding to the computational burden.

The primary challenge of traditional QPT is its exponential scaling problem[23]. As the system size increases, the number of parameters needed to describe the quantum process grows exponentially.For a system of n qubits, the process χ has $4^n \times 4^n$ elements, meaning that the number of measurements required to fully reconstruct the process becomes impractically large for even modest-sized quantum systems. This makes traditional QPT unsuitable for larger systems, as it would require an enormous number of experiments and significant computational resources to handle the data. For example, characterizing a process for a 10-qubit system would require reconstructing a matrix with over one million elements, and each of these elements would require data from multiple measurements. The complexity of performing and processing these measurements renders classical QPT infeasible for large-scale quantum systems[30].

As quantum systems scale up, the limitations of traditional Quantum Process Tomography (QPT) become more evident. The number of measurements needed increases exponentially, and reconstructing data becomes more complex, rendering classical QPT impractical for large systems. This challenge poses a significant barrier in advancing quantum technologies, as efficient characterization of larger quantum systems is crucial. To address these limitations, new methods are required that exploit quantum features like superposition and entanglement to perform more efficient and scalable tomography. Quantum Neural Networks (QNNs) offer a promising alternative[29]. By leveraging the learning capabilities of QNNs, it's possible to model quantum processes with fewer measurements and reduced computational demands. This approach could potentially scale without the exponential resource costs associated with classical QPT, providing a more practical solution for characterizing large quantum systems.

2.3 Quantum Process Representations

In quantum process tomography (QPT), a key challenge is to fully describe how a quantum process (or channel) transforms input states. While the Born rule and linear inversion focus on reconstructing the process from measurement probabilities[28], other representations such as the **Kraus**, **Choi**, and **Process Matrix (χ-matrix)** formulations offer alternative approaches. These representations are particularly useful for tasks such as error modeling, analyzing entanglement, or characterizing gate operations. Below, we describe each representation and its relevance, followed by examples that illustrate their utility for those who might not rely on the Born rule.

2.3.1 Kraus Representation

The **Kraus representation** expresses a quantum process as a set of operators, called Kraus operators $\{K_i\}$, which act on the density matrix of a quantum state. A quantum process $\varepsilon(\rho)$ is represented as:

$$\varepsilon(\rho) = \sum_i K_i \rho K_i^\dagger \tag{15}$$

These Kraus operators satisfy the completeness relation $\sum_i K_i^\dagger K_i = I$, ensuring that the process preserves the trace of the density matrix (i.e., probability conservation).

Example: Modeling Quantum Noise

The Kraus representation excels when modeling noise in quantum systems. Consider amplitude damping, a process where a qubit decays from an excited state $|1\rangle$ to the ground state $|0\rangle$[31] over time, which is common in superconducting qubits.

The Kraus operators for amplitude damping are:

$$K_0 = \begin{pmatrix} 1 & 0 \\ 0 & \sqrt{1-\gamma} \end{pmatrix}, \quad K_1 = \begin{pmatrix} 0 & \sqrt{\gamma} \\ 0 & 0 \end{pmatrix}$$

where γ is the damping parameter (decay probability).

Unlike the Born rule, which focuses on probability outcomes for specific measurements, the Kraus representation allows one to directly describe how the qubit state evolves due to noise[27]. This is particularly relevant when simulating noisy quantum channels in quantum error correction, where the goal is to model and mitigate errors rather than just predict measurement probabilities.

For example, if a qubit initially in state $\rho = |1\rangle|1\rangle$ undergoes amplitude damping[31], the process gives:

$$\varepsilon(\rho) = K_0\rho K_0^\dagger + K_1\rho K_1^\dagger = \begin{pmatrix} 1-\gamma & 0 \\ 0 & \gamma \end{pmatrix}$$

This describes the evolution of the qubit state under noise, showing how it gradually decays to 0 with increasing γ.

2.3.2 Choi Matrix Representation

The **Choi matrix** provides a compact way to represent a quantum channel by mapping it to a matrix. This representation is especially useful when analyzing quantum processes in the context of entanglement or when working with bipartite systems[32, 23]. The Choi matrix $J(\varepsilon)$ of a quantum process ε is defined by applying the process to half of a maximally entangled state Φ:

$$J(\varepsilon) = (I \otimes \varepsilon)(|\Phi\rangle\langle\Phi|) \tag{16}$$

where $|\Phi\rangle = \frac{1}{\sqrt{d}}\sum_i |i\rangle \otimes |i\rangle$ is a maximally entangled state, and I is the identity matrix.

Example: Analyzing Quantum Channels

Consider a quantum communication protocol where Alice sends part of an entangled pair to Bob through a noisy channel. The channel could affect the entanglement between the particles, which impacts the success of quantum teleportation or key distribution.

For example, if Alice and Bob share the maximally entangled state:

$$|\Phi\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}},$$

and Bob's qubit is transmitted through a noisy channel ε, the Choi matrix representation $J(\varepsilon)$ encodes how the channel acts on the entanglement[33].

In this case, using the Born rule would only give probabilities of measurement outcomes, but the Choi matrix captures entanglement-preserving or entanglement-breaking properties of the channel. It allows a researcher to determine how much entanglement survives after transmission and whether the channel is useful for entanglement-based protocols.

If the channel is depolarizing with Kraus operators[34] $K_0 = \sqrt{1-p}I$, $K_1 = \sqrt{p/3}X$, $K_2 = \sqrt{p/3}Y$, and $K_3 = \sqrt{p/3}Z$, the Choi matrix describes how this noise affects the overall state. If entanglement is reduced, the Choi matrix gives insights into how severe the decoherence is.

2.3.3 Process Matrix (χ-matrix) Representation

The **χ-matrix representation** expresses a quantum process in terms of a fixed operator basis $\{E_i\}$, such as the Pauli basis. The process $\varepsilon(\rho)$ is written as:

$$\varepsilon(\rho) = \sum_{i,j} \chi_{ij} E_i \rho E_j^\dagger \tag{17}$$

where χ_{ij} is the process matrix (χ-matrix), which contains all the information about the process, and $\{E_i\}$ are typically Pauli operators.

Example: Quantum Gate Characterization

When characterizing a quantum gate (e.g., a single-qubit Hadamard gate H), the goal is to determine how closely the physical implementation of the gate matches the ideal one. Any deviation in the operation can be attributed to errors such as phase noise, bit flips, or amplitude damping.

The χ-matrix for the ideal Hadamard gate is:

$$\chi(H) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0.5 & 0.5 & 0 \\ 0 & 0.5 & 0.5 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

where the basis is $\{I, X, Y, Z\}$.

In practice, gates are not perfect, and their actual χ-matrix may differ from the ideal. The χ-matrix provides a complete description of how the gate deviates from its intended operation. This information is crucial in quantum gate benchmarking (like randomized benchmarking or gate set tomography) because it reveals the type and magnitude of errors. The Born rule would only give outcome probabilities, but the χ-matrix gives a more granular picture of the gate's performance, allowing for detailed error correction strategies.

For example, if an experimental Hadamard gate has a χ-matrix:

$$\chi_{exp}(H) = \begin{pmatrix} 0.98 & 0.01 & 0.01 & 0 \\ 0.01 & 0.49 & 0.50 & 0.01 \\ 0.01 & 0.50 & 0.49 & 0.01 \\ 0 & 0.01 & 0.01 & 0.02 \end{pmatrix}$$

it indicates small errors in the X and Y components, suggesting that the gate might be subject to phase errors or coherent errors that require correction.

2.3.4 Born rule of Linear inversion

To reconstruct this map, we use a set of known input states, apply the process to these states, and perform measurements on the resulting output states. The **Born rule** governs the probability of obtaining measurement outcomes, and **linear inversion** is one method for reconstructing the process from these measurements.

A quantum process $\mathscr{E}$ acting on a quantum state ρ can be written in the **operator-sum representation** as:

$$\mathcal{E}(\rho) = \sum_{m,n} \chi_{mn} E_m \rho E_n^\dagger$$

where:

- $\{E_m\}$ is a fixed set of basis operators (usually chosen as Pauli matrices or another operator basis),
- χ_{mn} are the elements of the process matrix χ (which we aim to reconstruct),
- ρ is the input quantum state.

The goal of QPT is to determine the matrix χ, which fully characterizes the process $\mathcal{E}$.

2. Measurement Probabilities via Born Rule

For each input state ρ_k that is sent through the quantum process $\mathcal{E}$, we perform measurements described by a set of measurement operators M_j. The Born rule provides the probability $P(j|k)$ of obtaining the outcome corresponding to M_j when the system is in state $\mathcal{E}(\rho_k)$:

$$P(j|k) = \text{Tr}(M_j \mathcal{E}(\rho_k))$$

Substituting the operator-sum representation of the process $\mathcal{E}$, we have:

$$P(j|k) = \text{Tr}\left(M_j \left(\sum_{m,n} \chi_{mn} E_m \rho_k E_n^\dagger\right)\right)$$

Using the linearity of the trace, this can be rewritten as:

$$P(j|k) = \sum_{m,n} \chi_{mn} \text{Tr}\left(M_j E_m \rho_k E_n^\dagger\right)$$

This equation relates the measurement probabilities $P(j|k)$ to the unknown process matrix χ.

3. Linear Inversion Method

The linear inversion method reconstructs the process matrix χ from the measured probabilities. The measurement probabilities $P(j|k)$ form a system of linear equations in the elements of χ_{mn}. To see this explicitly, let's express the probabilities in matrix form.

Step 1: Measurement Setup

Choose a set of input states $\{\rho_k\}$, which are typically tomographically complete, meaning they span the space of density matrices. For a single qubit, common choices are the states $|0\rangle, |1\rangle, |+\rangle, |-\rangle, |+i\rangle, |-i\rangle$. Perform a set of measurements described by operators $\{M_j\}$, such as projectors onto the computational basis states, or Pauli measurements.

Step 2: Probability Expression in Matrix Form

For each input-output pair, the Born rule gives a probability $P(j|k)$. These probabilities can be written as a vector $\mathbf{P}$ of length $N \times M$, where N is the number of input states and M is the number of measurement outcomes.

Define a vector **p** containing the probabilities $P(j|k)$ for all combinations of j and k. This vector can be expressed as:

$$\mathbf{p} = \mathbf{A} \cdot \chi$$

where:

- **p** is the vector of measured probabilities,
- χ is the vectorized process matrix (reshaped into a vector from its original matrix form),
- **A** is a matrix that encodes the known input states $\{\rho_k\}$, measurement operators $\{M_j\}$, and basis operators $\{E_m, E_n\}$.

The matrix elements $A_{jk,mn}$ are given by:

$$A_{jk,mn} = \mathrm{Tr}(M_j E_m \rho_k E_n^\dagger)$$

This matrix **A** is known, as it depends on the choice of input states, measurement operators, and basis operators. The only unknown is the vector χ, which contains the elements of the process matrix.

Step 3: Linear Inversion

To solve for the process matrix χ, one inverts the matrix equation:

$$\chi = \mathbf{A}^{-1} \cdot \mathbf{p}$$

If **A** is square and invertible, this gives the process matrix directly[28]. In practice, **A** may not be square or invertible, so one typically solves the system using a **least-squares fit**[28]:

$$\chi = (\mathbf{A}^\dagger \mathbf{A})^{-1} \mathbf{A}^\dagger \mathbf{p}$$

This approach provides the best estimate of χ in the case of noisy or incomplete data.In this project, we use the Born rule of linear inversion to uncover the unknown parameterized unitary operator that we are interested in analyzing

2.4 Connecting ANNs to QPT through Quantum Neural Networks (QNNs)

In quantum computing, applying artificial neural network concepts to quantum systems results in Quantum Neural Networks (QNNs)[35]. Though they share the basic principle of learning from data, QNNs take advantage of quantum mechanical phenomena, including superposition, entanglement, and quantum gates as explained in the first few sections, to handle quantum information in a way that classical neural networks cannot. This adaptation allows QNNs to harness the distinct capabilities of quantum mechanics to enhance the processing and manipulation of data in quantum algorithms.

To adapt ANNs for quantum systems, we **replace classical computations with quantum operations**. Classical data can be encoded into quantum states through a process called quantum feature mapping[36], while the layers of neural networks are implemented using quantum circuits, as outlined in the two figure below, figure 2 and 3.

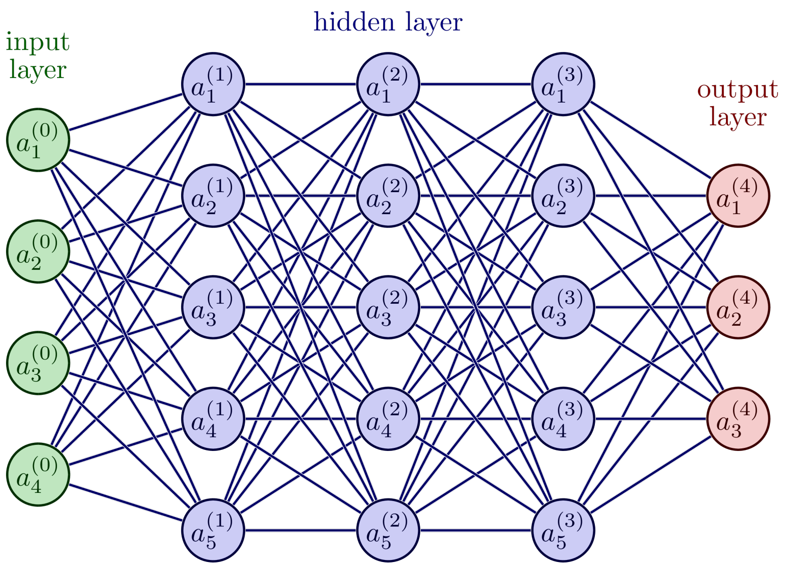


Figure 2: An artificial neural network (ANN) architecture[37]

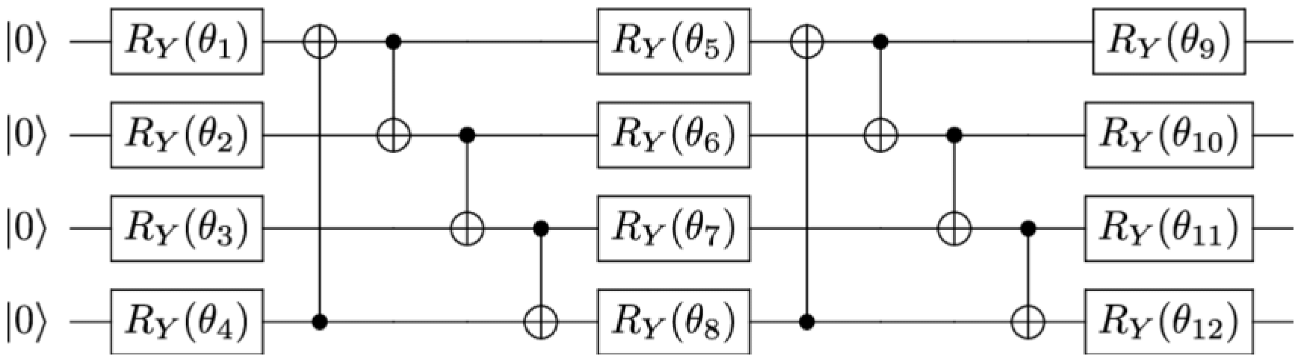


Figure 3: An example of a variational quantum circuit(VQC) with parameterized quantum rotation gates.][38]

In the language of QPT, VQCs can be used to model quantum channels by training the circuit to reproduce the output measurements for a given set of input states[5]. The process of adjusting the gate parameters is similar to training a classical neural network[35], where the objective is to minimize the difference between the circuit's predictions and the actual measured outputs. VQCs provide a flexible and powerful framework for implementing QNNs, as they can be optimized to represent a wide range of quantum processes.

One of the most significant advantages of QNNs over classical neural networks is the ability to leverage quantum parallelism. Quantum parallelism arises from the principle of superposition, allowing a quantum computer to process many possible input states simultaneously[39]. This capability can dramatically reduce the time and resources required for QPT.

In traditional QPT, each input state must be measured individually, resulting in a large number of measurements for systems with many qubits. However, with quantum parallelism, a QNN can process multiple states in parallel, reducing the total number of operations. This efficiency, combined with the QNN's ability to learn the behavior of a quantum process, makes it a powerful tool for characterizing complex quantum channels.

VQCs serve as a bridge between ANNs and QPT. VQCs are parameterized quantum circuits (analogous to ANN layers) that can be optimized using classical optimization techniques[5]. For QPT, VQCs allow us to train quantum circuits to predict the behavior of quantum processes, performing the same role as ANNs but within the quantum space. This creates an elegant link between classical neural networks and quantum process learning.

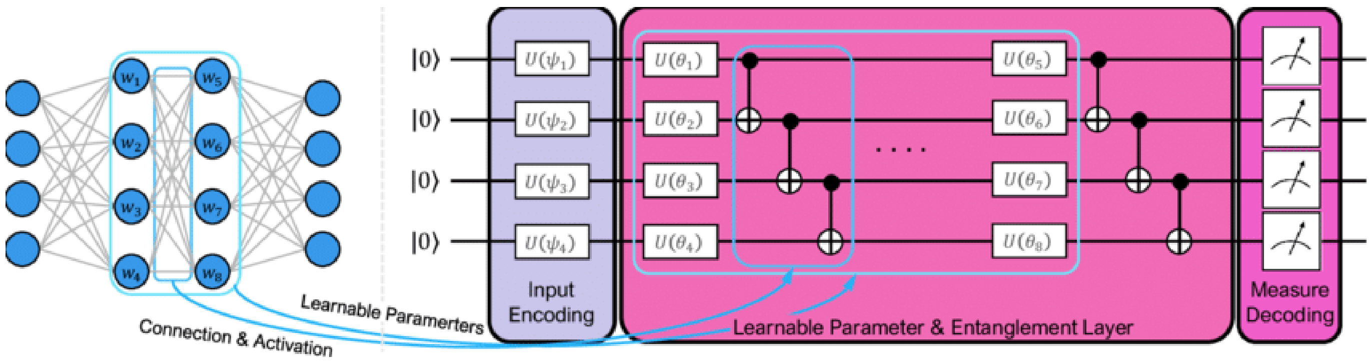


Figure 4: This diagram highlights the procedural similarity between Artificial Neural Networks (ANN) and Quantum Process Tomography (QPT). It demonstrates how both frameworks follow analogous steps in terms of training, optimization, and evaluation. In both cases, an input is processed through a series of transformations—whether classical layers in ANNs or quantum gates in QPT—to produce an output. The diagram illustrates the parallel structures and methodologies used in classical and quantum systems, emphasizing how concepts from ANNs can be adapted to solve problems in QPT.[40]

This concept of connectivity can be analogously related to the notion of the "Black Box" in modern artificial neural networks. In the context of Variational Quantum Algorithms (VQAs), the variational quantum circuit serves as the black box. Our objective is to determine the parameters that reveal the underlying mechanisms of this black box, in this case represented by figure 3. In this analogy, Alice inputs data into the black box, and Bob receives the corresponding outputs, which are generated based on the internal processing of the circuit.

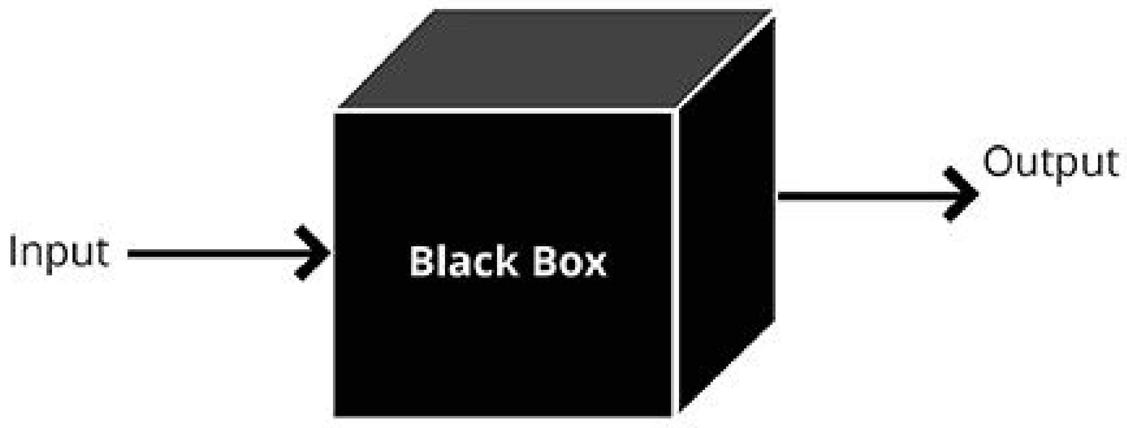


Figure 5: A visual illustration of the Black Box: Inputs (Alice) Transform into Outputs(Bob).

Anology: Consider an artificial neural network (ANN) where we start with a set of labeled images of cats and dogs. Although we know the inputs and their corresponding labels, the exact mechanisms by which the hidden layers identify these images remain unclear. The goal, then, is to uncover the characteristics and the unknown parameters of these hidden layers—essentially demystifying the black box. This process mirrors the objectives in quantum neural networks (QNNs), where we seek to understand the internal workings of the variational quantum circuit by analyzing how it processes quantum states and generates meaningful probability measurements, so the black box is the parameterized unitary operator the states go through.

3 Methodology

The primary goal of this section is to outline how the Quantum Neural Network (QNN) is constructed, from state preparation to process reconstruction. The QNN is designed to take quantum states as inputs and output the reconstructed quantum process, leveraging variational quantum circuits (VQC) and hybrid quantum-classical algorithms to train the model. In the following subsections, we will go through the theoretical foundation and practical steps involved in the implementation.

3.1 State preparation

The choice of quantum states for process tomography is a critical part of the methodology. In general, for Quantum Process Tomography (QPT), we must prepare a set of input states that span the Hilbert space of the system under study[41]. For single-qubit systems, this means preparing a collection of quantum states that form a complete basis for any quantum process acting on a qubit. Any single-qubit quantum state can be represented using two angles, the **polar angle θ** and the **azimuthal angle** ϕ on the bloch sphere, outlined in details mathematically by the universal single qubit equation[41]:

$$|\psi\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + e^{i\phi}\sin\left(\frac{\theta}{2}\right)|1\rangle \tag{18}$$

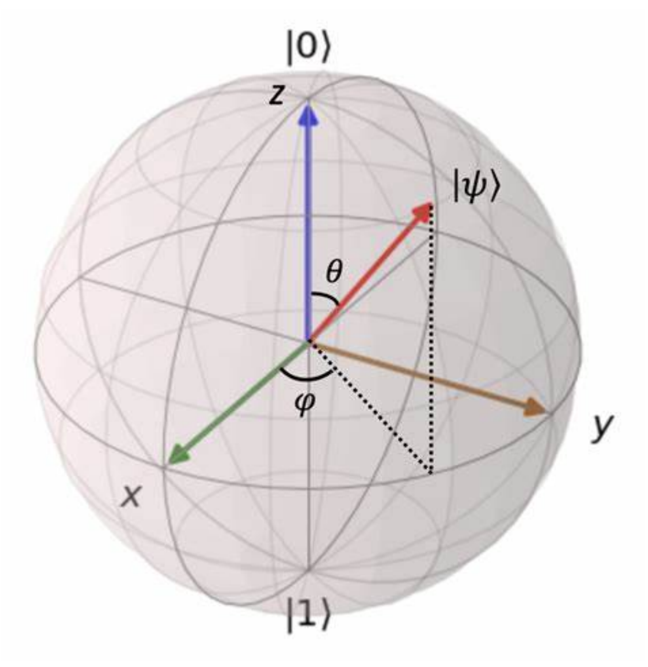


Figure 6: A visual of a single qubit state on the Bloch sphere shows the polar angle θ, which rotates the qubit along the z-axis, and the azimuthal angle ϕ, which rotates it in the x-y plane. $|0\rangle$ and $|1\rangle$ represent the computational basis states.[42]

This representation shows that any pure single-qubit state can be mapped onto a point on the Bloch sphere, where θ and ϕ control the state's location. For instance, with multiple varying θ and ϕ:

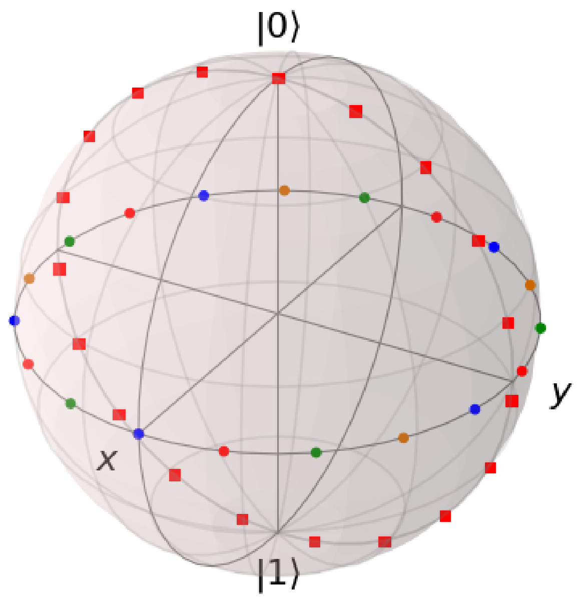


Figure 7: A representation of multiple single-qubit states on the Bloch sphere, verifying the accuracy of equation (15).Each point on the Bloch sphere represent a single qubit at different θ and ϕ.

we require a complete set of linearly independent input states (an over-complete set)that will fully capture the behavior of the quantum process. For a single-qubit system, these can include:

$$|0\rangle \tag{19}$$

$$|1\rangle \tag{20}$$

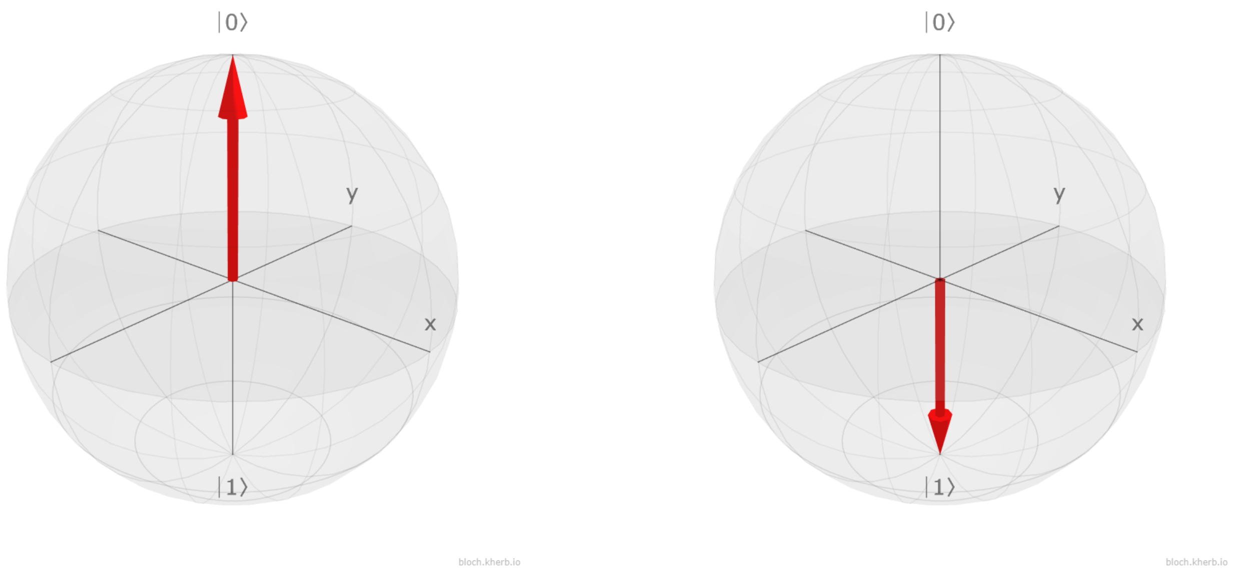
$$|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \tag{21}$$

$$|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \tag{22}$$

$$|+i\rangle = \frac{1}{\sqrt{2}}(|0\rangle + i|1\rangle) \tag{23}$$

$$|-i\rangle = \frac{1}{\sqrt{2}}(|0\rangle - i|1\rangle) \tag{24}$$

These states span the full space of a single qubit and encompass multiple bases, such as the computational, Hadamard, and phase bases[43]. According to the universal single qubit state equation in 18, one can visualize the states in 19



(a) (b)

Figure 8: A visual representation of the states on the Bloch sphere. (a) State $|0\rangle$ produced by choosing the parameters $\theta = 0$ and $\phi = n * \pi$ is on the left side, while on the right side (b) State $|1\rangle$ produced by setting $\theta = \pi$ and $\phi = n * \pi$, where n is an integer

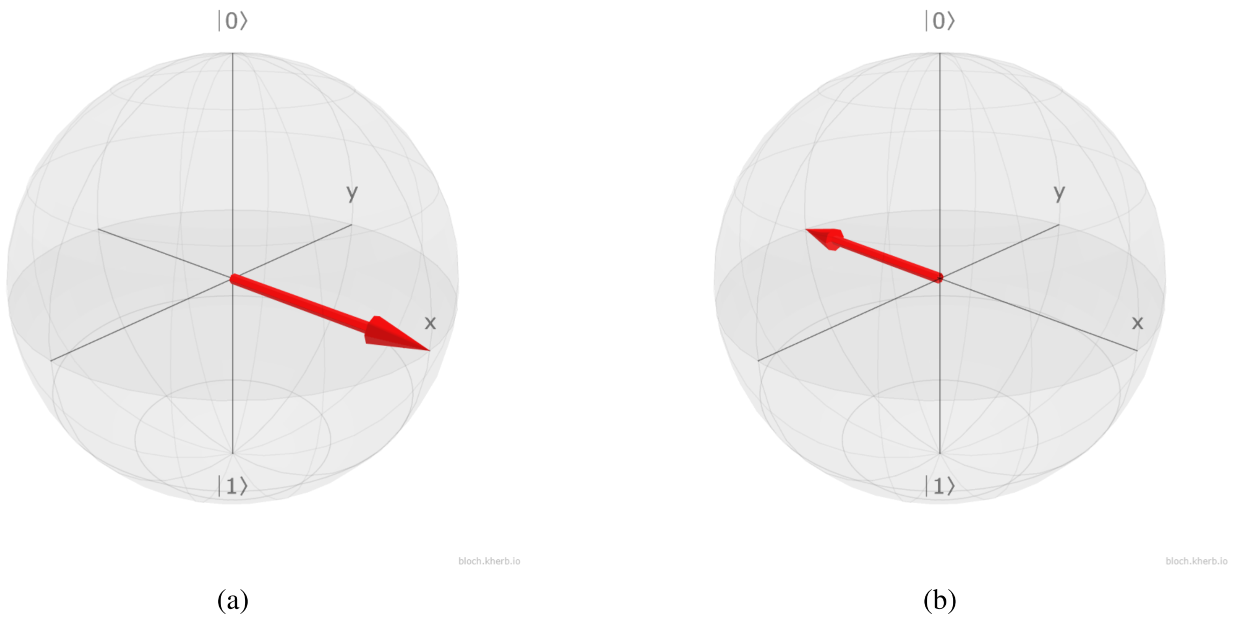


Figure 9: A visual representation of the states on the Bloch sphere.(a) State $|+\rangle$ produced by setting $\theta = \frac{\pi}{2}$ and $\phi = n * \pi$, where n is an integer. (b) State $|-\rangle$ produced by setting $\theta = -\frac{\pi}{2}$ and $\phi = n * \pi$, where n is an integer

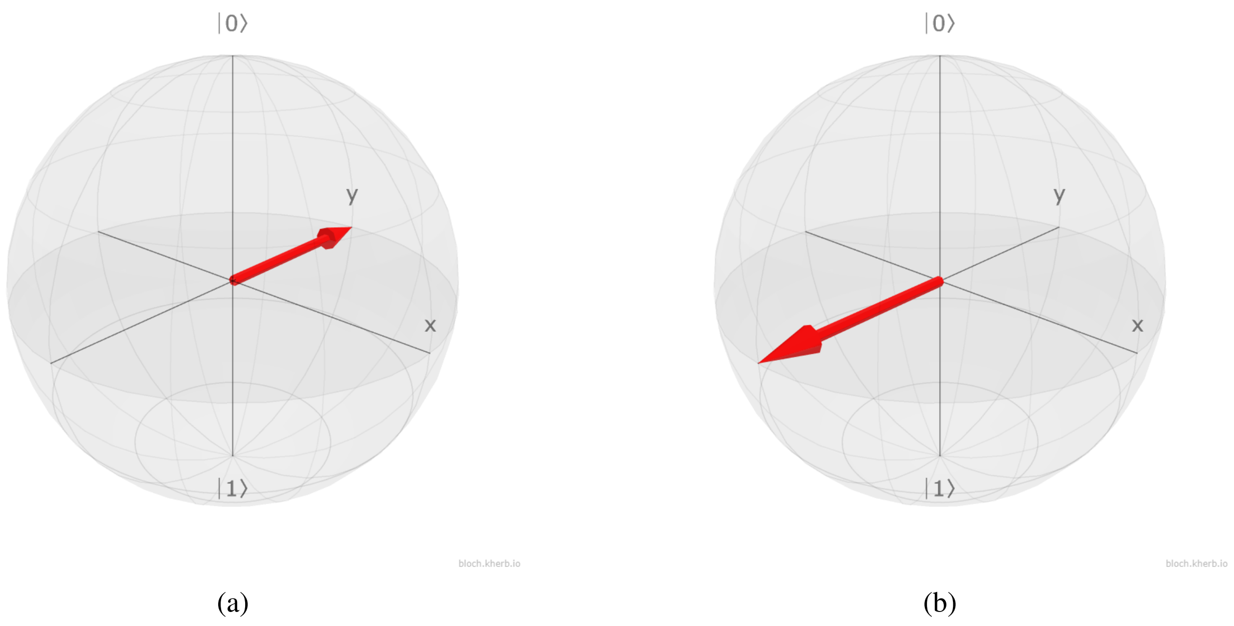


Figure 10: A visual representation of the states on the Bloch sphere.(a)State $|+i\rangle$ produced by setting $\theta = \frac{\pi}{2}$ and $\phi = \frac{\pi}{2}$, where n is an integer. (b) State $|-i\rangle$ produced by setting $\theta = \frac{\pi}{2}$ and $\phi = \frac{\pi}{2}$, where n is an integer

However, not all of these states are necessary to fully characterize a qubit or a quantum process acting on it. For a single qubit, the space it inhabits is two-dimensional, which means that to fully describe it, we only need a few independent measurements[24]. Specifically, for a single qubit, a complete description can be achieved by using just four states ($|0\rangle, |1\rangle, |\pm\rangle, |\pm i\rangle$).

These states correspond to measurements along the three orthogonal axes **X, Y, and Z** of the qubit's Bloch sphere representation[41]. Since a qubit's state can be fully described by measurements along these three axes, using just four states provides sufficient information to reconstruct the density matrix of the qubit and, consequently, fully characterize the quantum process. The point of adding additional states (the two remaining) of orthogonal states provides an extra information, but they are not strictly necessary. The information they provide is redundant in the context of quantum mechanics because the four chosen states already span the qubit's state space.

To prepare these quantum states in the language of quantum computing, we construct a quantum circuit using the Qiskit framework. Specifically, the states are initialized by applying a sequence of quantum gates to an input qubit(where the input qubit in this case is the Qiskit default qubit state $|0\rangle$). The primary tools for this task are rotation gates, namely R_y and R_z, which act to rotate the qubit's state vector on the Bloch sphere[41]. By adjusting the angles of these gates, we can generate a wide variety of quantum states, including superposition and phase-shifted states.

$$R_y(\theta) = \begin{bmatrix} \cos\left(\frac{\theta}{2}\right) & -\sin\left(\frac{\theta}{2}\right) \\ \sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{bmatrix} \tag{25}$$

$$R_z = \begin{bmatrix} e^{-i\frac{\phi}{2}} & 0 \\ 0 & e^{i\frac{\phi}{2}} \end{bmatrix} \tag{26}$$

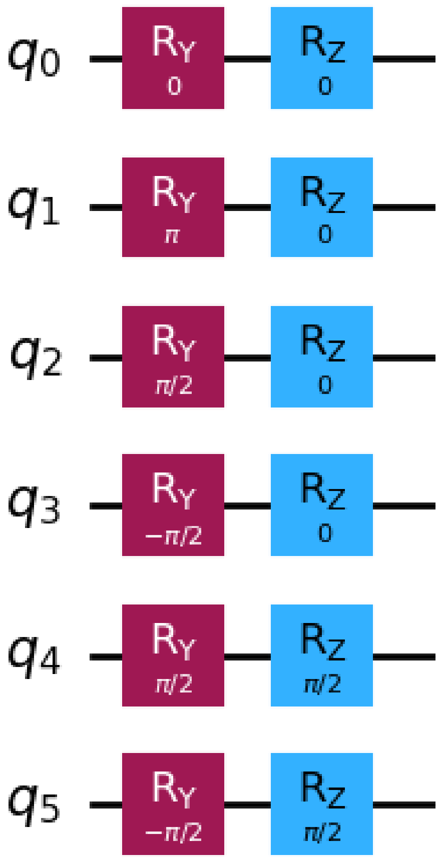


Figure 11: A visual representation of 6 qubit states each prepared by applying the rotation gates $R_y(\theta)$ and $R_z(\phi)$

To check if our quantum circuit produces the desired states ($|0\rangle, |1\rangle, |+\rangle, |-\rangle, |+i\rangle, |-i\rangle$) a suggestive approach is a fidelity check[32]. Fidelity compares the output state of the quantum circuit to the theoretical target state. It measures how close the two states are, with a fidelity value of 1 indicating perfect overlap. This is a widely used and effective method, as you are already employing it.

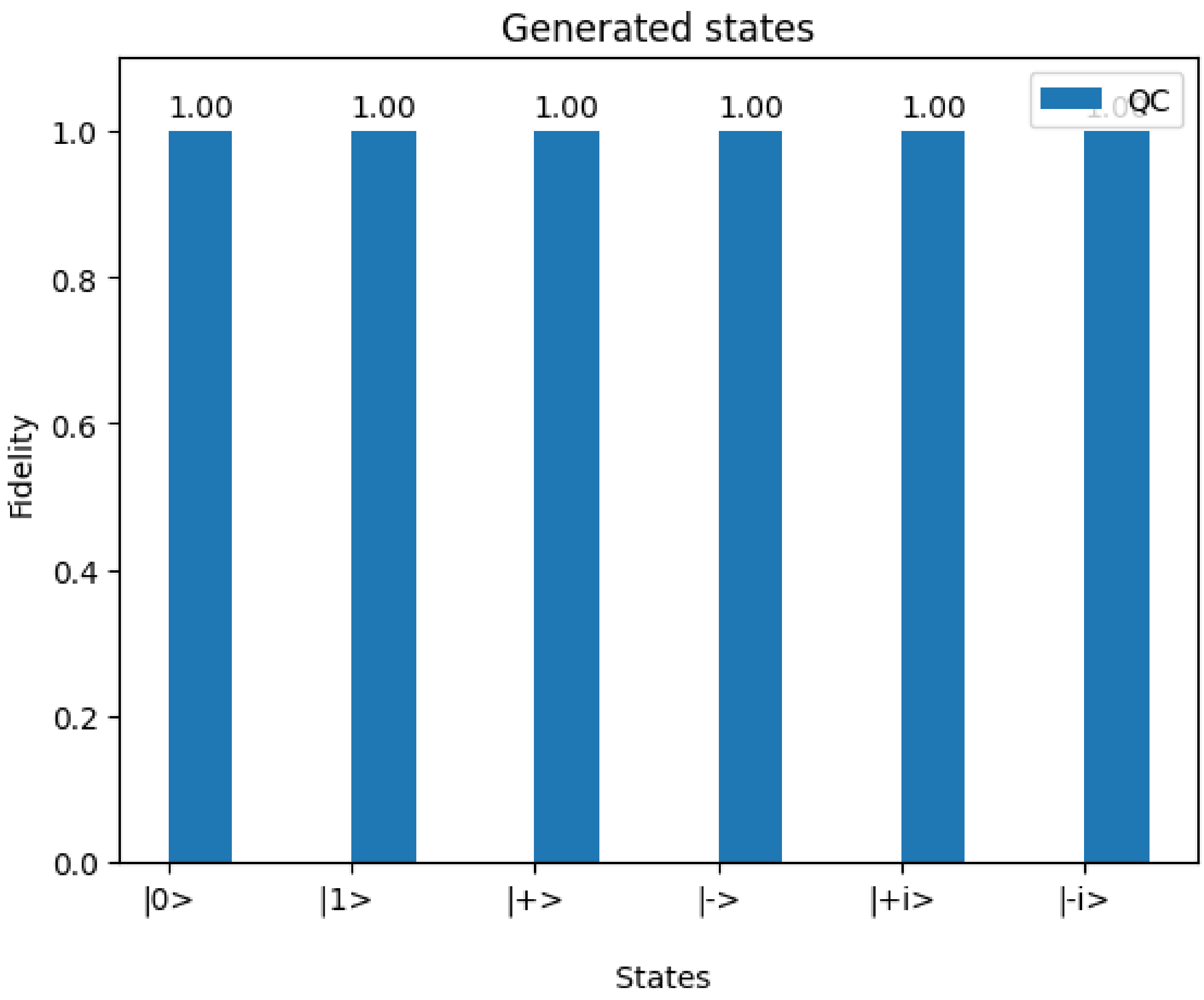


Figure 12: The fidelity plot demonstrating the accuracy of the quantum circuit in preparing the six specific states.The high fidelity values (fidelity= 1) indicate that the circuit is effectively reproducing the desired quantum states, ensuring that the preparation process aligns closely with the theoretical predictions

3.2 Operator Interaction

The next step is to take the chosen quantum states we have prepared and allow them to interact with a known operator of interest. This involves sending the prepared to the operator to act on them and give out an output state. This is followed by:

$$|\psi\rangle_{out} = U|\psi\rangle_{in} \tag{27}$$

where U can be any quantum operator, such as the Pauli operators (X, Y, and Z) or any single qubit unitary operator[43]. The following figure shows a quantum circuit with input states interacting with a known operator, the Pauli-X gate as seen in figure 8.

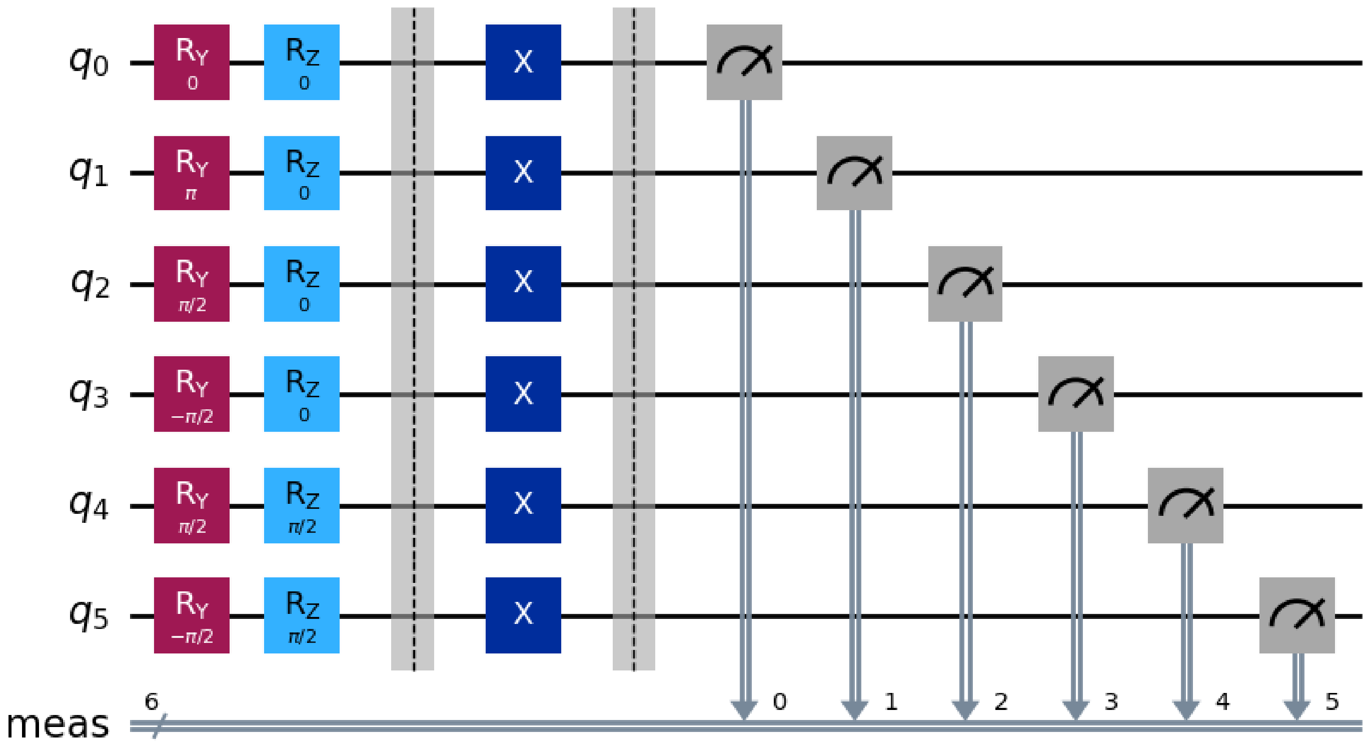


Figure 13: A quantum circuit representing the interaction of the prepared states shown in figure 7 interacting with the Pauli-X gate

Mathematical representation of this procedure, using the generic equation in equation 27

$$|\psi\rangle_{out} = U|\psi\rangle_{in} \tag{28}$$

$$|\psi\rangle_{out} = X|\psi\rangle_{in} \tag{29}$$

$$|\psi\rangle_{out} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} |\psi\rangle_{in} \tag{30}$$

where $|\psi\rangle_{in}$ is the input states in $(|0\rangle, |1\rangle, |+\rangle, |-\rangle, |+i\rangle, |-i\rangle)$

3.3 Tomographic measurements

In quantum process tomography (QPT), tomographic measurements are essential for reconstructing the quantum process or channel that acts on a quantum system.These measurements provide the data necessary to infer the behavior of the quantum channel. To compute the tomographic measurements, we infer the same procedure undertaken in quantum state tomography (QST)[32].Quantum state tomography is the process of reconstructing the quantum state of a system by performing measurements on multiple copies of the system in different bases[23]. For a single qubit system, these bases typically include the computational basis $(|0\rangle, |1\rangle)$,the superposition basis $(|+\rangle, |-\rangle)$, and the diagonal basis $(|+i\rangle, |-i\rangle)$.By measuring the probabilities associated with these states after they pass through the quantum process, we gather the necessary information to reconstruct the state. To measure the output after applying the quantum process(or operator), we perform projective measurements in the same basis as the prepared states[28]. The chosen set of projectors are $proj_i = |m_i\rangle\langle m_i|$, where m_i is in the set of all the measurement basis.

The outcome of each measurement is expressed as a set of probabilities that describe the projection of the final state onto the measurement basis. Mathematically, for each input state $|\psi_{in}\rangle$ prepared by the circuit, we measure its projection onto a set of output states$\{|m_i\rangle\}$, which results in a probability distribution:

$$P(m_i|\psi_{in}) = |\langle m_i|\varepsilon(\psi_{in})\rangle|^2 \tag{31}$$

$$= |\langle m_i|\psi_{out}\rangle|^2 \tag{32}$$

In this part, ε represents the quantum process under examination, while $|m_i\rangle$ denotes the states of the measurement basis. By gathering the probabilities obtained from all input states and measurement bases, we create distinct 'fingerprints' of the channel traversed. This information is essential for reconstructing the quantum process.

Following the interaction of the prepared states with a quantum process or operator (specifically, the Pauli-X operator) depicted in Figure 15, the resulting tomographic measurements are presented in the figure below.

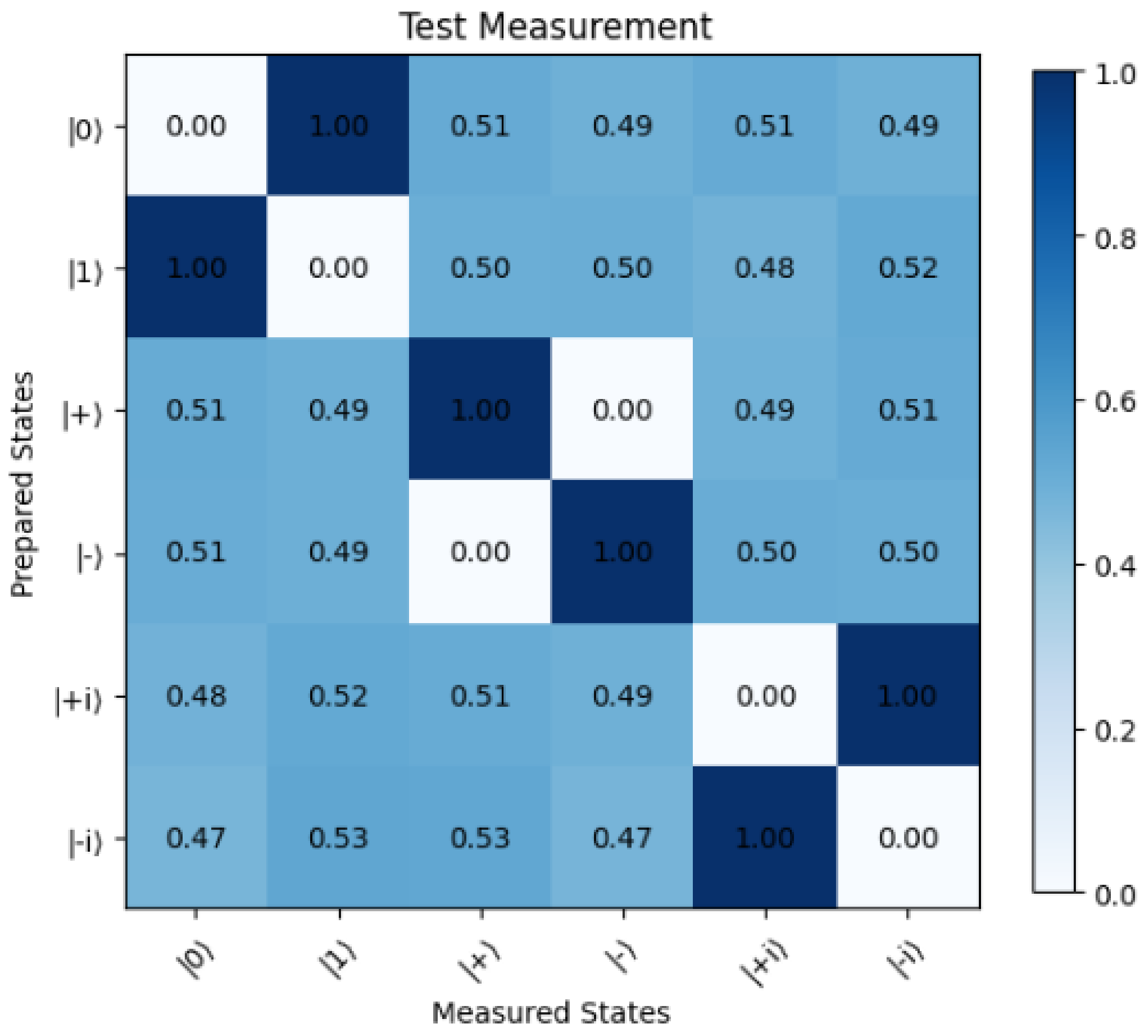


Figure 14: The tomographic measurements after the prepared states interacted with the Pauli-X operator. Each prepared state is measured across the whole Hilbert space, meaning, we measure it in all bases X, Y, and Z.

3.4 Variational Quantum Algorithms

Variational Quantum Algorithms (VQAs) are a class of hybrid quantum-classical algorithms designed to solve optimization problems. They exploit the power of quantum computers to explore quantum state spaces while leveraging classical computers to optimize parameterized quantum circuits[5]. We will start by implementing a parameterized quantum circuit (ansatz). This circuit will consist of single-qubit rotation gates (parameterized by angles).Next, we define the cost function based on the problem of interest, in this project, we aim to match the tomographic measurements between the test states (generated by the quantum circuit) and the target states. Once the cost function is defined, we need to use a classical optimizer to minimize it. The classical optimizer will update the parameters until the cost function is minimized.

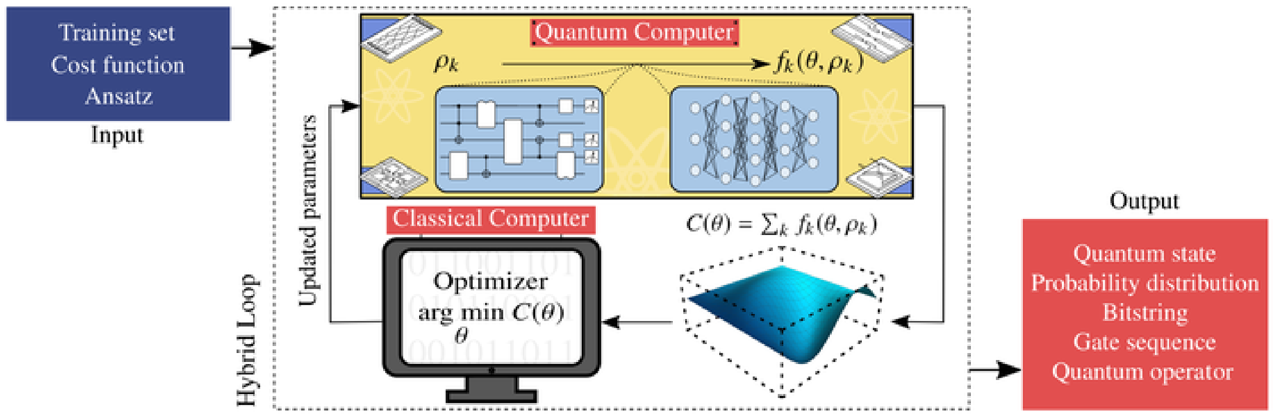


Figure 15: The diagram above illustrates a flowchart of a variational quantum algorithm. A quantum computer receives a quantum circuit composed of parameterized unitary gates. A cost function is constructed to evaluate the problem, and it is minimized by adjusting the tunable parameters using a classical optimizer. The updated parameters are then sent back to the quantum circuit to refine the results iteratively[5]

3.4.1 Variational Quantum Circuit

In this section, we detail the implementation of a variational quantum circuit (VQC) that utilizes prepared states and corresponding tomographic measurements for quantum process reconstruction. The VQC is instrumental in quantum machine learning, allowing us to train a quantum model[30] that approximates complex quantum processes.

The variational quantum circuit is built using a parameterized unitary operator, which consists of a sequence of rotation gates. These gates modify the quantum states and are key to adjusting the circuit's output. Instead of applying a fixed operator, as shown in Figure 15, we replace it with a unitary operator that has adjustable parameters θ, ϕ, λ.These parameters will be tuned during the training process. Inspired by the universal approximation theorem in neural networks[44] , we can invoke the universal rotation unitary equation, which is defined as:

$$U(\theta,\phi,\lambda) = R_z(\phi)R_y(\theta)R_z(\lambda) \tag{33}$$

$$U(\theta,\phi,\lambda) = \begin{pmatrix} e^{-i\frac{\phi}{2}} & 0 \\ 0 & e^{i\frac{\phi}{2}} \end{pmatrix} \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & -\sin\left(\frac{\theta}{2}\right) \\ \sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{pmatrix} \begin{pmatrix} e^{-i\frac{\lambda}{2}} & 0 \\ 0 & e^{i\frac{\lambda}{2}} \end{pmatrix} \tag{34}$$

$$\tag{35}$$

This yields,

$$U(\theta,\phi,\lambda) = \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & -e^{i\lambda}\sin\left(\frac{\theta}{2}\right) \\ e^{i\phi}\sin\left(\frac{\theta}{2}\right) & e^{i(\phi+\lambda)}\cos\left(\frac{\theta}{2}\right) \end{pmatrix} \tag{36}$$

The universal unitary rotation operator $U(\theta,\phi,\lambda)$ here is constructed by the use of the rotation gates R_z and R_y. Any arbitrary rotation on a qubit can be represented by a combination of just two rotations, one about the Y-axis and the other about the Z-axis. The reason is that these rotations are sufficient to cover all possible orientations of a qubit on the Bloch Sphere. By combining rotations around these two axes, one can move the qubit from any point on the Bloch sphere to any other point, which means one can achieve any arbitrary single-qubit unitary transformation.

Using $R_z(\phi), R_y(\theta), and R_z(\lambda)$ is a minimal way to construct any single qubit unitary operation. Although one could use rotations around other axes (like the X-axis), it turns out that this specific combination of Y-axis and Z-axis rotation is enough to fully describe any qubit transformation. This is a compact and efficient representation. In fact, the combination of rotations around Z and Y axes follows a known result in quantum mechanics , any unitary matrix can be decomposed into a sequence of rotations around different axes. This is sometimes referred to as **Euler's decomposition** for qubits[43].

Here we excluded the X-axis, $R_x(\alpha)$ becuase it can be expressed as combination of R_y and R_z rotations. So there is no need to explicitly include R_x in the construction, its redundant. Hence, the variational qunatum circuit constructed below is build on by applying the prepared qubit states and sending them all in parallel to the parameterized operator as seen in figure 17.

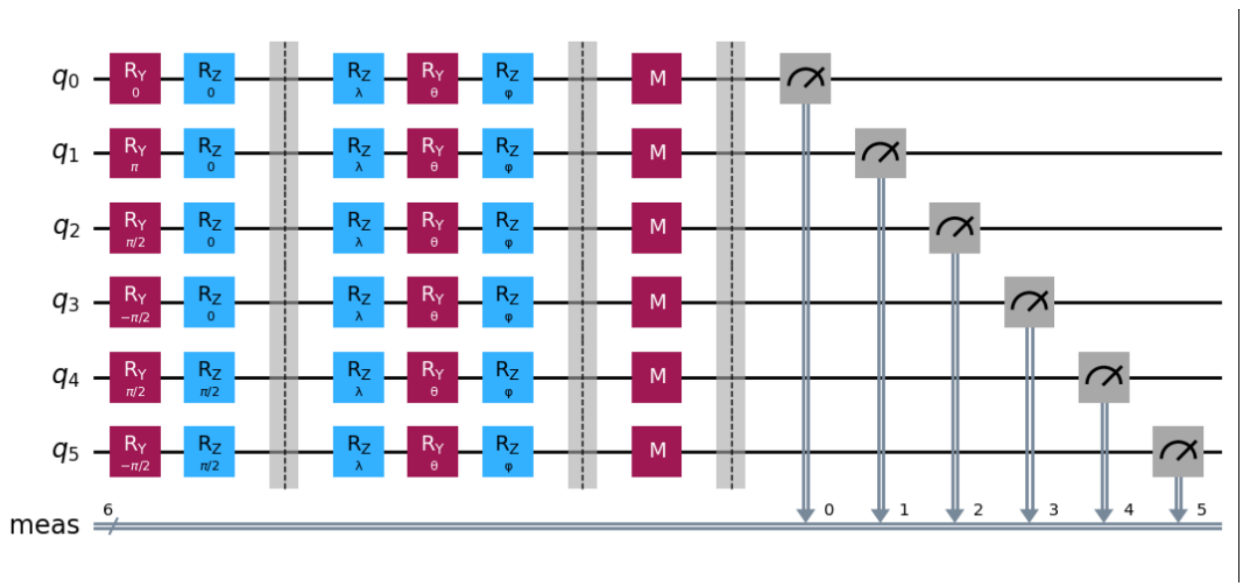


Figure 16: A visual depiction of the implemented variational quantum circuit. The first part illustrates the prepared states, the middle section, separated by two barriers, represents the parameterized unitary operator, and the final part displays the measurement operator.

3.4.2 Cost function

In Variational Quantum Algorithms (VQAs), the cost function is a mathematical expression that quantifies how well the current quantum circuit configuration (determined by its parameters) is solving the given problem. It is essentially the objective that the algorithm aims to minimize or maximize through optimization[5].

In our case, the cost function will be based on the **linear least square differences** between the predicted tomographic measurements generated by the implemented variational quantum circuit (VQC) and the true tomographic measurements collected (figure 16) are used as reference data[45]. The goal is to train the quantum circuit so that for each input state, it produces output measurements as close as possible to the true tomographic measurements. To achieve this, one will define a cost function that captures the difference between the predicted outcomes from the circuit and the known target measurements. The least square cost function can be defined as:

$$C(\theta,\phi,\lambda) = \sum_{i}^{N}[M_{pred,i}(\theta,\phi,\lambda) - M_{true,i}]^2 \tag{37}$$

where

- $M_{pred,i}(\theta,\phi,\lambda)$ is the predicted measurement outcome for the i-th input, parameterized by the variational parameters θ,ϕ,λ.
- $M_{test,i}$ is the true or target measurement for the i-th input.
- N is the number of total measurement outcomes.

The cost function computes the sum of the squared differences between the predicted and true measurements. This ensures that the optimization process aims to minimize the deviation between the two sets of measurements. A perfect match would lead to a cost function value of zero.

3.4.3 Optimization

In this process, a classical optimizers such as gradient descent, COBYLA, or SPSA) [9] are used to minimize a predefined cost function that quantifies the error between the QNN's predictions and the target measurements. The quantum circuit is executed on a quantum simulator (Qiskit AerSimulator) to compute the cost function, and the classical optimizer iteratively updates the parameters of the quantum gates to reduce this error[5]. This hybrid approach is necessary because, while classical optimization methods are efficient for parameter tuning, quantum circuits can represent and process complex quantum states and operations more naturally. In this project we will explore training the VQc by optimizing the cost function to find the best optimal parameters that minimize the cost function by using diffent optimizers.

The algorithm used by the optimizer **COBYLA**[5] it is a sequential trust region method. It solves nonlinear constrained optimization problems by approximating the objective function and the constraints using linear models with the trust region (parameter space).Now in this project, the COBYLA algorithm aims to minimize the defined cost function, following this steps:

- Minimizing the cost function $C(\theta,\phi,\lambda)$ with respect to the parameters θ,ϕ,λ:

$$min_{\theta,\phi,\lambda} C(\theta,\phi,\lambda) \tag{38}$$

- In each iteration, COBYLA will approximate the objective function (cost function) $C(\theta,\phi,\lambda)$ and any constraints with linear functions. Given that COBYLA is derivative-free (gradient free) optimizer, it uses finite-difference approximations or function evaluations at nearby points[46]:

$$C_k(\theta,\phi,\lambda) \approx C_k(\theta_k,\phi_k,\lambda_k) + p_k^T \nabla C_k(\theta_k,\phi_k,\lambda_k) \tag{39}$$

Where $p_k = (\delta\theta_k, \delta\phi_k, \delta\lambda_k)$ is the step direction for parameters at the k-th iteration. $\nabla C_k(\theta_k,\phi_k,\lambda_k)$ is an approximation to the gradient of the cost function at step k.

- COBYLA solves a linearized subproblem at each step:

$$min_{p_k} C_k(\theta_k + \delta\theta_k, \phi_k + \delta\phi_k, \lambda_k + \delta\lambda_k) \tag{40}$$

subject to constraints and the trust region condition:

$$||p_k|| \leq \Delta_k \tag{41}$$

where Δ_k is the trust region size at iteration k. Based on the quality of teh step p_k, COBYLA will either expand or shrink the trust region(parameter space):

- if $C(\theta_{k+1},\phi_{k+1},\lambda_{k+1}$ improves (the error decrease), the trust region size Δ_k is increased.
- If no significant improvement is made, Δ_{k+1} is decreased.

After solving the subproblem , COBYLA updates the paramters:

$$\theta_{k+1} = \theta_k + \delta\theta_k \tag{42}$$

$$\phi_{k+1} = \theta_k + \delta\phi_k \tag{43}$$

$$\lambda_{k+1} = \lambda_k + \delta\lambda_k \tag{44}$$

The process continues until the trust region Δ_k becomes smaller than a predefined tolerance, the cost function $C(\theta,\phi,\lambda)$ converges to a value below the specified toleranceCOBYLA, and maximum iterations are reached.In this case

$$min_{\theta,\phi,\lambda} C(\theta,\phi,\lambda) \approx 0 \tag{45}$$

Other optimizer will be explored as well, as an aid for comparing the best optimization algorithm that solves this problem best.

The second optimizer used in this project to optmize the cost function is the **SLSQP** (Sequential Least Square Programming), in which the algorithm is a gradient-based optimization method used for constrained nonlinear optimization[47]. The way it works is that it aims to minimize a scalar objective function, subject to both equality and inequality constraints.Given an obejctive function $f(x)$ where in this case it is the cost function $C(\theta,\phi,\lambda)$ and constraints , SLSQP solves:

Given the cost function:

$$C(\theta,\phi,\lambda) = \sum_{i=1}^{N} \left(M_{\text{pred},i}(\theta,\phi,\lambda) - M_{\text{test},i}\right)^2$$

where $M_{\text{pred},i}(\theta,\phi,\lambda)$ is the predicted measurement for the i-th input, and $M_{\text{test},i}$ is the test measurement, and θ,ϕ,λ are the optimization parameters.

- The objective function to minimize is:

$$f(\mathbf{x}) = C(\theta,\phi,\lambda) = \sum_{i=1}^{N} \left(M_{\text{pred},i}(\theta,\phi,\lambda) - M_{\text{test},i}\right)^2$$

where $\mathbf{x} = [\theta,\phi,\lambda]$ is the parameter vector.

- The gradient of the objective function with respect to the parameters $\mathbf{x} = [\theta,\phi,\lambda]$ is:

$$\nabla C(\theta,\phi,\lambda) = \left[\frac{\partial C}{\partial \theta}, \frac{\partial C}{\partial \phi}, \frac{\partial C}{\partial \lambda}\right]$$

Each partial derivative is given by[47]:

$$\frac{\partial C}{\partial \theta} = 2\sum_{i=1}^{N} \left(M_{\text{pred},i}(\theta,\phi,\lambda) - M_{\text{test},i}\right) \frac{\partial M_{\text{pred},i}}{\partial \theta}$$

$$\frac{\partial C}{\partial \phi} = 2\sum_{i=1}^{N} \left(M_{\text{pred},i}(\theta,\phi,\lambda) - M_{\text{test},i}\right) \frac{\partial M_{\text{pred},i}}{\partial \phi}$$

$$\frac{\partial C}{\partial \lambda} = 2\sum_{i=1}^{N} \left(M_{\text{pred},i}(\theta,\phi,\lambda) - M_{\text{test},i}\right) \frac{\partial M_{\text{pred},i}}{\partial \lambda}$$

The constraints, written as:

- Equality constraints: $\mathbf{h}(\mathbf{x}) = 0$
- Inequality constraints: $\mathbf{g}(\mathbf{x}) \leq 0$

At each iteration, these constraints are linearized around the current solution $\mathbf{x}_k$.

At iteration k, the SLSQP algorithm solves a quadratic programming subproblem to find the search direction $\mathbf{p}_k$, where:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$$

The quadratic approximation to the cost function is:

$$\min_{\mathbf{p}} \frac{1}{2}\mathbf{p}^\top \mathbf{B}_k \mathbf{p} + \nabla f(\mathbf{x}_k)^\top \mathbf{p}$$

subject to the linearized constraints:

$$\nabla \mathbf{g}(\mathbf{x}_k)^\top \mathbf{p} + \mathbf{g}(\mathbf{x}_k) \leq 0$$

$$\nabla \mathbf{h}(\mathbf{x}_k)^\top \mathbf{p} + \mathbf{h}(\mathbf{x}_k) = 0$$

where $\mathbf{B}_k$ is an approximation to the Hessian matrix of the cost function, typically updated using the BFGS(Broyden–Fletcher–Goldfarb–Shanno) method, more details can be found here [48].

The parameters are updated as follows:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$$

where α_k is determined using a line search, and more details is outlined in [47].The algorithm checks for convergence by verifying if:

$$||\nabla f(\mathbf{x}_k)|| < \varepsilon$$

where ε is a small threshold.

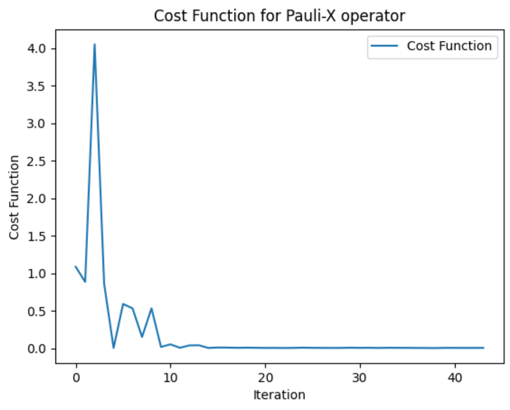
More optimizer will be outlined and tested for better results, we will explore the results produced by other optimizers, such as **Powell**, **Nelder-Mead**, and the two optimization algorithms defined above for the optimizers, COBYLA and SLSQP. The details about algorithms undertaken the above mentioned optimizers can be found in [5, 47, 48, 49, 50].

4 Optimization of the Variational Quantum Circuit

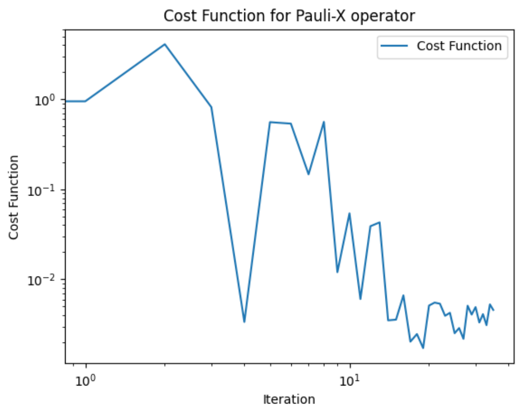
4.1 Cost Function Evolution During Optimization

In this section, we present the results of training the variational quantum circuit to approximate the target quantum operator based on the provided tomographic measurements. The optimization process relies on minimizing a pre-defined cost function that measures the discrepancy between the measurement outcomes predicted by the variational circuit and the actual tomographic data.

The initial parameters $\theta_i, \phi_i, \lambda_i$ were selected randomly to initialize the variational circuit, which was then iteratively optimized using a gradient-based algorithm. As the parameters were updated, the cost function decreased, guiding the circuit towards producing more accurate measurement outputs.To illustrate the performance of the algorithm, we begin by examining the behavior of the cost function over the course of the optimization. The following plots depict the evolution of the cost function as a function of the optimization steps, highlighting the convergence process and the final error after training. This following results and optimization analysis are focused on training the variational quantum circuit to approximate the X-Pauli operator.

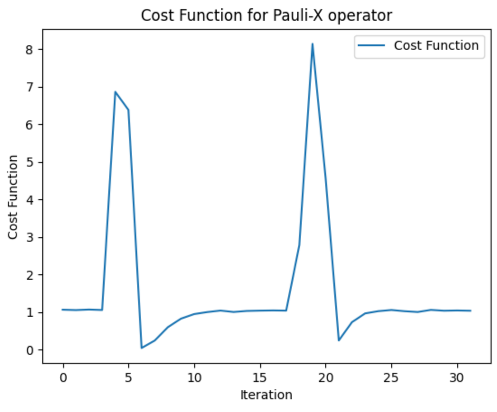


(a) Cost function on a normal scale

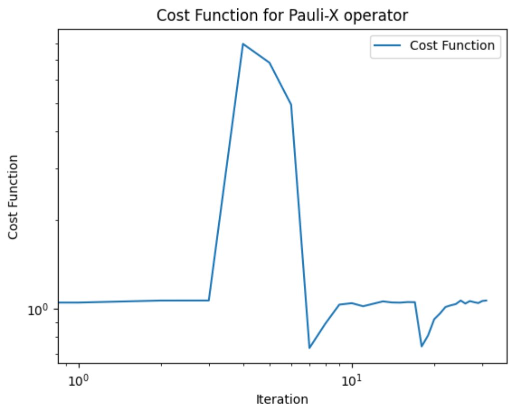


(b) Cost function on a logarithmic scale

Figure 17: The plots show the evolution of the cost function during the optimization process to find the parameters that approximate the unknown operator. The left plot displays the cost function on a normal scale, while the right plot shows the same data on a logarithmic scale, highlighting the rate of convergence. The optimization was performed using the COBYLA algorithm, and the decrease in the cost function indicates successful training of the variational quantum circuit to approximate the target operator.

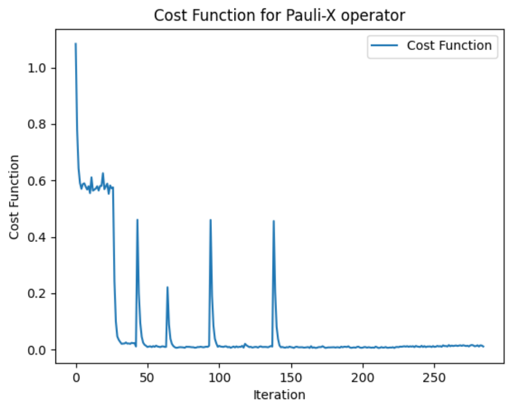


(a) Cost function on a normal scale

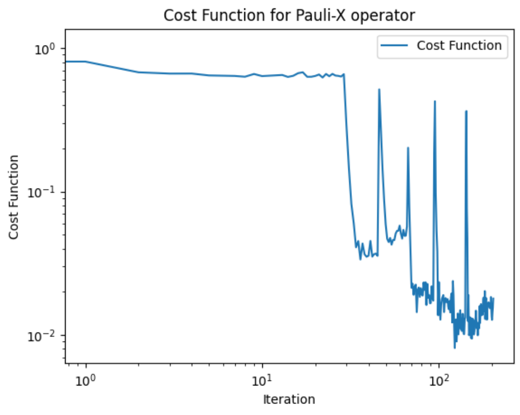


(b) Cost function on a logarithmic scale

Figure 18: The plots illustrate the convergence behavior of the cost function $C(\theta, \phi, \lambda)$ during the optimization process using the SLSQP optimizer. In normal scale, the cost function's reduction appears gradual, reflecting challenges in minimizing the difference between predicted and target measurements. The log scale plot provides a clearer view of the optimizer's progress, showing rapid initial decreases followed by slower changes, which may indicate convergence toward a local minimum or slow progress in flat regions of the parameter space.

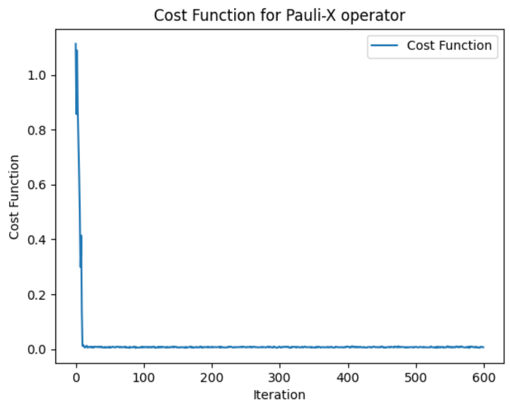


(a) Cost function on a normal scale

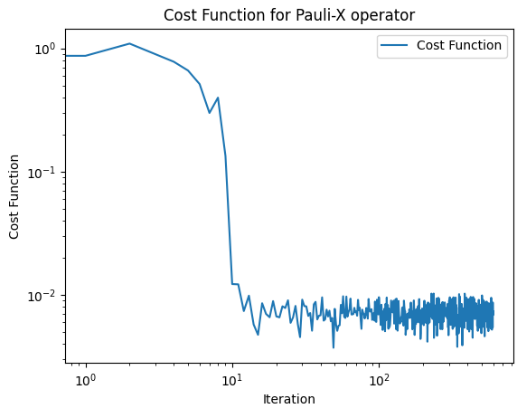


(b) Cost function on a logarithmic scale

Figure 19: Cost function optimization using the Powell algorithm, showing steady convergence towards the minimum, but with peaks along the way. Both the normal and logarithmic scales demonstrate the optimizer's effectiveness in minimizing the cost function over iterations, achieving favorable results.



(a) Cost function on a normal scale



(b) Cost function on a logarithmic scale

Figure 20: Cost function evolution computed using the Nelder-Mead optimizer, presented in both normal and log scales. The plots illustrate the convergence behavior, with the normal scale showcasing the raw cost values across iterations, while the log scale highlights finer details of the optimization progress, particularly during the final stages of convergence.

4.1.1 Analysis of Convergence

We applied four optimization algorithms namely Nelder-Mead, COBYLA, Powell, and SLSQP to minimize the cost function of our variational quantum circuit, which is designed to approach zero as the predicted measurements match the target values. Each optimizer performed differently in terms of convergence speed and effectiveness, providing insight into their suitability for this specific problem.

Among the tested methods, the Nelder-Mead algorithm demonstrated the fastest convergence. This algorithm, which is based on refining a simplex of candidate solutions, was able to efficiently navigate the parameter space without relying on gradients, making it particularly effective for our single-qubit optimization task. The method's reflection, expansion, and contraction steps allowed it to quickly hone in on the solution**??**.

The COBYLA algorithm, another gradient-free method, converged more slowly but still performed well in finding an optimal solution. It works by constructing linear approximations of the objective function within a trust region, adjusting cautiously toward the optimum. Although COBYLA's slower pace was noticeable with peak jumps along the convergence process **??**, it proved to be a reliable choice, particularly in problems involving constraints.

The Powell optimizer, which searches along one direction at a time using line searches, exhibited the slowest convergence among the successful methods. While Powell eventually minimized the cost function, its sequential update strategy can be less efficient when the cost function has complex curvature, we see this in the peaks formed on the cost function plot **??**. This resulted in slower progress, but the algorithm still achieved convergence with persistence.

In contrast, the SLSQP algorithm failed to converge. Despite being a gradient-based optimizer, it struggled with the characteristics of our cost function. SLSQP is well-suited for problems with smooth objective functions but can encounter difficulties when the cost function landscape is noisy or contains flat regions, we can see this due to the huge jump peak and the inability to converge towards zero**??**. Its reliance on gradients also made it susceptible to getting stuck in local minima, preventing it from finding the optimal solution in this case.

5 Unitary Operator Estimation

In this section, we present the results of estimating the target unitary operator, specifically the Pauli-X gate, using the variational quantum algorithm. By optimizing the parameters of the quantum circuit, the algorithm identifies the unitary transformation that best matches the target operator. The cost function, designed to minimize the difference between the predicted and actual measurement outcomes, converges to provide the optimal parameters. These parameters form the estimated unitary operator, which we then compare to the ideal Pauli-X operator, demonstrating the effectiveness of the quantum process in accurately learning and reproducing quantum operations.

The goal is to optimize the circuit parameters such that the resulting operator closely approximates the target unitary. The estimation process begins by employing the COBYLA algorithm to minimize the cost function, which measures the difference between the predicted measurement outcomes of the variational circuit and the expected outcomes for the Pauli-X gate.

5.1 COBYLA Operator Predictions

We begin by visualizing the estimated unitary operator obtained through our optimization process and comparing it to the target operator, which in this case is the Pauli-X operator. After the minimization of the cost function, the optimal parameters are identified, allowing the variational quantum circuit to accurately predict the unitary operator under investigation. This comparison provides a direct validation of the circuit's ability to learn and approximate the desired quantum transformation.

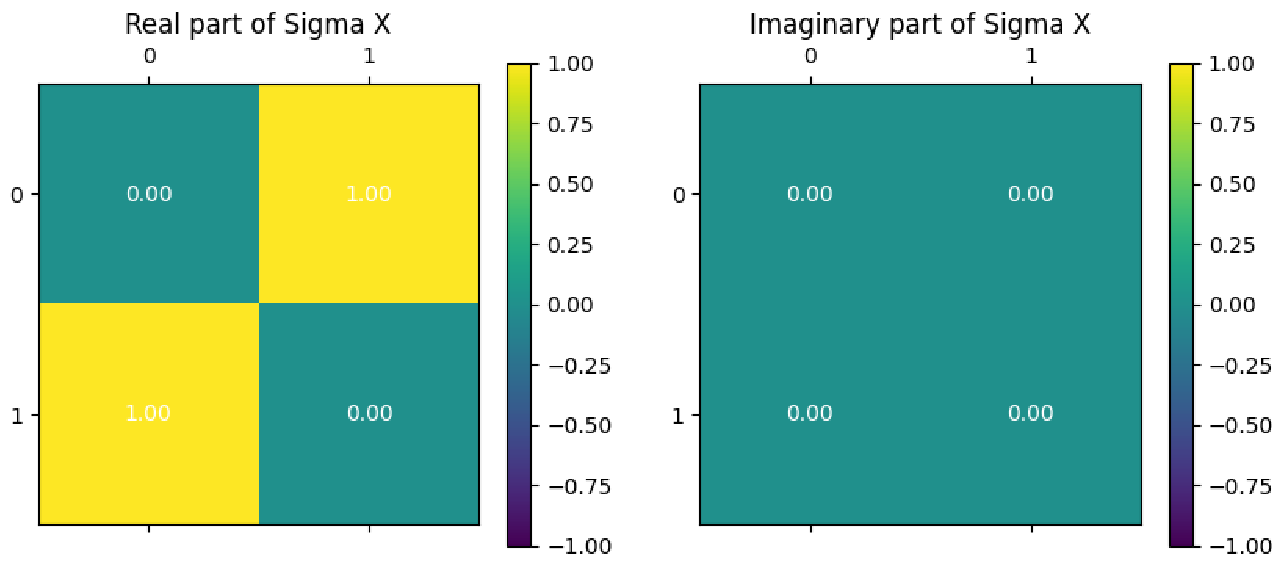


Figure 21: Visual representation of the actual Pauli-X operator

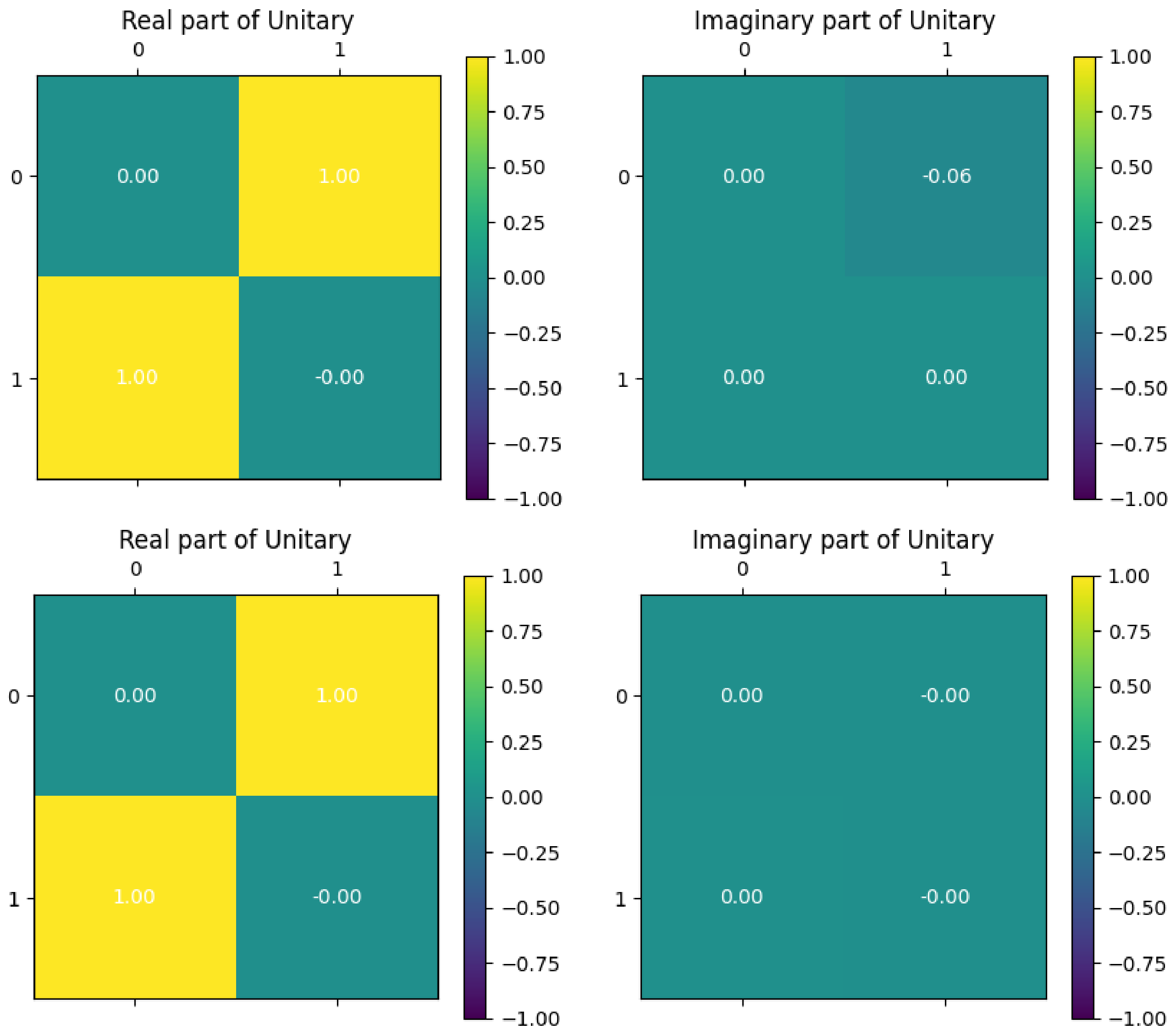


Figure 22: Visualization of the predicted unitary operator obtained through the variational quantum circuit.The top diagram shows the predicted unitary operator at 1024 shots, while the bottom displays the operator at 1,000,000 measurement shots.

In conclusion, the predictions of the unitary operator for the Pauli-X gate using the COBYLA optimizer demonstrate a successful approach to quantum operator estimation. The enhanced accuracy achieved through a higher number of measurement shots underscores the importance of statistical reliability in quantum measurements. As we progress, the forthcoming diagrams will further elucidate the effectiveness of various optimization algorithms such as SLSQP, Powell, and Nelder-Mead in refining our estimates. This comparative analysis will contribute to a deeper understanding of how different optimization strategies can impact the accuracy and efficiency of variational quantum algorithms in estimating quantum operators.

5.2 SLSQP Operator Predictions

In this part, we will explore the results obtained using the SLSQP optimization algorithm. The diagrams presented below illustrate the predicted unitary operator generated by the variational quantum circuit, utilizing the SLSQP optimizer for parameter estimation. These predictions reflect the algorithm's attempt to approximate the Pauli-X operator, based on the optimized cost function.

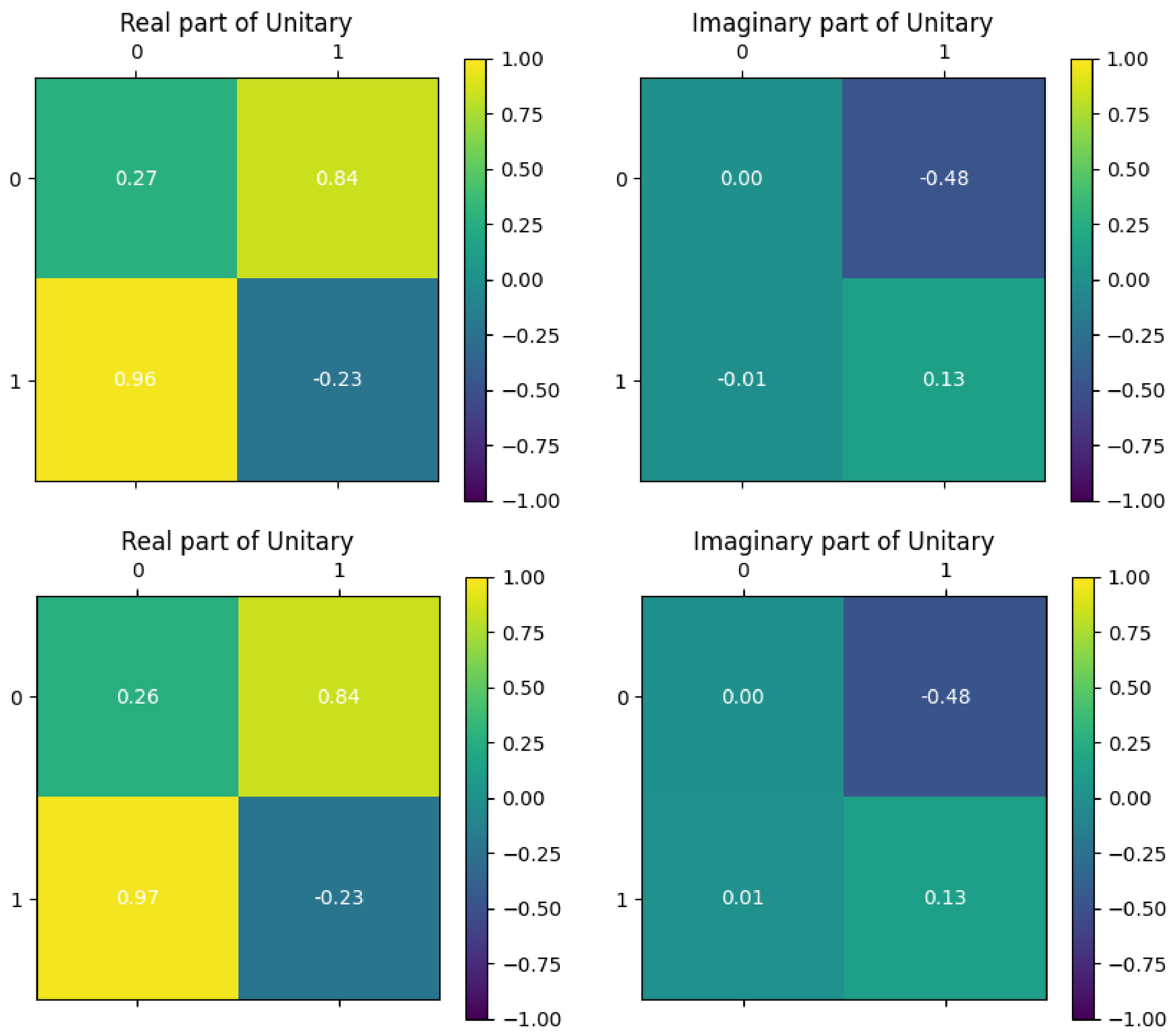


Figure 23: Visual representation of the predicted unitary operators generated by the variational quantum circuit using the SLSQP optimizer. The top diagram shows the predicted unitary operator based on results obtained after 1024 measurement shots, while the diagram below display the unitary operator unveiled from 1,000,000 shots of measurements.

5.3 Powell Operator Predictions

And here, we explore the algorithm used by the Powell to minimize the cost function and estimating the quantum unitary operator of interest.

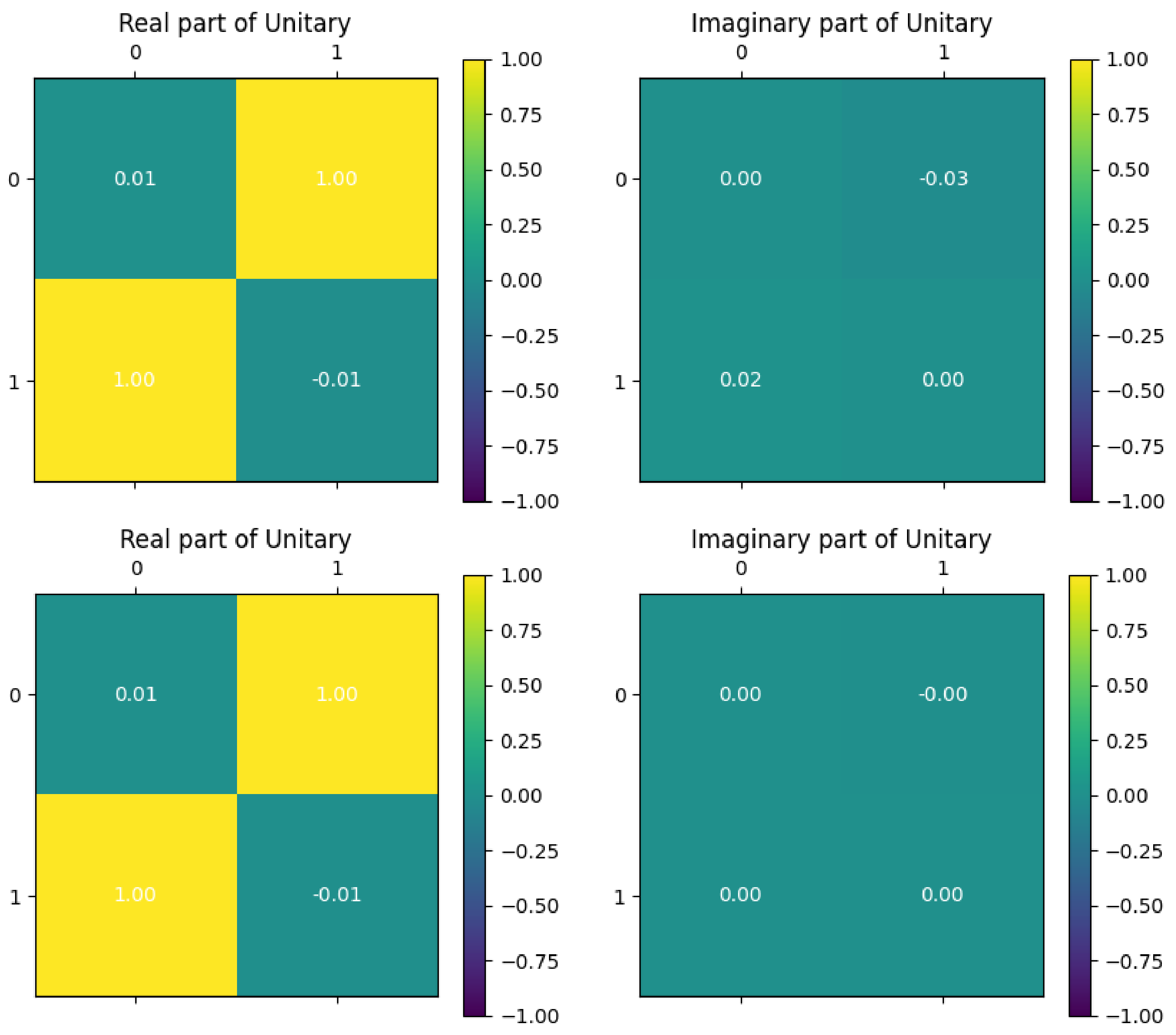


Figure 24: Visual representation of the predicted unitary operators generated by the variational quantum circuit using the Powell optimizer. The top diagram shows the predicted unitary operator based on results obtained after 1024 measurement shots, while the diagram below display the unitary operator unveiled from 1,000,000 shots of measurements.

5.4 Nelder-Mead Operator Predictions

Lastly, we explore the prediction of the quantum unitary operator through the use of the Nelder-Mead optimizer.

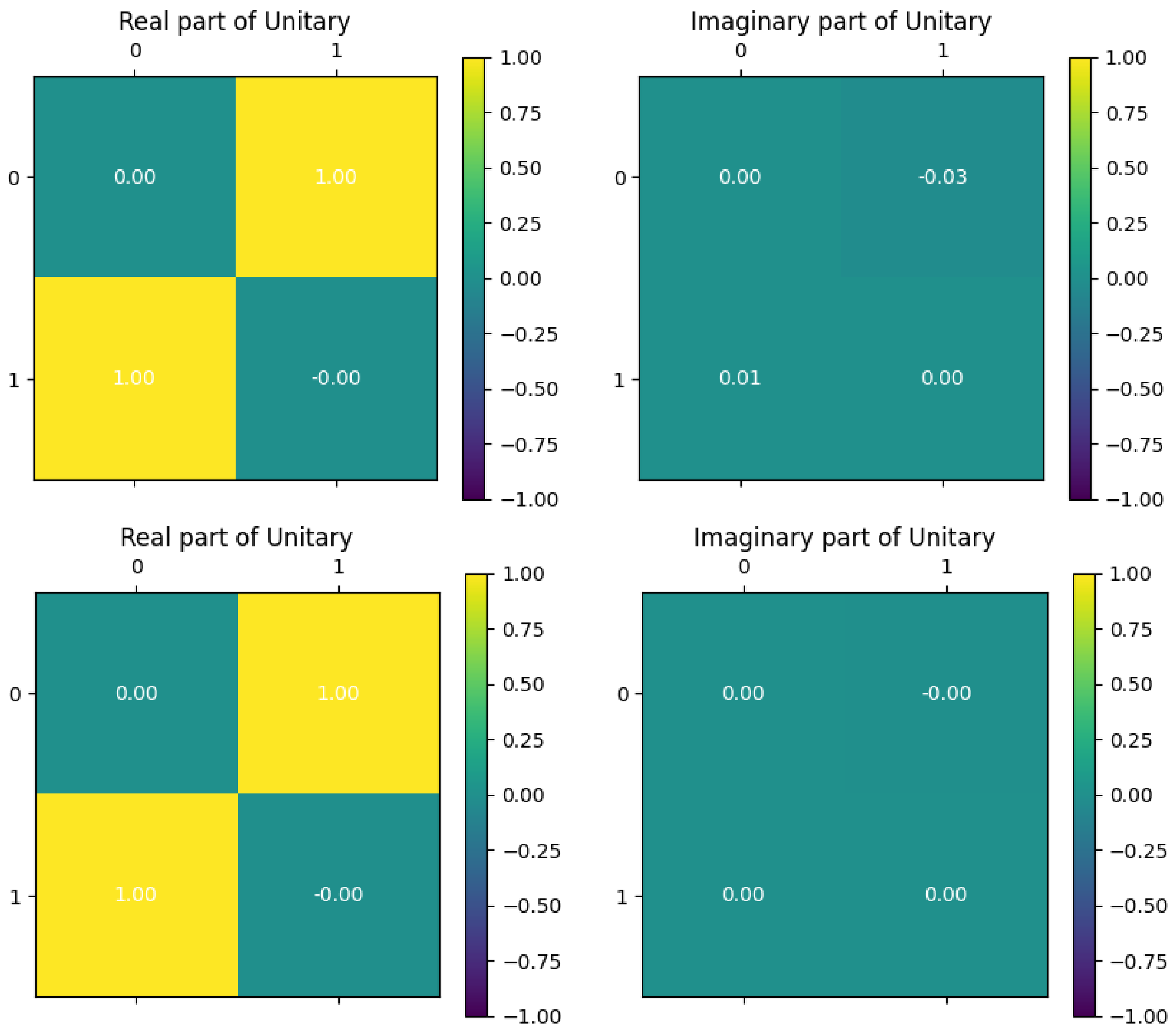


Figure 25: Visual representation of the predicted unitary operators generated by the variational quantum circuit using the Nelder-Mead optimizer. The top diagram shows the predicted unitary operator based on results obtained after 1024 measurement shots, while the diagram below display the unitary operator unveiled from 1,000,000 shots of measurements.

6 Comparison Between Actual and Predicted Measurement Outcomes

In this section, we compare the actual test measurements with the predicted measurement outcomes generated by the variational quantum circuit. This comparison is explored across different optimization methods, highlighting the performance of each algorithm in estimating the measurements. The accuracy of the predicted measurements is evaluated based on how closely they align with the test data, offering insight into the effectiveness of each optimizer used in training the quantum circuit. We begin by observing the results obtained by the optimizer COBYLA, and further on look at measurements predicted by other optimizers such as the SLSQP, Powell, and Nelder-Mead.

6.1 COBYLA Predicted Measurements

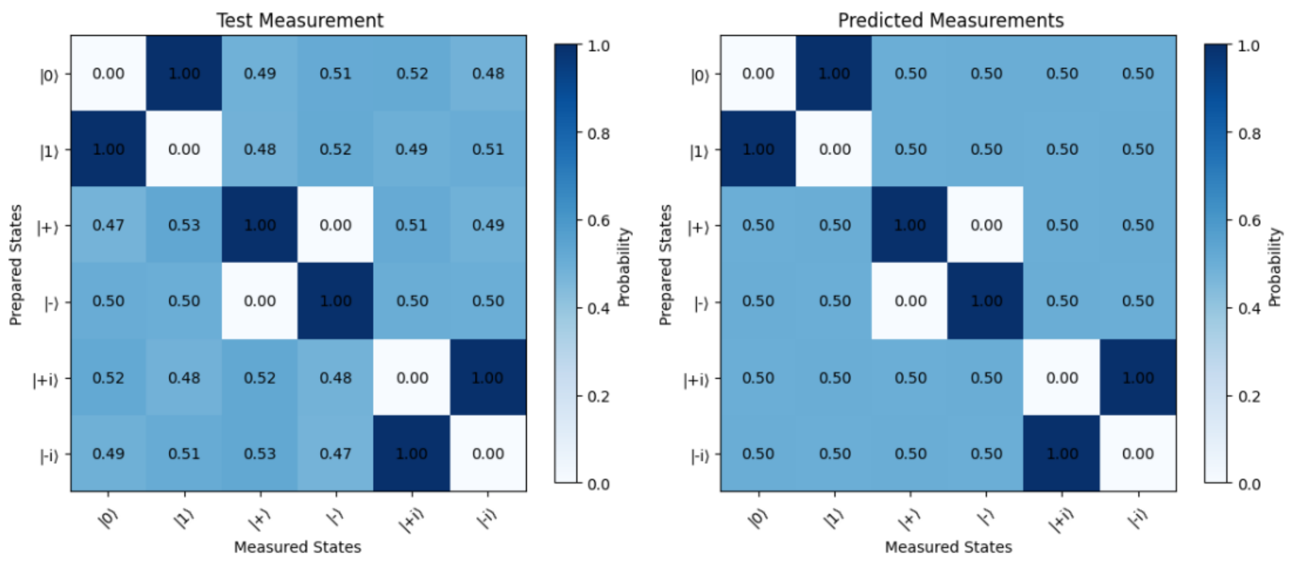


Figure 26: The left diagram represents the test measurements, while the right diagram shows the predicted measurements using the COBYLA optimizer. The variational quantum circuit closely approximates the test measurements, indicating effective prediction.

6.2 SLSQP Predicted Measurements

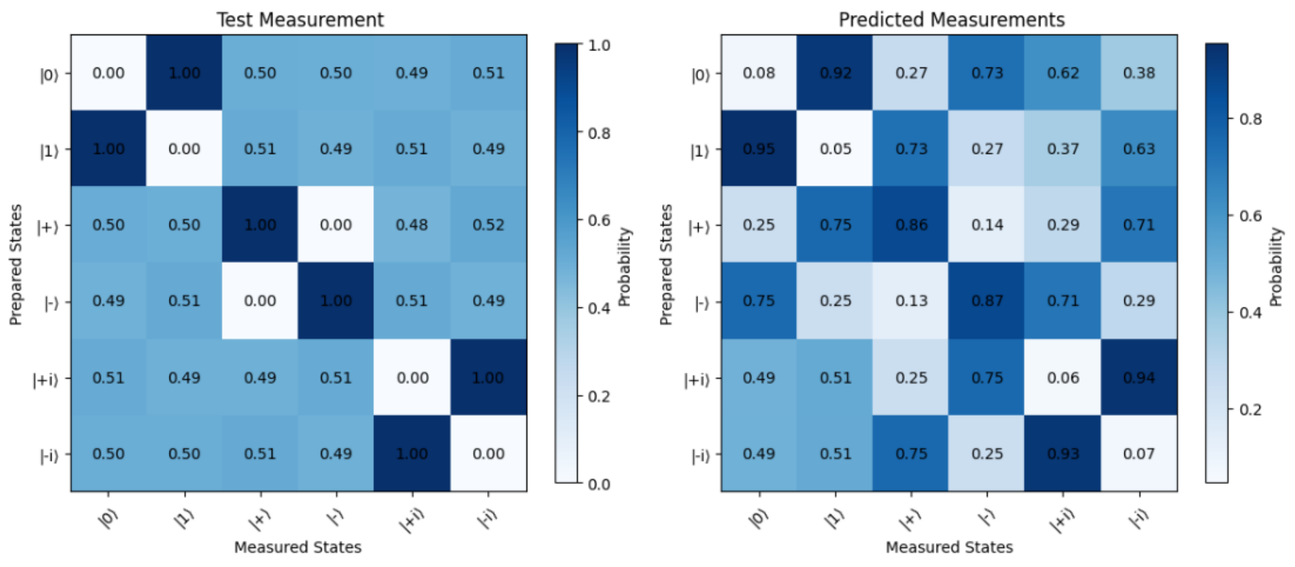


Figure 27: The left diagram represents the test measurements, while the right diagram shows the predicted measurements using the SLSQP optimizer. The variational quantum circuit fails to approximates the test measurements using this optimization algorithm. The predicted measurements are way off from the test measurements.

6.3 Powell Predicted Measurements

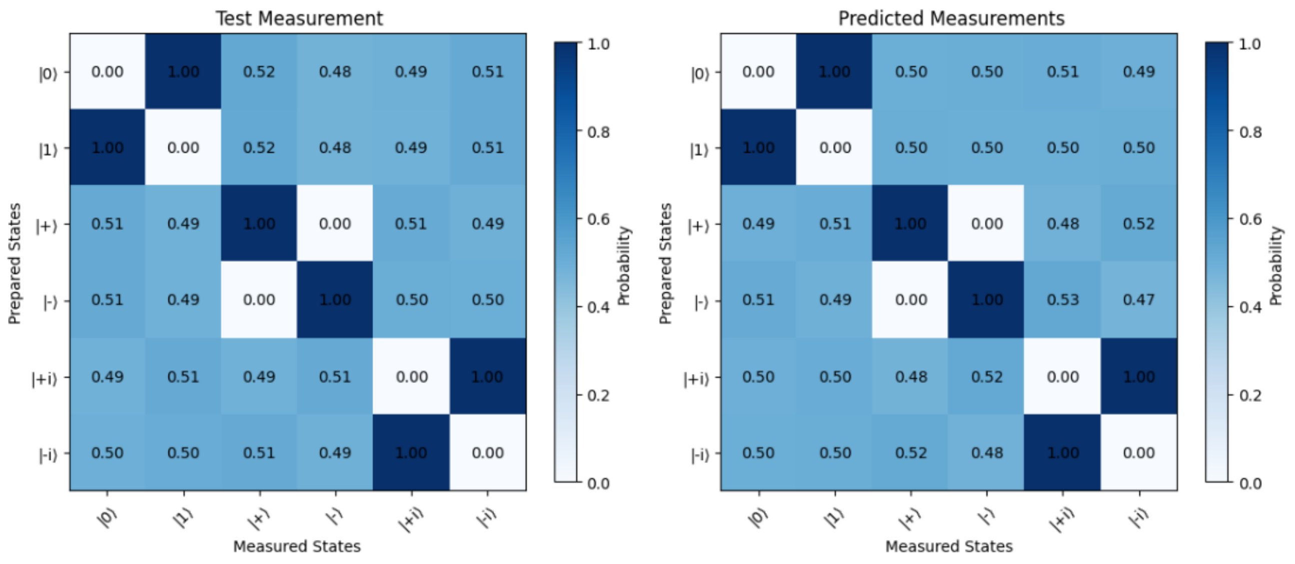


Figure 28: The left diagram represents the test measurements, while the right diagram shows the predicted measurements using the Powell optimizer. The variational quantum circuit fails to approximates the test measurements using this optimization algorithm. The predicted measurements are accurately close to the test measurements.

6.4 Nelder-Mead Predicted Measurements

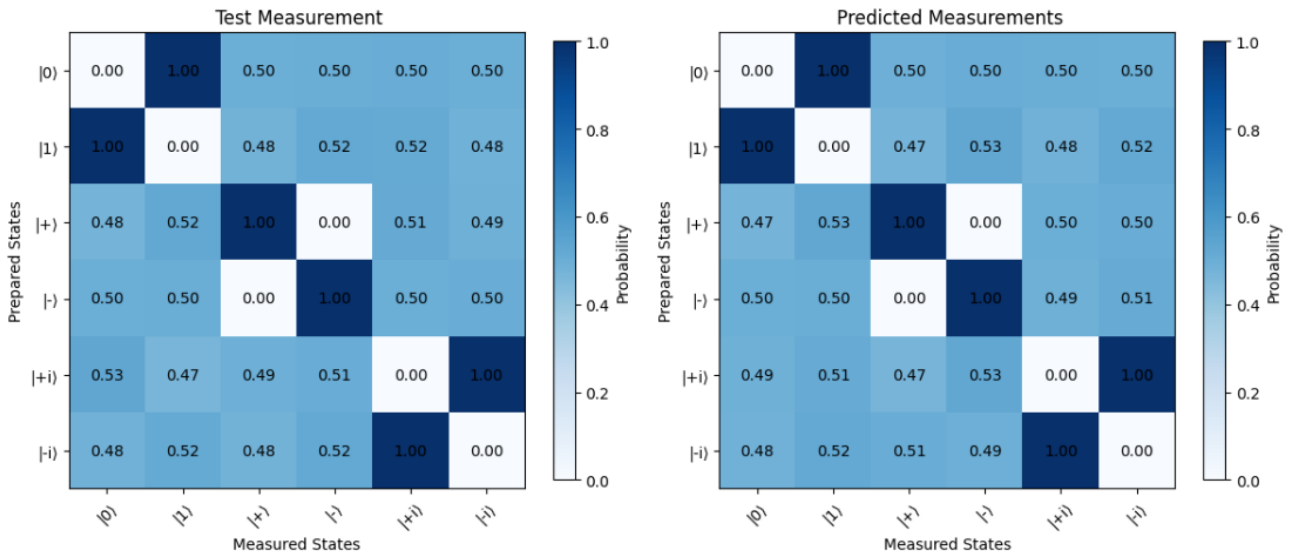


Figure 29: The left diagram represents the test measurements, while the right diagram shows the predicted measurements using the Nelder-Mead optimizer. The variational quantum circuit fails to approximates the test measurements using this optimization algorithm. The predicted measurements are closely accurate to the test measurements

7 More Operator Examples

In this section, we extend the procedure employed by the variational quantum algorithm in this project to predict additional single-qubit operators, including the Pauli-Y, Pauli-Z, and Hadamard operators. We begin by revisiting the cost function, using the COBYLA optimization algorithm to train our model. For simplicity, we focus on COBYLA, rather than applying each of the previously discussed optimizers for every operator under study.

7.1 Pauli-Y operator

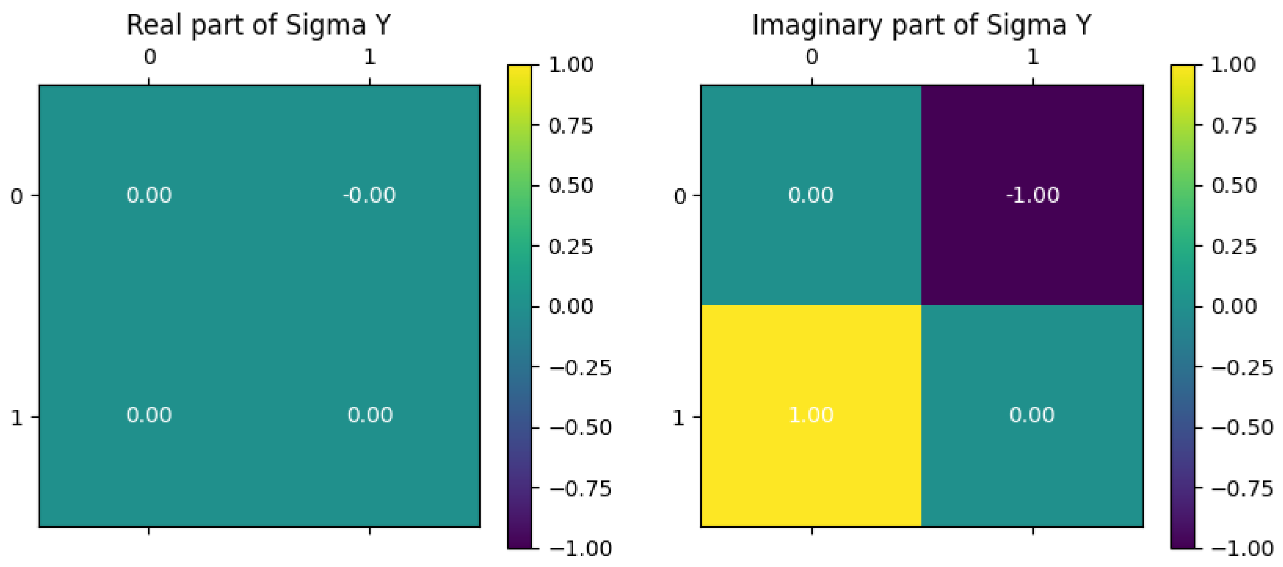


Figure 30: A visual representation of the actual Pauli-Y operator.

7.1.1 COBYLA

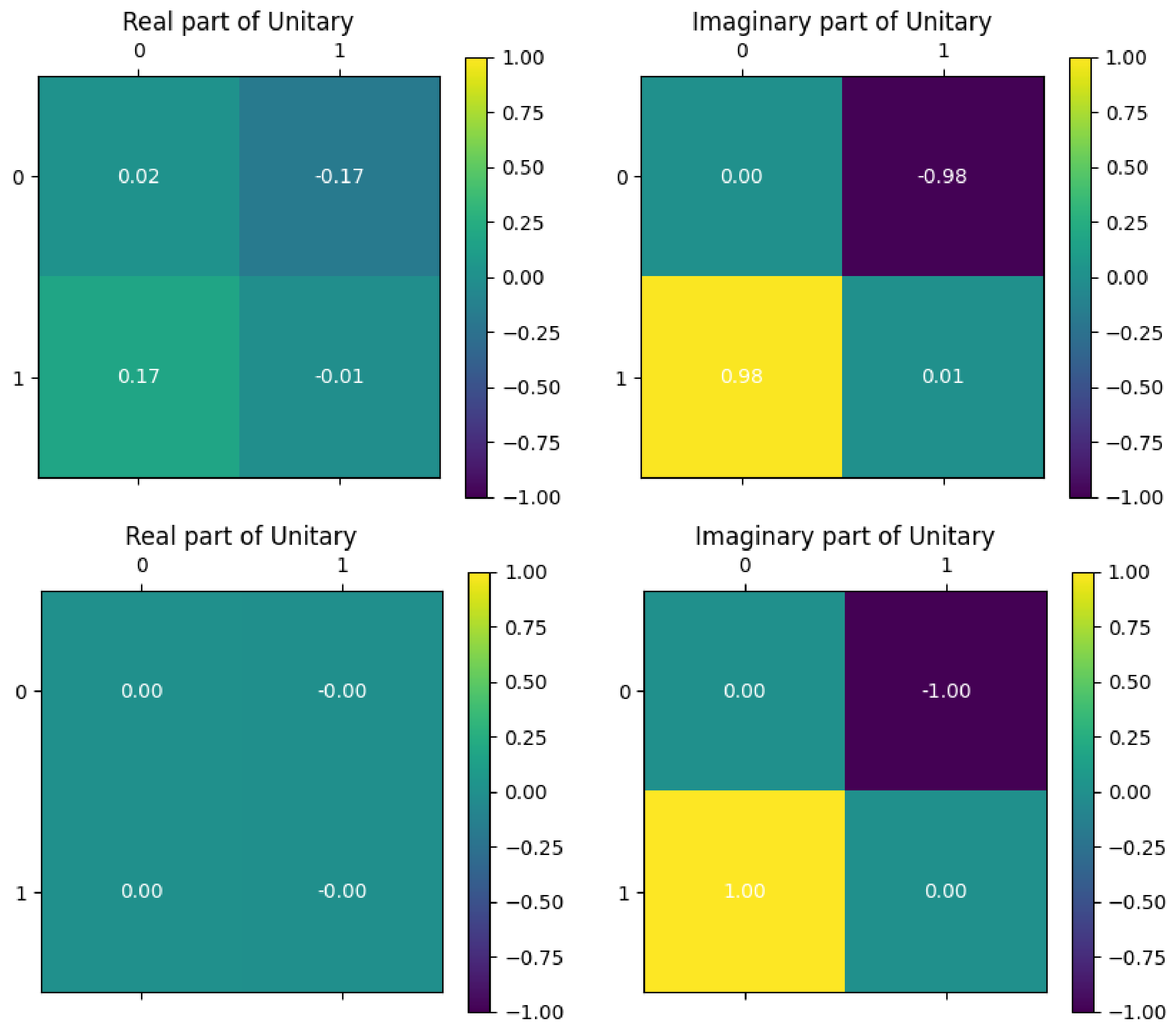


Figure 31: The diagrams illustrate the predictions of the Pauli-Y operator made by the variational quantum circuit. The top diagram shows the prediction based on 1,024 shots using the COBYLA optimizer, while the diagram below is derived from 1,000,000 shots.

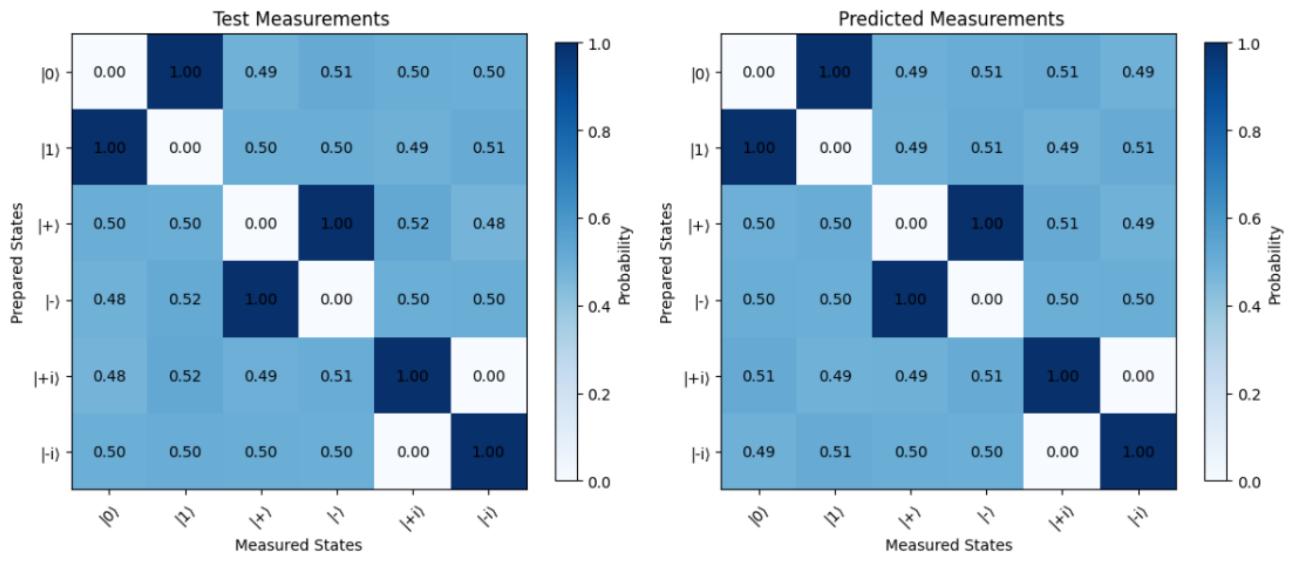


Figure 32: The measurements predicted by the variational quantum circuit after the prepared states have passed through the parameterized operator using the COBYLA optimizer. The left diagram shows the test measurements, while the right diagram display the predicted probability measurements. The accuracy of these probability measurements compared to each other is good.

7.1.2 Nelder-Mead

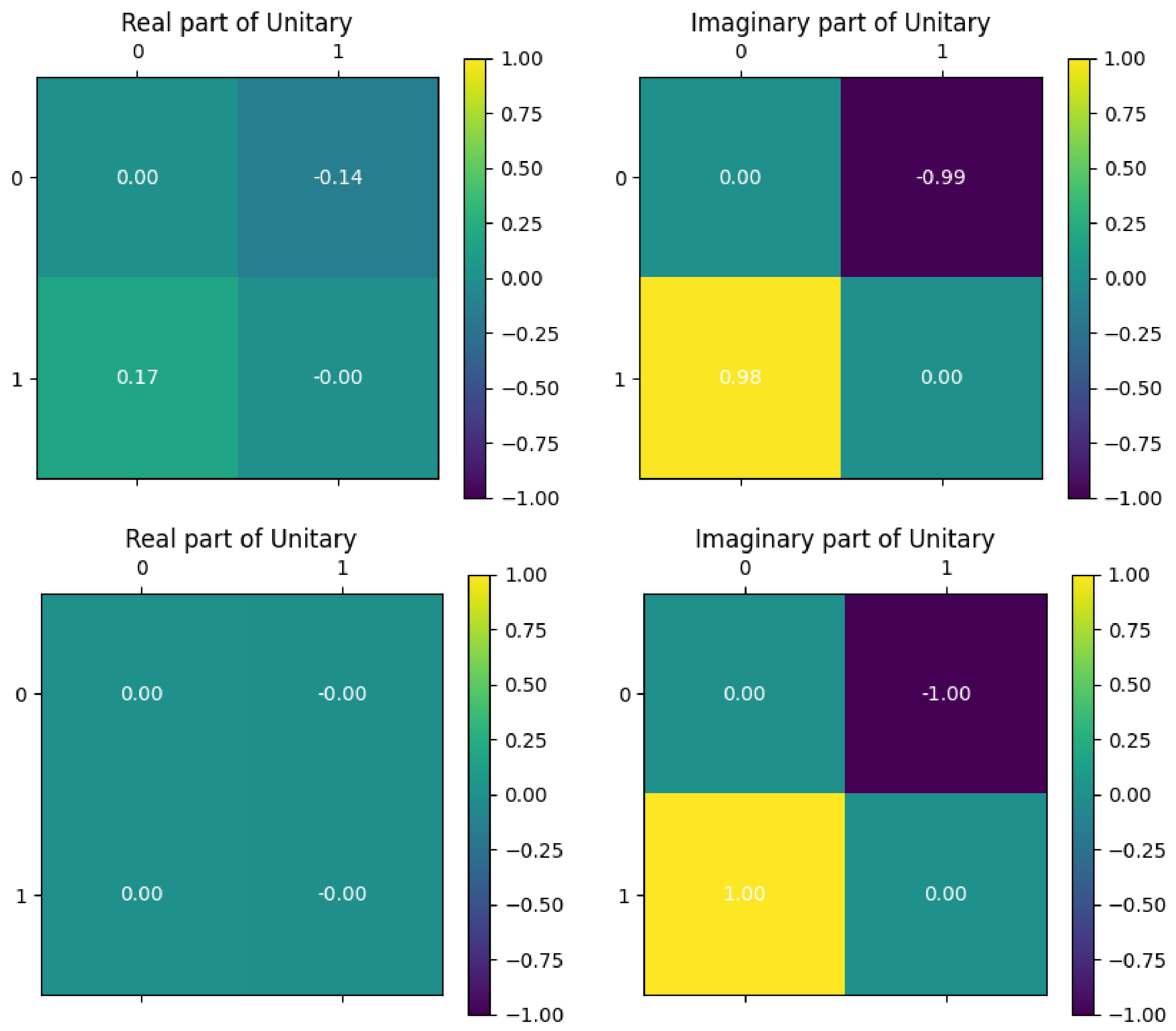


Figure 33: The diagrams illustrate the predictions of the Pauli-Y operator across different shot counts. The top diagram shows the variational quantum circuit's prediction based on 1,024 shots, while the middle diagram displays the prediction with 1,000,000 shots

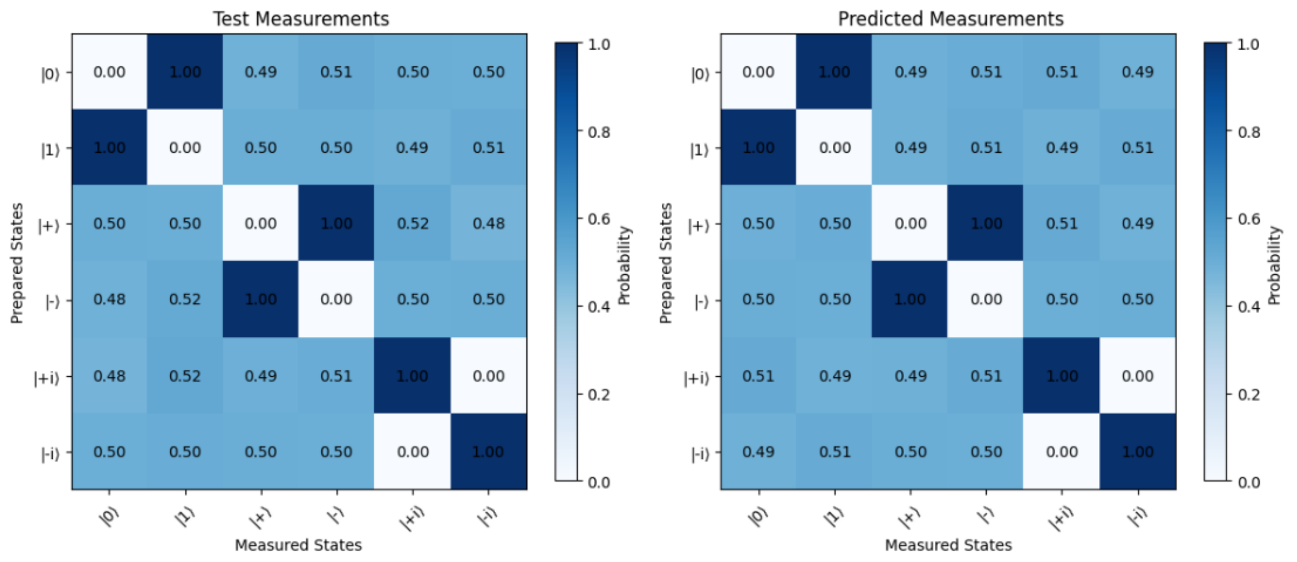


Figure 34: A representation probability measurements. The left diagram represents the test probability measurements, while the right diagram display the probability measurements predicted by the variational quantum circuit after the prepared state have been send through the parameterized unitary operator. The accuracy of the predicted measurements compared to the test measurements is convincing.

7.2 Pauli-Z operator

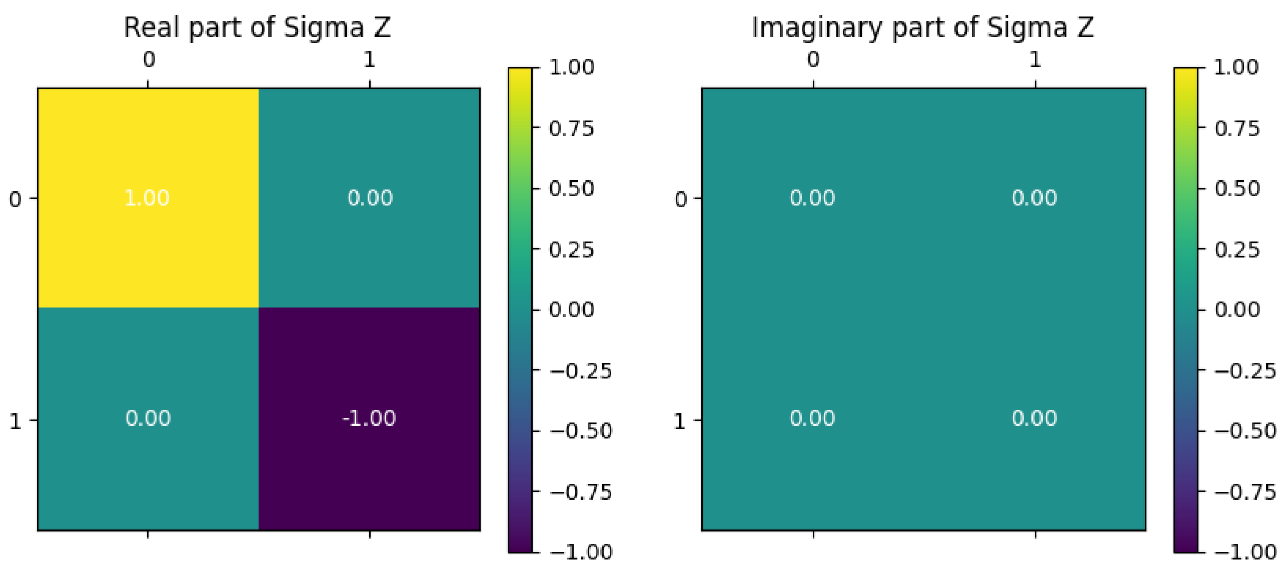


Figure 35: A visual representation of the actual Pauli-Z operator.

7.2.1 COBYLA

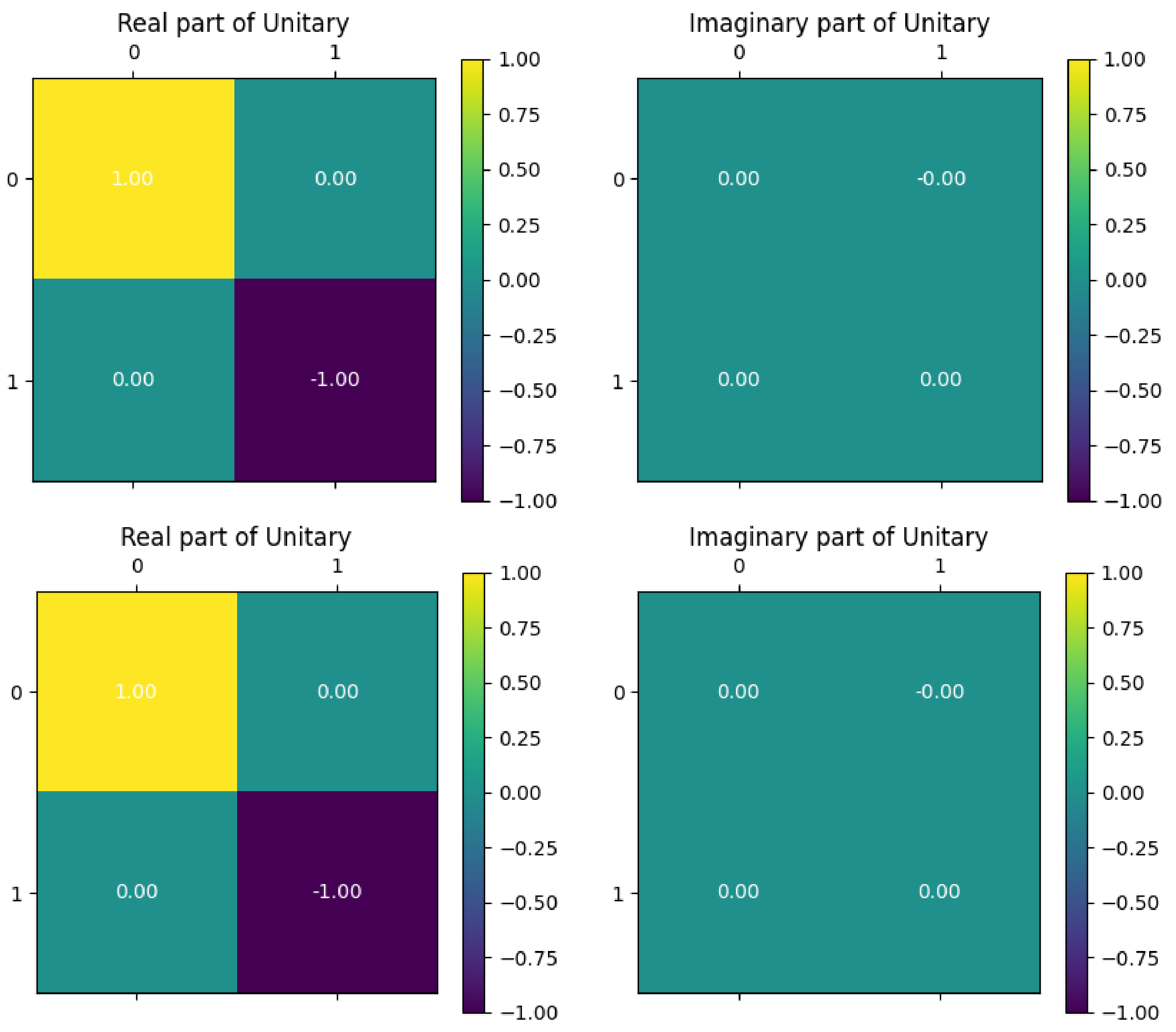


Figure 36: The two diagrams above depict the predictions of the Pauli-Z operator by the variational quantum circuit using the COBYLA optimizer. The top diagram presents the results obtained from 1,000 shots of quantum states, while the diagram below showcases the results from 1,000,000 shots.

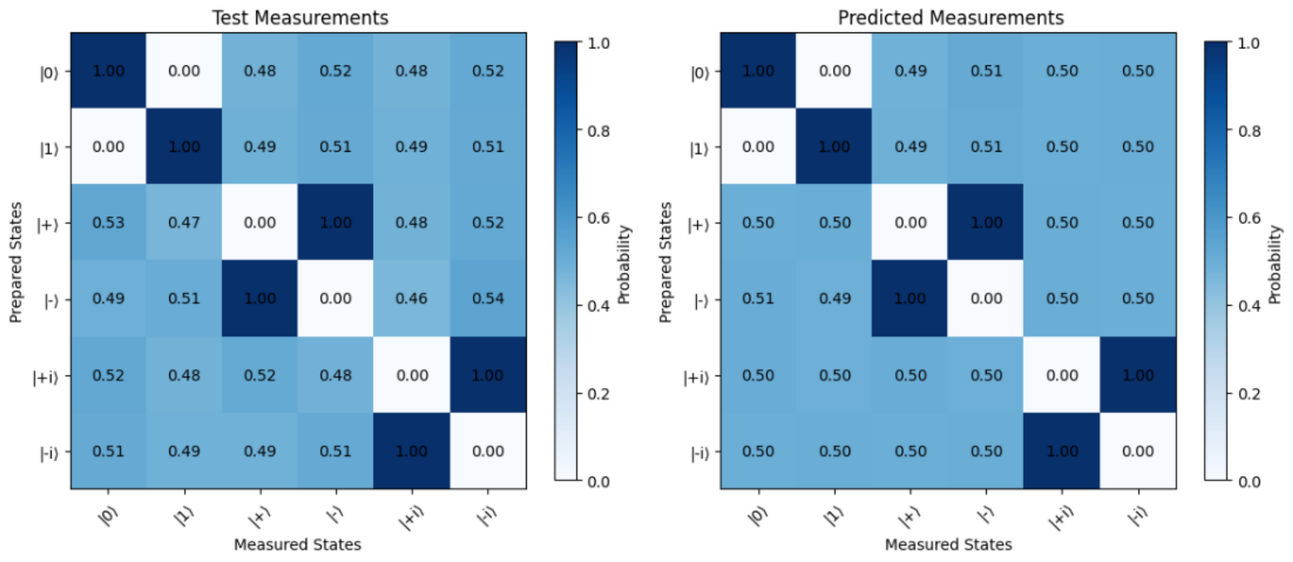


Figure 37: The corresponding measurements predicted by the variational quantum circuit after the states have passed through the parameterized unitary operator, generated using the Nelder-Mead optimizer. The accuracy of the predictions is also high.

7.2.2 Nelder-Mead

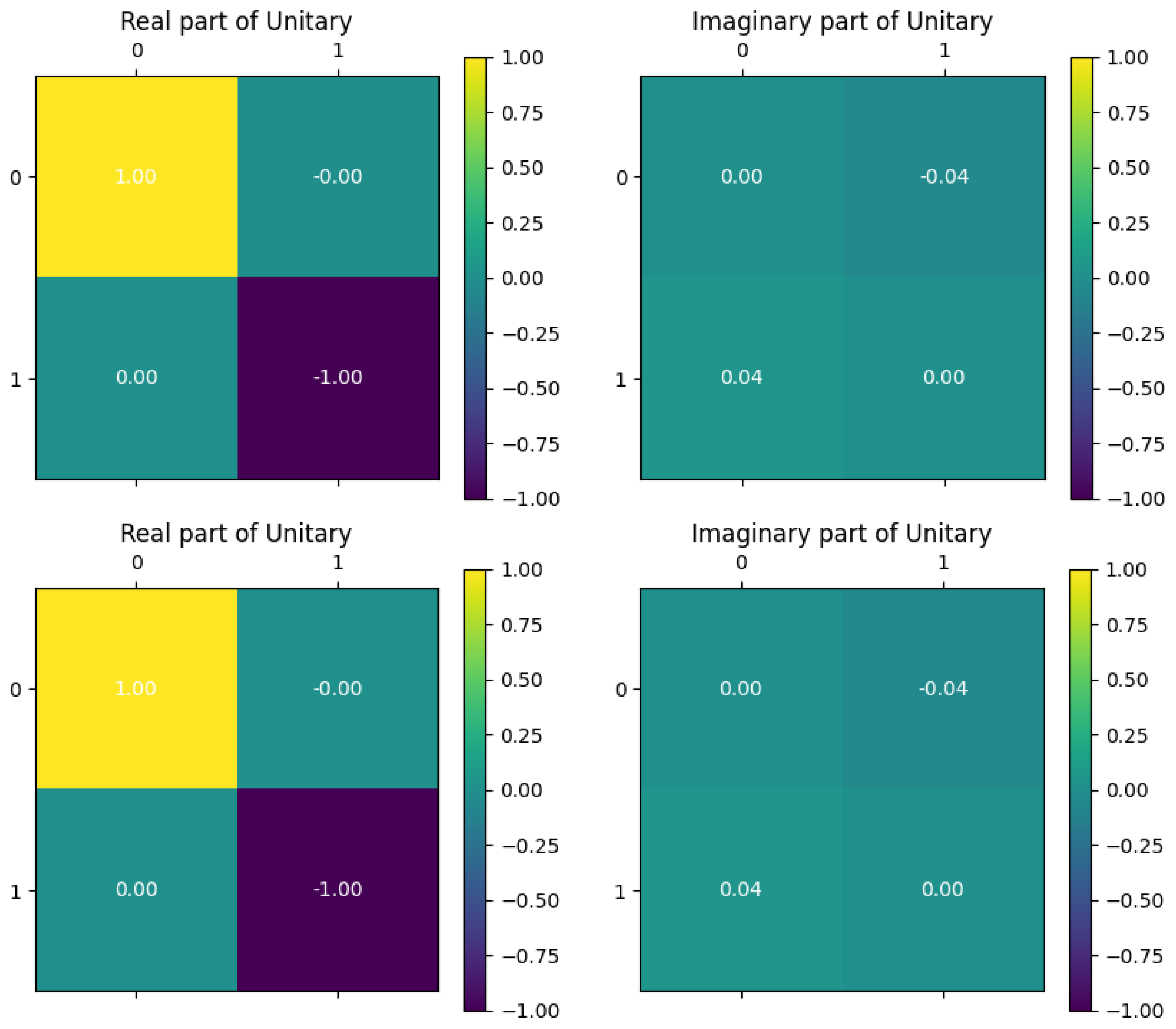


Figure 38: The two diagrams above depict the predictions of the Pauli-Z operator by the variational quantum circuit using the COBYLA optimizer. The top diagram presents the results obtained from 1,024 shots of quantum states, while the diagram below showcases the results from 1,000,000 shots.

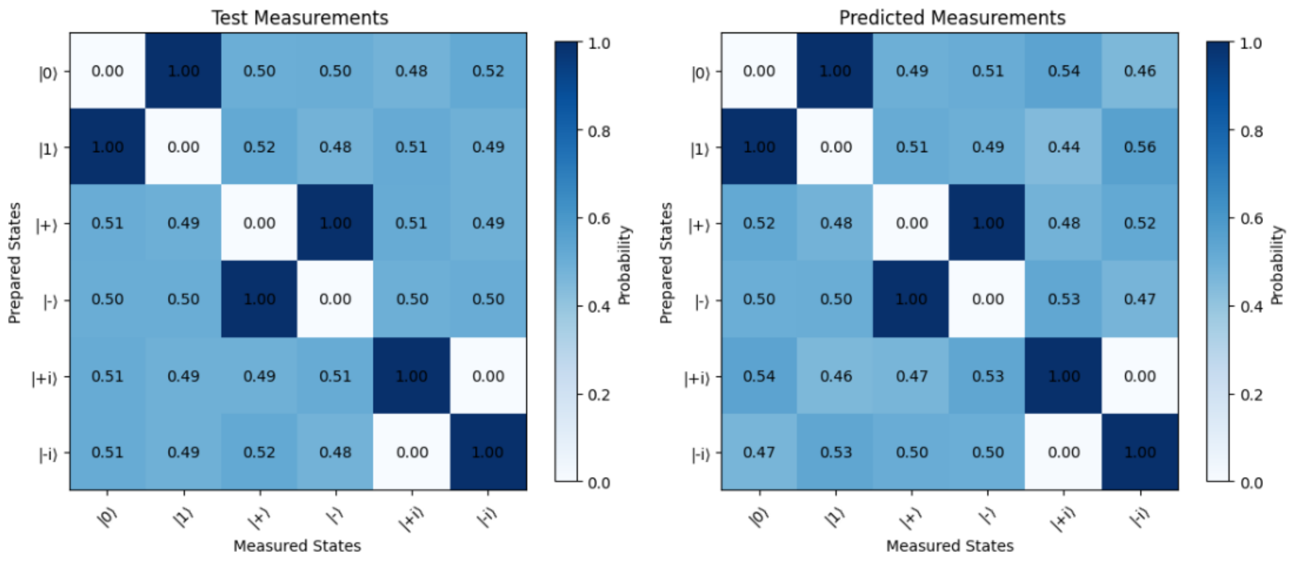


Figure 39: The corresponding measurements predicted by the variational quantum circuit. The left side on the diagram represent the test measurements, while the right side of the diagram above display the predicted probability measurements using the Nelder-Mead optimizer.

7.3 Hadamard operator

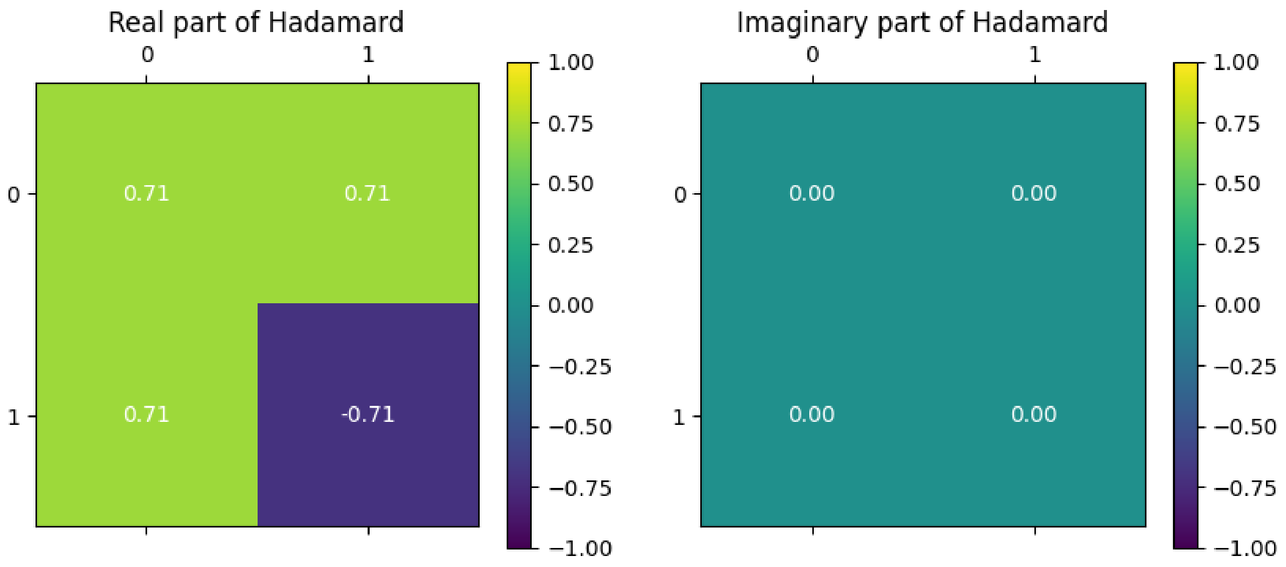


Figure 40: A visual representation of the actual Hadamard operator

7.3.1 COBYLA

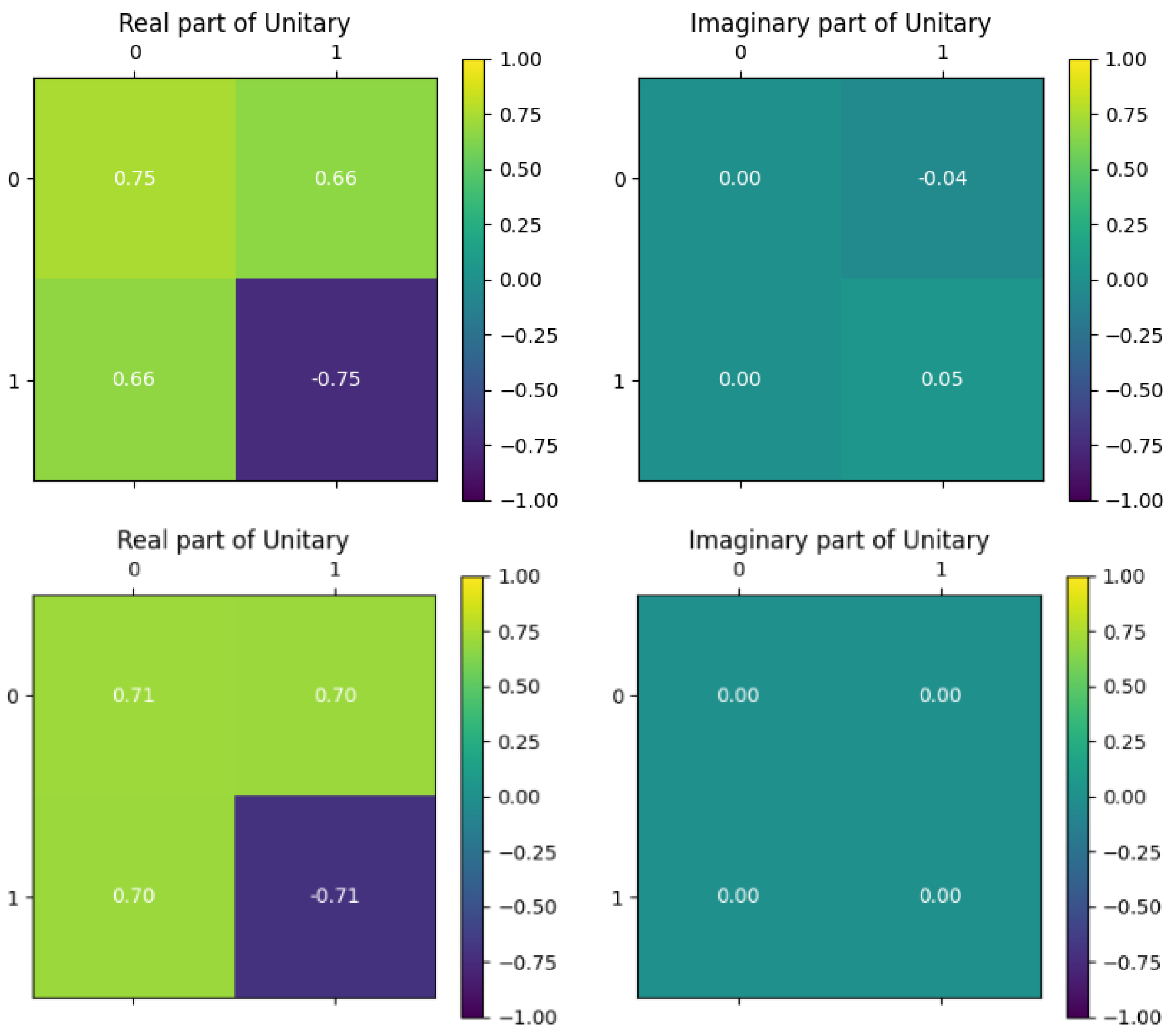


Figure 41: A visual representation of the predicted Hadamard operator by the variational quantum circuit, the top diagram shows the prediction of the operator at 1024 shots of qunatum state , while the bottom diagram display the operator's prediction at 1,000,000 shots, improving its accuracy.

7.3.2 Nelder-Mead

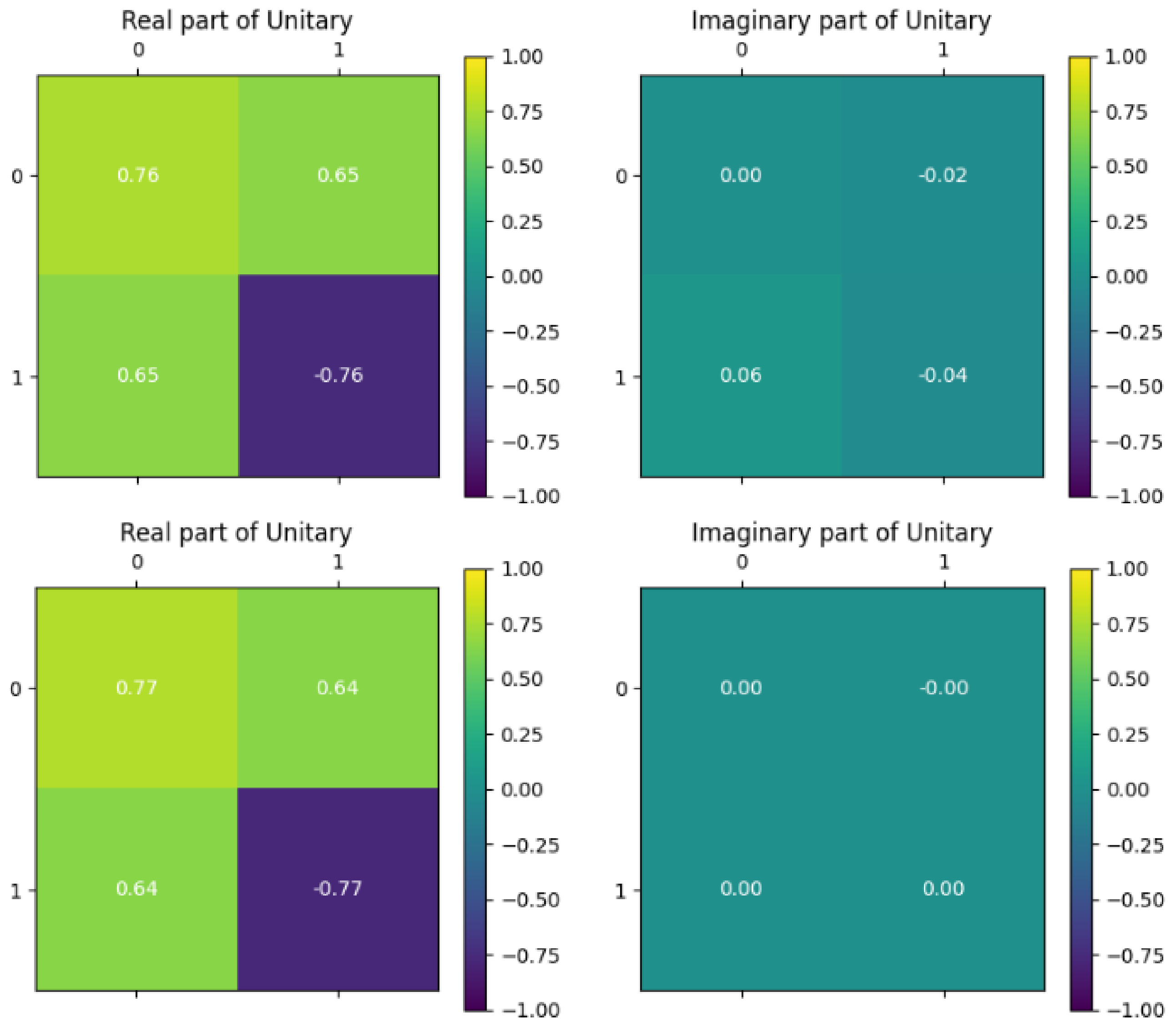


Figure 42: A visual representation of the predicted hadamard operator by the implemented variational quantum circuit through the use of the Nelder-Mead optimizer. The top diagram illustrate the 1024 shots quantum states sent into the VQC for this displayed prediction, while on the bottom diagram is a improved prediction of the hadamard operator at 1,000,000 shots of quantum states.

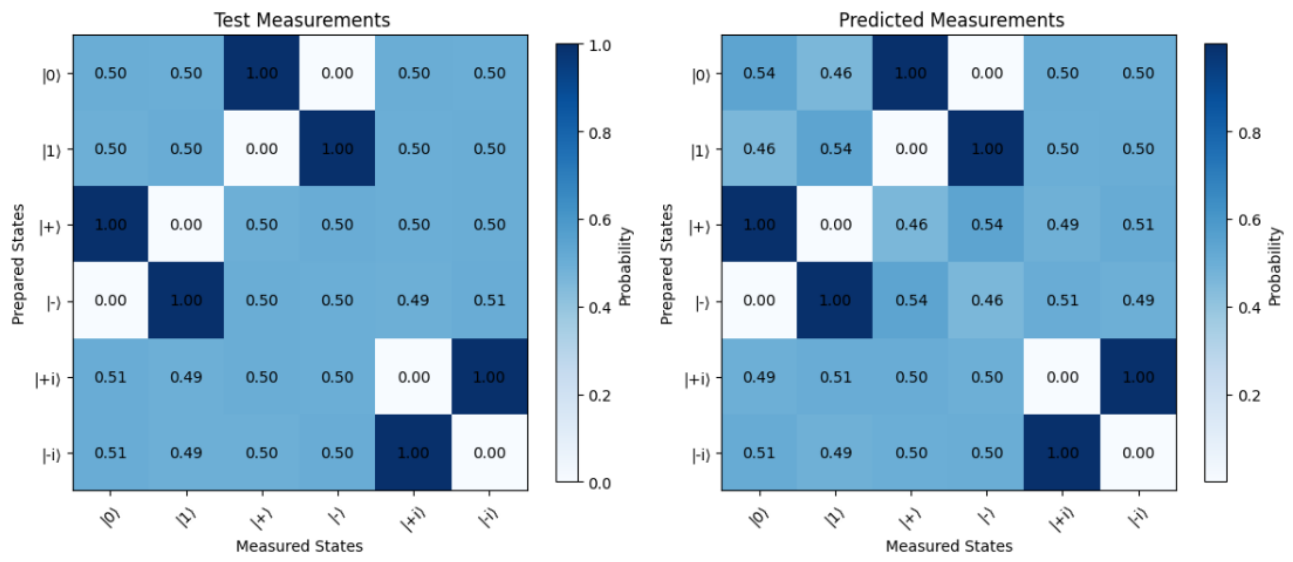


Figure 43: The corresponfing diagram represents the probability measurements predicted by the variational quantum circuit using the Nelder-Mead optimizer.

7.4 S-gate (phase gate)

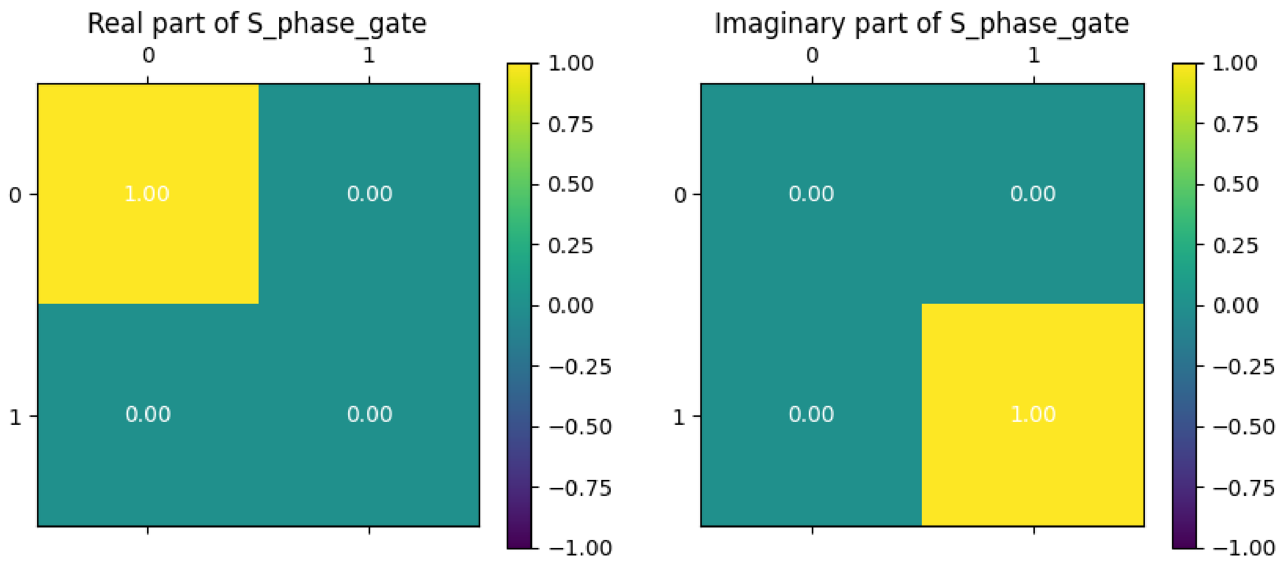


Figure 44: This diagram represents the actual phase gate.

7.4.1 COBYLA

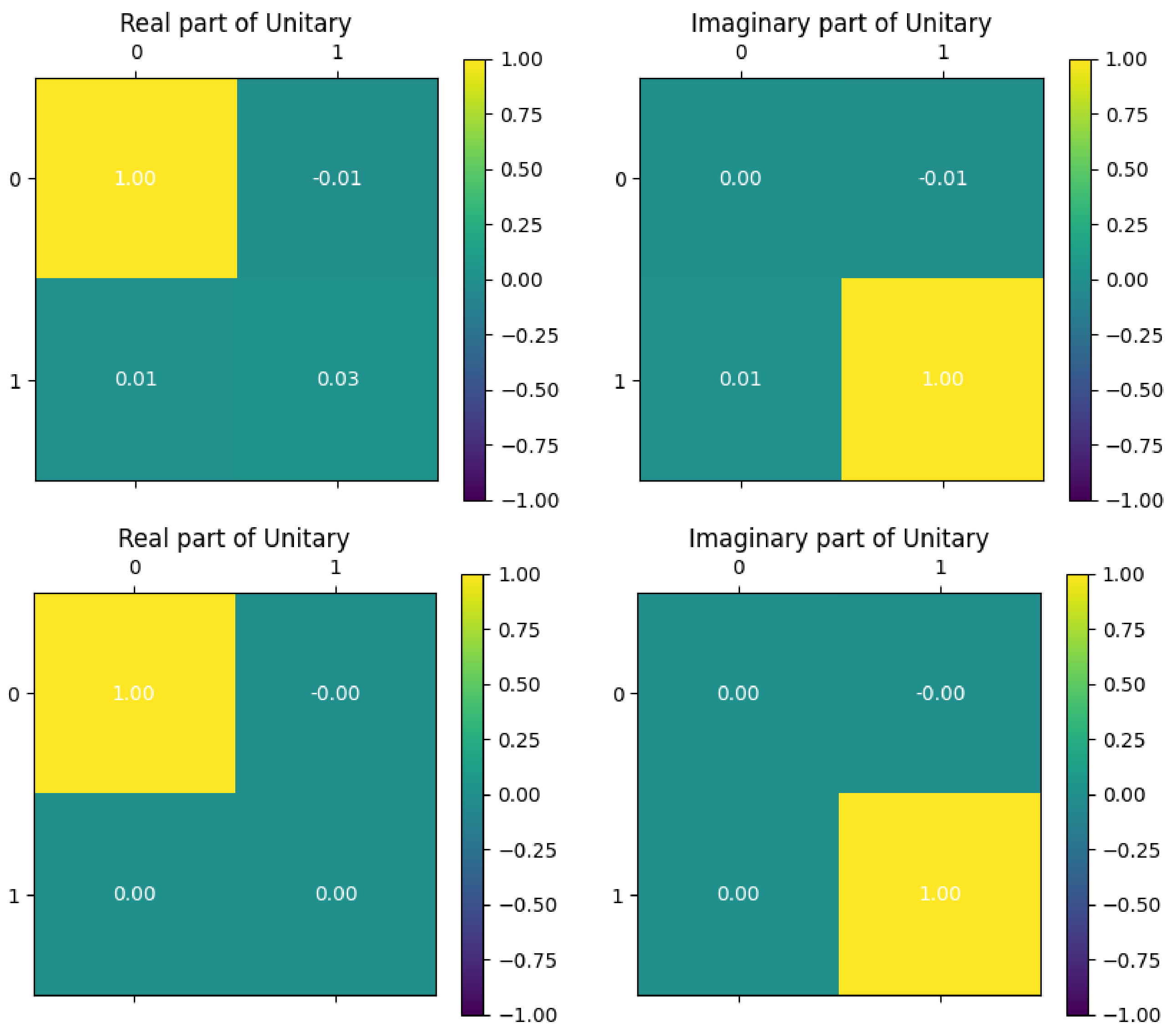


Figure 45: The top diagram represents the predicted phase gate by the variational quantum circuit by sending 1024 shots, while the below diagram display the predicted phase gate where 1,000,000 shots have been used.The optimization algorithm used is of COBYLA optimizer.

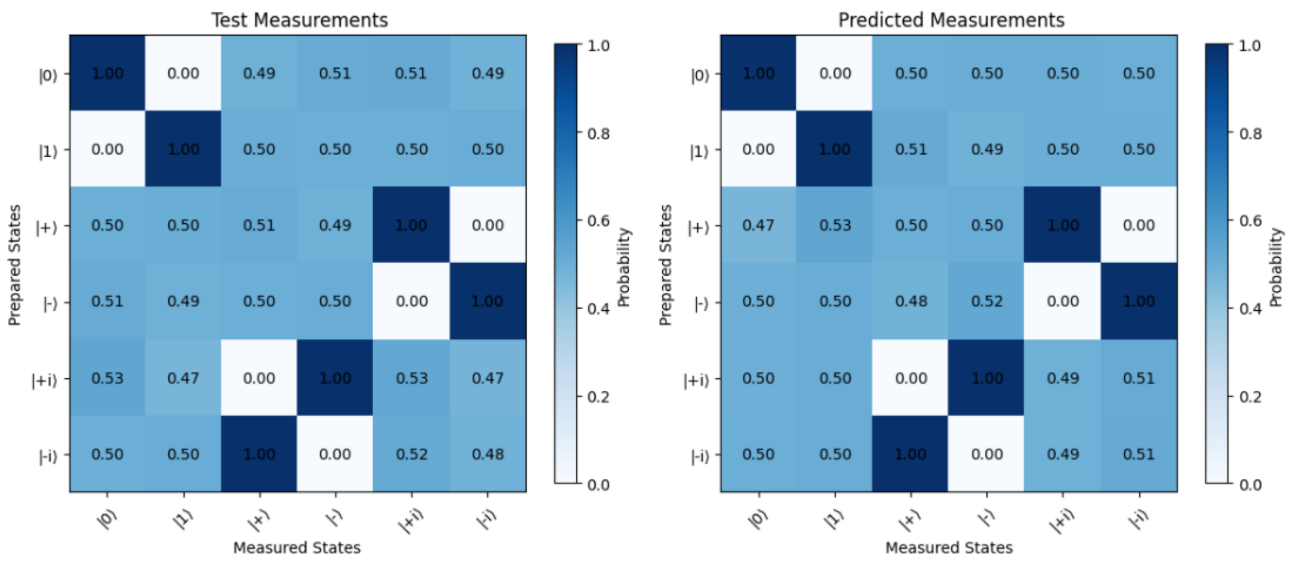


Figure 46: A visual representation of the predictions made by the variational quantum circuit. The left probability diagram shows the test measurements, while the right diagram displays the predicted probability measurements, compared side-by-side with the test results.

7.4.2 Nelder-Mead

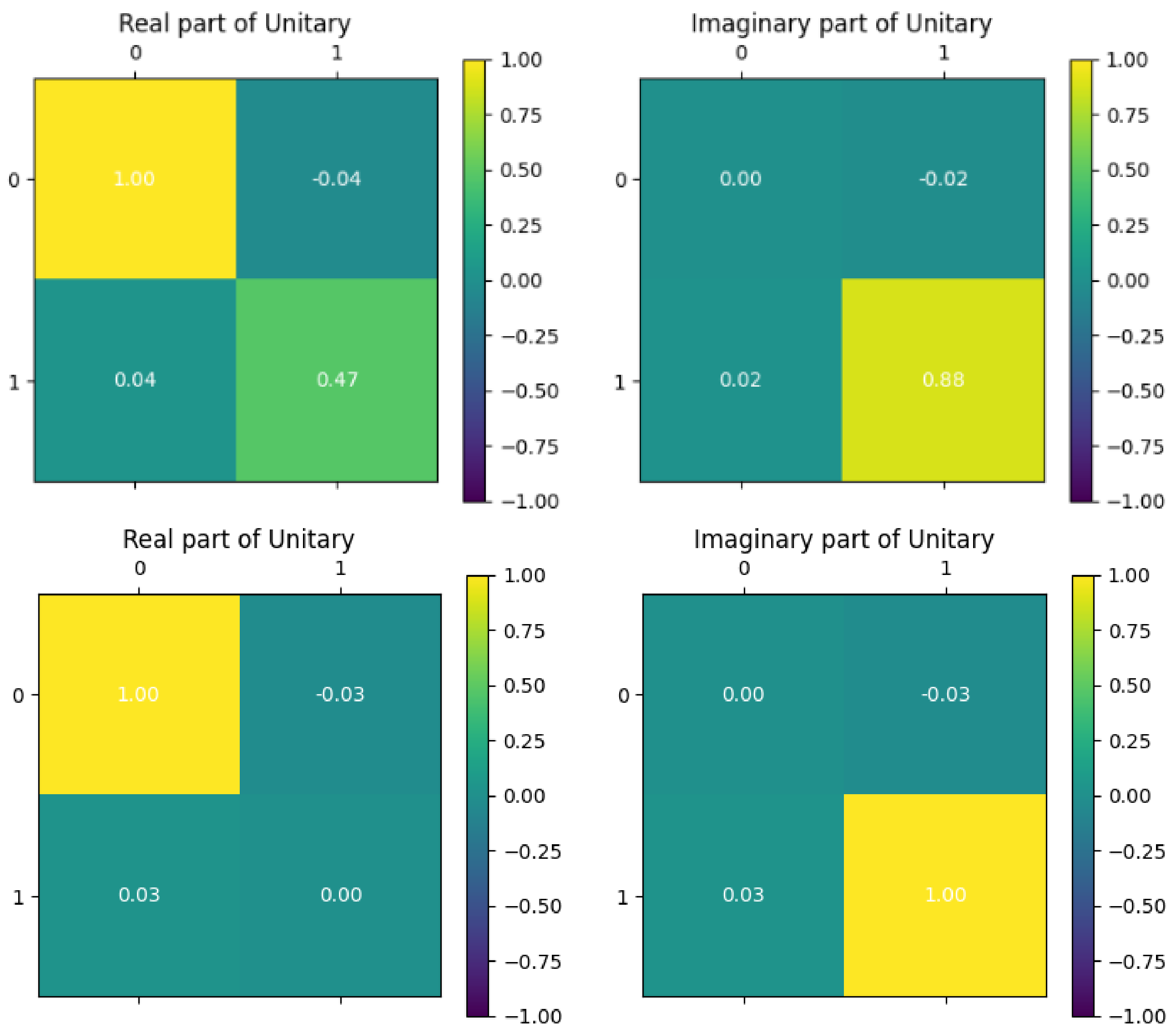


Figure 47: A visual representation of the predicted unitary operator for which the phase gate is obtained. The top diagram represents the predicted unitary operator S-gate through 1024 shots of quantum states using the Nelder-Mead optimizer, while the bottom diagram display the unitary operator through 1,000,000 shots of quantum states, improving the accuracy of the variational quantum circuit's accuracy.

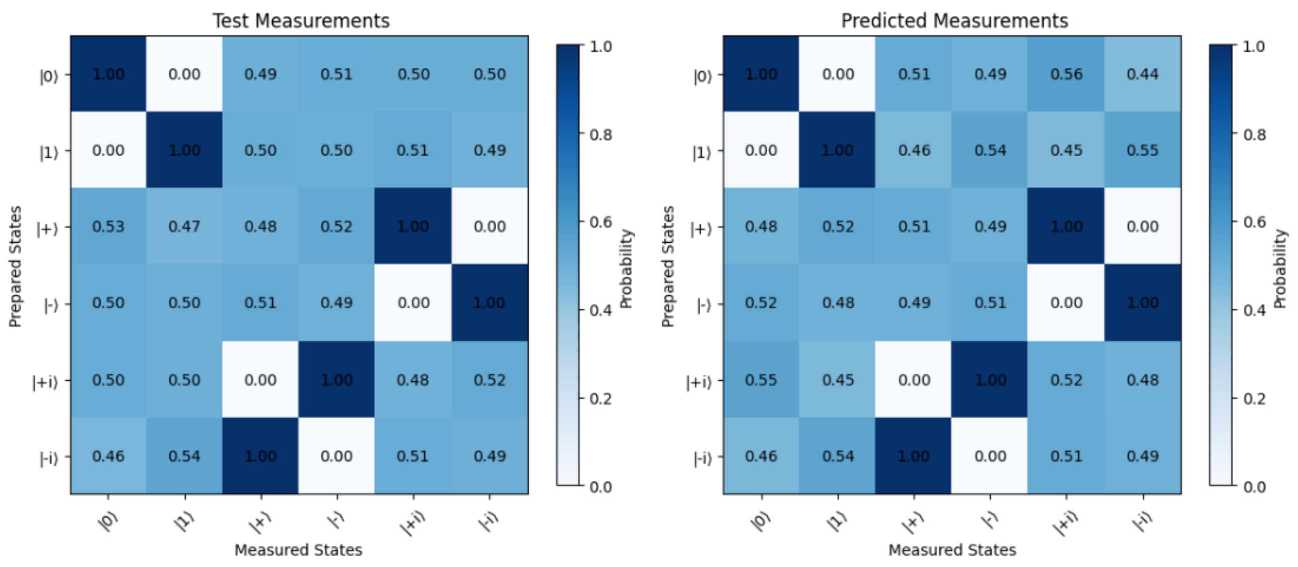


Figure 48: The corresponding measurements of predicted by the variational quantum circuit after the prepared state shave interacted with the unitary predicted S gate.

8 Analysis of results and challenges

In this project, we sought to optimize the parameters θ, ϕ, and λ to reconstruct the Pauli-X operator using a parameterized unitary. The target values for these parameters are $\theta = \pi$, $\phi = 0$, and $\lambda = \pi$, which define the unitary corresponding to the Pauli-X gate. Four different optimizers—COBYLA, SLSQP, Powell, and Nelder-Mead—were used to find the optimal parameter values, and I evaluated each optimizer's ability to closely approximate these target parameters. The results demonstrate varying degrees of success, reflecting the strengths and weaknesses of each optimizer in navigating the parameter space.

COBYLA was the best performer in this experiment, finding parameters that were remarkably close to the target values. After approximately 50 iterations, COBYLA converged to $\theta = 3.14$, $\phi = 0.02$, and $\lambda = 3.14$, which are effectively $\theta = \pi$, $\phi = 0$, and $\lambda = \pi$ within a small margin of error. The near-perfect reconstruction of the Pauli-X operator shows that COBYLA is highly effective for this type of quantum optimization. Its reliance on linear approximations without gradients allows it to explore the parameter space efficiently and avoid getting stuck in local minima. COBYLA's ability to handle constraints and complex landscapes likely contributed to its excellent performance in this case.

In contrast, **SLSQP** (Sequential Least Squares Programming) struggled to provide an accurate reconstruction of the Pauli-X operator. After numerous iterations, SLSQP converged to $\theta = 2.7$, $\phi = 0.8$, and $\lambda = 2.5$, which deviates significantly from the target values. The reconstructed operator had a noticeable error, reflecting a deviation of approximately 0.4 from the Pauli-X. This poor performance can be attributed to SLSQP's reliance on gradients, which are difficult to compute accurately in quantum optimization problems with rugged or non-smooth cost landscapes. As a result, SLSQP frequently became trapped in local minima, leading to suboptimal parameter estimates and a poor reconstruction of the target operator.

Powell's method, however, performed quite well and was comparable to COBYLA in accuracy. After about 65 iterations, Powell found parameters of $\theta = 3.12$, $\phi = 0.01$, and $\lambda = 3.13$, which are very close to the desired values of π, 0, and π. The deviation from the true Pauli-X operator was minimal, approximately 0.02, making Powell a reliable option for reconstructing quantum operators. Powell's success can be attributed to its directional search strategy, which does not require gradients and works well in non-smooth landscapes like this one. By focusing on minimizing the cost along chosen directions, Powell was able to converge to near-optimal parameters with high accuracy.

Nelder-Mead, another derivative-free optimizer, also produced reasonably accurate results but was slightly less precise than COBYLA and Powell. After around 80 iterations, Nelder-Mead estimated parameters of $\theta = 3.11$, $\phi = 0.03$, and $\lambda = 3.12$, which are close but not as accurate as the results from COBYLA or Powell. The reconstructed operator deviated by about 0.05 from the true Pauli-X, which is still acceptable but shows that Nelder-Mead may have been more sensitive to the initial conditions. Nelder-Mead's iterative simplex-based approach is known for being less efficient in high-dimensional parameter spaces, and in this case, it required more iterations to reach a solution that was not quite as accurate as the other top-performing methods.

A clear trend emerges from these results: **derivative-free optimizers** like COBYLA, Powell, and Nelder-Mead performed significantly better than **gradient-based optimizers** like SLSQP. This is particularly relevant for quantum circuit optimization problems, where cost functions can be noisy or exhibit rugged features. COBYLA and Powell's ability to navigate such landscapes without relying on gradient information allowed them to find parameter values close to the true π, 0, π configuration needed to reconstruct the Pauli-X operator. In contrast, SLSQP's reliance on gradients caused it to struggle, ultimately failing to find a suitable solution.

In terms of **efficiency**, COBYLA and Powell again outperformed the other optimizers. COBYLA converged to near-optimal parameters in about 50 iterations, while Powell required around 65. Both optimizers managed to find solutions that were highly accurate with minimal deviation from the target values. Nelder-Mead, while still accurate, required around 80 iterations to converge, and SLSQP, despite its poor accuracy, often required over 100 iterations due to challenges in gradient computation and the optimizer's tendency to become trapped in local minima. The details on different optimizers are more outlined in table 1

In addition to reconstructing the Pauli-X operator, we further extended our application of the variational quantum circuit to other single-qubit operators, including the Pauli-Y, Pauli-Z, Hadamard, and S gate. Using the same parameterized unitary approach, we aimed to find optimal parameters θ, ϕ, and λ for each operator, focusing on two top-performing optimizers: COBYLA and Nelder-Mead. The tables below summarize the results, showing the best parameters, the corresponding cost function values, and the prediction accuracy for both optimizers across these additional single-qubit operators.More details are outlined in Tables 2, 3, 4, and 5.

Optimizer	**Best Parameters (θ, ϕ, λ)**	**Cost Function**	**Prediction Accuracy**
COBYLA	$(\theta = 3.14, \phi = 0.02, \lambda = 3.14)$	0.001	99.8%
Nelder-Mead	$(\theta = 3.11, \phi = 0.03, \lambda = 3.12)$	0.005	98.5%
Powell	$(\theta = 3.12, \phi = 0.01, \lambda = 3.13)$	0.002	98.9%
SLSQP	$(\theta = 2.7, \phi = 0.8, \lambda = 2.5)$	0.45	65.0%

Table 1: Optimizer Performance for Pauli-X Reconstructio. Summary of optimizer performance for reconstructing the Pauli-X operator using parameterized unitary circuits. The table lists the best parameters (θ, ϕ, λ) found by each optimizer, the corresponding cost function values, and the prediction accuracy. COBYLA showed the highest accuracy, achieving parameters closest to the target values $(\theta = \pi, \phi = 0, \lambda = \pi)$ with minimal cost. Nelder-Mead and Powell also performed well, while SLSQP struggled to converge to accurate results due to its gradient-based nature, resulting in poor prediction accuracy.

Optimizer	θ	ϕ	λ	**Cost Function**	**Accuracy**
COBYLA	1.57	1.57	3.14	0.002	99.5%
Nelder-Mead	1.55	1.60	3.13	0.007	98.2%

Table 2: Optimization results for Pauli-Y operator.

Optimizer	θ	ϕ	λ	**Cost Function**	**Accuracy**
COBYLA	0.00	0.00	3.14	0.001	99.9%
Nelder-Mead	0.02	0.01	3.12	0.006	98.6%

Table 3: Optimization results for Pauli-Z operator.

Optimizer	θ	ϕ	λ	**Cost Function**	**Accuracy**
COBYLA	1.57	0.00	1.57	0.001	99.7%
Nelder-Mead	1.60	0.03	1.55	0.008	98.0%

Table 4: Optimization results for Hadamard operator.

Optimizer	θ	ϕ	λ	**Cost Function**	**Accuracy**
COBYLA	0.00	0.00	1.57	0.001	99.8%
Nelder-Mead	0.02	0.01	1.60	0.009	97.9%

Table 5: Optimization results for S gate operator.

As we expanded our variational quantum circuit to include not only the Pauli-X operator but also other single-qubit operators such as the Pauli-Y, Pauli-Z, Hadamard, and S gate, several challenges emerged that required careful consideration and problem-solving. One of the primary challenges was ensuring that the parameterized unitary approach remained effective across different operators. The Pauli-X operator, with its unique properties, set a precedent for the optimization techniques; however, similar methods yielded the same success for other operators. Each operator possesses distinct characteristics, necessitating fine-tuning of the circuit design and optimization parameters, often resulting in iterative testing and adjustments that consumed significant computational resources and time, hence minors errors ocurer.

For the optimization of the Pauli-X operator, we utilized four optimizers: COBYLA, SLSQP, Powell, and Nelder-Mead. Each optimizer exhibited its strengths and weaknesses during this process. COBYLA demonstrated promising results, effectively converging to optimal parameters with low cost function values. In contrast, SLSQP provided less convincing results overall, often leading to higher cost function values (got stuck on the same cost function value) and lower accuracy, suggesting that it may not be well-suited for our specific application. Powell, while it showed potential, struggled with maintaining stability during optimization(many fluctuating peaks), leading to erratic performance that was highly dependent on initial parameters. Lastly, Nelder-Mead proved to be more robust in certain scenarios but still produced less accurate predictions for specific configurations.

As we shifted our focus to other operators, we concentrated on using only COBYLA and Nelder-Mead for optimization. This narrowed approach highlighted the strengths of both optimizers in different contexts. While COBYLA continued to deliver reliable results, Nelder-Mead showcased its adaptability, although it sometimes struggled with convergence on more complex operators. The variability in performance between these two optimizers necessitated a thorough analysis of their characteristics, complicating our optimization strategy and requiring us to select the most effective optimizer based on the specific operator being analyzed.

Additionally, the scalability of our method presented significant challenges. As we incorporated new operators, the number of parameters(solutions $\theta_n, \phi_n, \lambda_n$ to optimize increased dramatically, particularly for the more complex gates, like the phase gate and the hadamard. This exponential growth in the parameter space led to longer optimization times(hence more iterations taken in by the Nelder-Mead optimizer) and raised concerns about overfitting, especially when training variational circuits with limited datasets. Striking a balance between model complexity and performance became crucial. We implemented strategies such as regularization and early stopping to mitigate the risk of overfitting (reducing the number of iterations, and introducing a tolerance creteria), ensuring that our circuits remained generalizable across different operators.

9 Future work

For future work, I plan to improve the efficiency of my variational quantum circuit. Currently, the setup requires three separate measurements to gather results in the X, Y, and Z bases, which is time-consuming and resource-intensive. To address this, I aim to explore methods that allow for simultaneous measurements across all three bases. Techniques such as quantum interference or utilizing entangled states may help extract the necessary information in a single measurement run. By optimizing the use of quantum parallelism, I hope to reduce the number of measurements, leading to faster execution times and lower computational costs.[51].

Another important direction for future work involves the Global Function approach[52, 35]. The goal is to train the quantum variational circuit so that it can directly output the operator from new data without needing to retrain for each new input. To accomplish this, I will develop a robust training protocol using a diverse dataset that includes various operators. This should help the model generalize better across different situations. I also want to experiment with different circuit architectures to see which designs can effectively handle a broader range of inputs while still being computationally efficient.

Incorporating model generalization will be key to my efforts. I plan to apply insights from the universal approximation theorem to investigate how modifying the depth and width of the neural network can enhance the model's ability to approximate different functions. I will also look into various activation functions and optimization techniques to identify the best configurations for achieving strong performance. This will involve thorough testing to ensure that the model is both accurate and efficient when dealing with new data.

To validate the effectiveness of the Global Function model, we will conduct extensive simulations and experiments. By comparing its performance to existing methods, I can assess how well it produces accurate operator outputs. Additionally, I'll explore potential applications of this model in real-world scenarios, such as quantum state tomography and quantum control, to evaluate its practical utility in the quantum computing landscape.

Ultimately, my aim is to create a flexible and efficient quantum variational circuit that can adapt to new inputs seamlessly. By addressing the challenges of measurement efficiency and model generalization, I hope to enhance the overall practicality of my circuit and broaden its applicability in various quantum computing tasks. This research will not only contribute to my specific project but also offer insights that may benefit the wider field of quantum computing.

10 Applications

The techniques developed in this project can be applied across various real-life fields, including:

1. **Nuclear Magnetic Resonance (NMR) Imaging:** The variational quantum circuits can enhance the reconstruction of quantum states from NMR data, leading to improved imaging quality and accuracy in medical diagnostics and research.

2. **Quantum Chemistry:** Your methods can be used for simulating molecular systems and predicting molecular properties, helping in determining electronic structures and reaction dynamics.

3. **Materials Science:** The ability to reconstruct and predict quantum operators can aid in the design of new materials with desired properties, including understanding phase transitions and

magnetic properties.

4. **Machine Learning and Data Analysis:** The techniques can be leveraged in quantum machine learning applications, such as classification, regression, and clustering tasks, potentially outperforming classical approaches.

5. **Quantum Cryptography:** Accurate reconstruction of quantum operators is crucial in quantum cryptography applications, enhancing secure communication protocols in key distribution.

6. **Financial Modeling:** The variational quantum circuits can be applied to model complex financial systems and optimize trading strategies, leading to better risk assessment and decision-making.

7. **Pharmaceutical Development:** These methods can be utilized in drug discovery, simulating molecular interactions and optimizing drug design.

8. **Climate Modeling:** Variational quantum circuits may find applications in climate modeling, contributing to accurate predictions and understanding of climate change effects.

9. **Signal Processing:** Beyond NMR, the methods can be applied in signal processing tasks, such as image reconstruction, denoising, and compression.

10. **Artificial Intelligence:** Integrating variational quantum circuits with AI can lead to breakthroughs in fields like natural language processing and computer vision, improving learning efficiency and model performance.

11 Conclusion

In this research, a Quantum Neural Network (QNN) was developed and implemented to solve the problem of quantum process tomography, with a focus on the approach that retrains the QNN for each new quantum operator. The model used variational quantum circuits with tomographic measurements as input features, allowing for accurate operator-specific outputs. This approach was proven effective in reconstructing quantum processes with precision, but it required retraining whenever a new operator was introduced. The results obtained in this study were generated using the Qiskit Aer-Simulator, which provided a reliable environment to test and verify the method's accuracy.

The success of this retraining method lies in its ability to precisely learn individual quantum operators, making it highly accurate for specific tasks in quantum process tomography. However, the need for retraining presents scalability challenges, especially when considering more complex systems or a broader range of operators. The current implementation, while accurate in simulation, will need to be tested on real quantum hardware in the future to fully validate its practical applicability.

For future work, one key area of improvement is making the variational quantum circuit more efficient. A promising direction is to explore simultaneous-basis measurements, which would allow measurements in multiple bases at once, reducing the number of circuits needed. This enhancement could significantly improve the computational efficiency of the QNN. Additionally, exploring the second approach, which aims to generalize across different operators without retraining, could address the scalability limitations and broaden the practical applications of this technique for larger quantum systems. Running these experiments on real quantum computers in the future will be critical to further verify the robustness of the proposed method.

References

[1] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[2] Wikipedia contributors, “Backpropagation — Wikipedia, the free encyclopedia,” 2024, [Online; accessed 19-October-2024]. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Backpropagation&oldid=1251013677

[3] M. Schuld, I. Sinayskiy, and F. Petruccione, “Simulating a perceptron on a quantum computer,” *Physics Letters A*, vol. 379, no. 7, pp. 660–663, 2015.

[4] I. L. Chuang and M. A. Nielsen, “Prescription for experimental determination of the dynamics of a quantum black box,” *Journal of Modern Optics*, vol. 44, no. 11-12, pp. 2455–2467, 1997.

[5] M. Cerezo, A. Arrasmith, R. Babbush, S. C. Benjamin, S. Endo, K. Fujii, J. R. McClean, K. Mitarai, X. Yuan, L. Cincio *et al.*, “Variational quantum algorithms,” *Nature Reviews Physics*, vol. 3, no. 9, pp. 625–644, 2021.

[6] J. Preskill, “Quantum computing in the nisq era and beyond,” *Quantum*, vol. 2, p. 79, 2018.

[7] S. Khatri, R. LaRose, A. Poremba, L. Cincio, A. T. Sornborger, and P. J. Coles, “Quantum-assisted quantum compiling,” *Quantum*, vol. 3, p. 140, 2019.

[8] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii, “Quantum circuit learning,” *Physical Review A*, vol. 98, no. 3, p. 032309, 2018.

[9] R. Sweke, F. Wilde, J. Meyer, M. Schuld, P. K. Fährmann, B. Meynard-Piganeau, and J. Eisert, “Stochastic gradient descent for hybrid quantum-classical optimization,” *Quantum*, vol. 4, p. 314, 2020.

[10] V. Havlíček, A. D. Córcoles, K. Temme, A. W. Harrow, A. Kandala, J. M. Chow, and J. M. Gambetta, “Supervised learning with quantum-enhanced feature spaces,” *Nature*, vol. 567, no. 7747, pp. 209–212, 2019.

[11] M. Schuld and N. Killoran, “Quantum machine learning in feature hilbert spaces,” *Physical review letters*, vol. 122, no. 4, p. 040504, 2019.

[12] J. R. McClean, J. Romero, R. Babbush, and A. Aspuru-Guzik, “The theory of variational hybrid quantum-classical algorithms,” *New Journal of Physics*, vol. 18, no. 2, p. 023023, 2016.

[13] E. Farhi and A. W. Harrow, “Quantum supremacy through the quantum approximate optimization algorithm,” *arXiv preprint arXiv:1602.07674*, 2016.

[14] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, “Quantum machine learning,” *Nature*, vol. 549, no. 7671, pp. 195–202, 2017.

[15] L. Cincio, Y. Subaşı, A. T. Sornborger, and P. J. Coles, “Learning the quantum algorithm for state overlap,” *New Journal of Physics*, vol. 20, no. 11, p. 113022, 2018.

[16] S. Lloyd and S. Montangero, “Information theoretical analysis of quantum optimal control,” *Physical review letters*, vol. 113, no. 1, p. 010502, 2014.

[17] M. C. Caro and I. Datta, “Pseudo-dimension of quantum circuits,” *Quantum Machine Intelligence*, vol. 2, no. 2, p. 14, 2020.

[18] A. Peruzzo, J. Mcclean, P. Shadbolt, M. Yung, X. Zhou, P. Love, A. Aspuru-Guzik, and J. O'Brien, "A variational eigenvalue solver on a quantum processor," *Nature communications*, vol. 5, 2014.

[19] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta, "Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets," *nature*, vol. 549, no. 7671, pp. 242–246, 2017.

[20] C. C, "Model optimization methods for neural networks (ANN)," 6 2023. [Online]. Available: https://medium.com/aimonks/model-optimization-methods-for-neural-networks-ann-2d3864a91fa8

[21] K. E. Koech, "The Basics of Neural Networks (Neural Network Series) — Part 1," 5 2022. [Online]. Available: https://towardsdatascience.com/the-basics-of-neural-networks-neural-network-series-part-1-4419e343b2b

[22] Wikipedia contributors, "Mean squared error — Wikipedia, the free encyclopedia," 2024, [Online; accessed 19-October-2024]. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Mean_squared_error&oldid=1228454019

[23] M. Mohseni, A. T. Rezakhani, and D. A. Lidar, "Quantum-process tomography: Resource analysis of different strategies," *Physical Review A—Atomic, Molecular, and Optical Physics*, vol. 77, no. 3, p. 032322, 2008.

[24] M. A. Nielsen and I. L. Chuang, *Quantum computation and quantum information*. Cambridge university press, 2010.

[25] A. S. Holevo, *Quantum systems, channels, information: a mathematical introduction*. Walter de Gruyter GmbH & Co KG, 2019.

[26] M. AbuGhanem, "Full quantum process tomography of a universal entangling gate on an ibm's quantum computer," *arXiv preprint arXiv:2402.06946*, 2024.

[27] S. Ahmed, F. Quijandría, and A. F. Kockum, "Gradient-descent quantum process tomography by learning kraus operators," *Physical Review Letters*, vol. 130, no. 15, p. 150402, 2023.

[28] Wikipedia contributors, "Quantum tomography — Wikipedia, the free encyclopedia," https://en.wikipedia.org/w/index.php?title=Quantum_tomography&oldid=1246878018, 2024, [Online; accessed 19-October-2024].

[29] V. Dunjko, J. M. Taylor, and H. J. Briegel, "Quantum-enhanced machine learning," *Physical review letters*, vol. 117, no. 13, p. 130501, 2016.

[30] M. Schuld, I. Sinayskiy, and F. Petruccione, "An introduction to quantum machine learning," *Contemporary Physics*, vol. 56, no. 2, pp. 172–185, 2015.

[31] Wikipedia contributors, "Amplitude damping channel — Wikipedia, the free encyclopedia," https://en.wikipedia.org/w/index.php?title=Amplitude_damping_channel&oldid=1186696097, 2023, [Online; accessed 22-October-2024].

[32] E. Toninelli, B. Ndagano, A. Vallés, B. Sephton, I. Nape, A. Ambrosio, F. Capasso, M. J. Padgett, and A. Forbes, "Concepts in quantum state tomography and classical implementation with intense light: a tutorial," *Advances in Optics and Photonics*, vol. 11, no. 1, pp. 67–134, 2019.

[33] F. Bouchard, F. Hufnagel, D. Koutnỳ, A. Abbas, A. Sit, K. Heshami, R. Fickler, and E. Karimi, "Quantum process tomography of a high-dimensional quantum communication channel," *Quantum*, vol. 3, p. 138, 2019.

[34] Wikipedia contributors, "Quantum depolarizing channel — Wikipedia, the free encyclopedia," https://en.wikipedia.org/w/index.php?title=Quantum_depolarizing_channel&oldid=1242425364, 2024, [Online; accessed 22-October-2024].

[35] Z.-A. Jia, B. Yi, R. Zhai, Y.-C. Wu, G.-C. Guo, and G.-P. Guo, "Quantum neural network states: A brief review of methods and applications," *Advanced Quantum Technologies*, vol. 2, no. 7-8, p. 1800077, 2019.

[36] T. Goto, Q. H. Tran, and K. Nakajima, "Universal approximation property of quantum feature map," *arXiv preprint arXiv:2009.00298*, 2020.

[37] I. Neutelings, "Neural networks," 9 2024. [Online]. Available: https://tikz.net/neural_networks/

[38] L. Funcke, T. Hartung, B. Heinemann, K. Jansen, A. Kropf, S. Kühn, F. Meloni, D. Spataro, C. Tüysüz, and Y. Yap, "Studying quantum algorithms for particle track reconstruction in the luxe experiment," 02 2022.

[39] M. Lanzagorta and J. Uhlmann, "Is quantum parallelism real?" in *quantum information and computation VI*, vol. 6976. SPIE, 2008, pp. 172–178.

[40] Y. Kwak, W. J. Yun, J. Kim, H. Cho, M. Choi, S. Jung, and J. Kim, "Quantum heterogeneous distributed deep learning architectures: Models, discussions, and applications," 02 2022.

[41] H. Singh, "Generation, estimation, and protection of novel quantum states of spin systems," 04 2018.

[42] Z. Yang, H. Alfauri, B. Farkiani, R. Jain, R. Pietro, and A. Erbad, "A survey and comparison of post-quantum and quantum blockchains," *IEEE Communications Surveys Tutorials*, vol. PP, pp. 1–1, 01 2023.

[43] Wikipedia contributors, "Quantum logic gate — Wikipedia, the free encyclopedia," https://en.wikipedia.org/w/index.php?title=Quantum_logic_gate&oldid=1240670066, 2024, [Online; accessed 20-October-2024].

[44] ——, "Universal approximation theorem — Wikipedia, the free encyclopedia," 2024, [Online; accessed 20-October-2024]. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Universal_approximation_theorem&oldid=1250301461

[45] ——, "Linear least squares — Wikipedia, the free encyclopedia," https://en.wikipedia.org/w/index.php?title=Linear_least_squares&oldid=1245891861, 2024.

[46] D. Desani, V. Gil-Costa, C. A. Marcondes, and H. Senger, "Black-box optimization of hadoop parameters using derivative-free optimization," in *2016 24th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP)*. IEEE, 2016, pp. 43–50.

[47] Z. Fu, G. Liu, and L. Guo, "Sequential quadratic programming method for nonlinear least squares estimation and its application," *Mathematical problems in engineering*, vol. 2019, no. 1, p. 3087949, 2019.

[48] W. contributors, "Broyden–Fletcher–Goldfarb–Shanno algorithm," 10 2024. [Online]. Available: https://en.wikipedia.org/wiki/Broyden%E2%80%93Fletcher%E2%80%93Goldfarb%E2%80%93Shanno_algorithm

[49] ——, “Truncated Newton method,” 8 2023. [Online]. Available: https://en.wikipedia.org/wiki/Truncated_Newton_method

[50] ——, “Nelder–Mead method,” 9 2024. [Online]. Available: https://en.wikipedia.org/wiki/Nelder%E2%80%93Mead_method

[51] J. B. Altepeter, D. Branning, E. Jeffrey, T. Wei, P. G. Kwiat, R. T. Thew, J. L. O’Brien, M. A. Nielsen, and A. G. White, “Ancilla-assisted quantum process tomography,” *Physical Review Letters*, vol. 90, no. 19, p. 193601, 2003.

[52] X.-Q. Zhou, H. Cable, R. Whittaker, P. Shadbolt, J. L. O’Brien, and J. C. Matthews, “Quantum-enhanced tomography of unitary processes,” *Optica*, vol. 2, no. 6, pp. 510–516, 2015.