Physics-Informed Learning for Scalable Single-Qubit Control Simulations

Diego Alducin, Ph.D.

November 18, 2024

Abstract

This study evaluates Physics-Informed Neural Networks (PINNs) and data-driven neural networks (QubiNNs) for single-qubit quantum control. While QubiNN achieved a slightly better accuracy and fidelity in predicting expectation values, both models struggled to approximate optimal control fields, with the PINN offering better performance. QubiNN also demonstrated higher computational efficiency due to its simpler architecture, achieving greater speedup compared to the PINN. These results highlight the trade-offs between data-driven and physics-informed approaches, emphasizing the potential for hybrid models to balance accuracy, physical consistency, and efficiency in quantum control simulations.

1 Introduction

Neural networks have proven to be powerful tools for solving complex physics problems [1] and partial differential equations (PDEs) [2, 3, 4, 5] across various domains, including fluid dynamics [6], heat transfer [7], and quantum mechanics [8]. By leveraging their ability to approximate high-dimensional functions and learn non-linear relationships from data, neural networks have been applied to model complex physical systems with remarkable accuracy. In particular, advancements in deep learning have enabled the development of architectures that not only fit data but also incorporate governing physical laws, paving the way for more robust and interpretable solutions.

Physics-Informed Neural Networks (PINNs) have emerged as a powerful tool for solving high-dimensional partial differential equations (PDEs) and modeling dynamical systems by embedding physical laws directly into the learning process [9]. This framework has gained traction in quantum computing and control, where accurately modeling quantum dynamics is critical for tasks such as state preparation, quantum gates, and optimization of control pulses. Recent studies have explored PINNs for quantum systems, including continuous variable quantum computing [10], eigenvalue problems [11], and quantum gates [12]. However, the potential of PINNs to approximate control fields and expectation values in qubit systems remains an open challenge.

Data-driven neural networks have also demonstrated remarkable flexibility and efficiency in approximating mappings in complex systems. These models excel when high-quality

training data is available, offering computational efficiency and simplicity [13]. However, their lack of inherent physical constraints can lead to inaccurate or unphysical predictions in scenarios where the data distribution is incomplete or complex dynamics are involved.

This work bridges these approaches by evaluating a PINN and a data-driven feed forward neural networks (QubiNNs) in predicting control fields and expectation values for single-qubit closed systems. While the PINN incorporates the Schrödinger equation as a physics-based constraint, the QubiNN relies solely on data-driven learning. By benchmarking both models against QuTiP [14] simulations, this study explores the trade-offs between physical consistency, computational efficiency, and predictive accuracy.

The results indicate that QubiNN consistently outperforms the PINN in predicting expectation values due to its ability to learn data-driven patterns efficiently. In contrast, the PINN provides slight improvements in control field predictions, although both models face challenges in this area. QubiNN also achieves superior computational efficiency, making it better suited for real-time applications. These findings highlight the need for hybrid models that combine the strengths of physics-informed and data-driven approaches, offering both accuracy and scalability for quantum control tasks.

2 Single Qubit Dynamics

The dynamics of a single qubit transitioning from one basis state to another can be visualized using the Bloch sphere representation. The Bloch sphere [15] is a geometrical representation of the state space of a qubit (Figure 1), where any pure state of a qubit corresponds to a point on the surface of a unit sphere.

In the Bloch sphere:

- The north pole typically represents the $|0\rangle$ state
- The south pole represents the $|1\rangle$ state
- Any superposition state is represented by a point on the sphere's surface

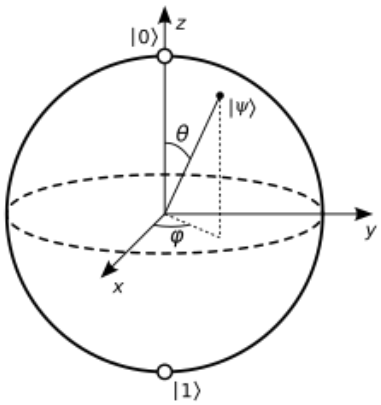

Figure 1: Bloch Sphere (wikimedia commons).

When a qubit transitions from one basis state to another, it can be visualized as the state vector moving across the surface of the Bloch sphere. This movement is governed by the Hamiltonian of the system, which determines the path the qubit state takes.

For example, consider a qubit starting in the $|0\rangle$ state (north pole) and transitioning to the —1⟩ state (south pole). This transition could occur through the following steps:

1. The qubit starts at the north pole ($|0\rangle$ state)
2. A rotation around the x-axis (caused by applying an X-gate or a π pulse around the x-axis) would move the state vector down the y-z plane
3. Halfway through this rotation, the qubit would be in an equal superposition state $(|0\rangle + |1\rangle)/\sqrt{2}$, represented by a point on the equator of the Bloch sphere
4. Completing the rotation would bring the state vector to the south pole, representing the $|1\rangle$ state

The time-dependent Schrödinger equation (Equation 1) describes how this state evolves over time.

$$i\hbar\frac{\delta}{\delta(t)}|\psi(t)\rangle = H(t)|\psi(t)\rangle, \tag{1}$$

The control fields $\Omega_x(t)$, $\Omega_y(t)$, and $\Omega_z(t)$ in the Hamiltonian (Equation 2) correspond to rotations around the x, y, and z axes of the Bloch sphere, respectively.

$$H(t) = \frac{\hbar}{2}(\Omega_x(t)\sigma_x + \Omega_y(t)\sigma_y + \Omega_z(t)\sigma_z), \tag{2}$$

The interaction between the control fields and Pauli operators in the Hamiltonian (Equation 2) is fundamental to understanding qubit manipulation. Each term in the Hamiltonian represents a rotation around a specific axis of the Bloch sphere:

- $\Omega_x(t)\sigma_x$: This term corresponds to rotations around the x-axis. When $\Omega_x(t)$ is non-zero, it induces rotations that can flip the qubit between $|0\rangle>$ and $|1\rangle>$ states.
- $\Omega_y(t)\sigma_y$: This term generates rotations around the y-axis. It can create superpositions with complex phases between $|0\rangle>$ and $|1\rangle>$ states.
- $\Omega_z(t)\sigma_z$: This term produces rotations around the z-axis. It affects the relative phase between $|0\rangle>$ and $|1\rangle>$ components of the qubit state.

The strength and time-dependence of each control field $\Omega_i(t)$ determine the rate and extent of rotation around its respective axis. By precisely controlling these fields qubit's state can be driven to any point on the Bloch sphere, enabling the implementation of arbitrary single-qubit gates and state preparations.

By carefully manipulating these control fields, it's possible to guide the qubit from any initial state to any desired final state, tracing a path on the surface of the Bloch sphere. This ability to precisely control the qubit's state is fundamental to quantum computing operations and forms the basis for implementing quantum gates and algorithms.

3 Qubit Quantum Control Simulations

To create a dataset suitable for training a neural network to predict control fields and expectation values for various target states, simulations of single-qubit quantum control implemented in QuTiP (Quantum Toolbox in Python) [14] were used. QuTiP is an open-source library for simulating quantum systems, offering tools for solving the dynamics of quantum states and operators in both closed and open systems. It provides efficient, high-level functions for performing quantum operations, including optimized control of quantum states, numerical solvers for quantum evolution, and visualization tools. In this work, QuTiP's control optimization algorithm and state evolution solvers were used to compute time-dependent control fields and expectation values that would guide the qubit's initial state toward various target transformations.

The data generation process employed the Gradient Ascent Pulse Engineering (GRAPE) algorithm to optimize control fields and the quantum state evolution tool, 'mesolve', to solve the time-dependent dynamics. This approach allows the optimal control field computation and extract state trajectories, providing a robust training dataset for a Physics-Informed Neural Network (PINN) designed to generate control fields and predict qubit evolution toward any desired target state.

3.1 Target Unitaries and Initial State

For each simulation, a target unitary transformation U was defined that maps the qubit's initial state $|0\rangle$ to a desired target state on the Bloch sphere. This target state is specified as a rotation (Equation 3 from $|0\rangle$ to a new orientation, parameterized by rotation angles θ and ϕ:

$$U = R_z(\phi)R_x(\theta), \tag{3}$$

where the individual rotations (Equations 4 about the z- and x-axes are defined as:

$$R_z(\phi) = e^{-i\frac{\phi}{2}\sigma_z} \quad \text{and} \quad R_x(\theta) = e^{-i\frac{\theta}{2}\sigma_x}, \tag{4}$$

This approach produces unitary transformations that represent a wide range of qubit states on the Bloch sphere. In practice, target states included basis states $|0\rangle$ and $|1\rangle$, as well as superpositions $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and $|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$. Random target states were also generated to ensure variety in the training dataset.

3.2 Control Field Optimization Using GRAPE

For each target unitary, a time-dependent control fields $u_x(t)$, $u_y(t)$, and $u_z(t)$ was optimized to drive the qubit from $|0\rangle$ to the desired target state. The optimization relied on the GRAPE (Gradient Ascent Pulse Engineering) algorithm as implemented in QuTiP's 'cy_grape_unitary' function. In the Hamiltonian formulation (Equations 5), the control fields apply to the Pauli operators σ_x, σ_y, and σ_z:

$$H(t) = u_x(t)\sigma_x + u_y(t)\sigma_y + u_z(t)\sigma_z, \tag{5}$$

The goal of GRAPE is to minimize the distance between the actual and target unitary transformations by maximizing fidelity (Equation 6). The fidelity F between the target unitary U and the achieved unitary U_{evolved} at the final time is computed as:

$$F = \left|\text{Tr}(U^{\dagger}U_{\text{evolved}})\right|^2, \tag{6}$$

The control fields u_x, u_y, and u_z were initialized as smoothed random functions and iteratively adjusted them to maximize fidelity. QuTiP's 'cy_grape_unitary' function allows efficient computation of the gradient-based adjustments to the control fields over a fixed number of time slices.

3.3 Quantum State Evolution and Expectation Values

With the optimized control fields obtained from GRAPE, the time evolution of the qubit state was calculated using QuTiP's 'mesolve' function, which solves the Lindblad master equation for quantum systems. For the closed system, the evolution is governed by the Schrödinger equation, and the Lindblad (dissipative) terms were set to zero (Equation 7). The qubit's density matrix $\rho(t)$ evolves under the Hamiltonian $H(t)$ as follows:

$$\frac{d\rho(t)}{dt} = -i[H(t), \rho(t)], \tag{7}$$

Using 'mesolve', $\rho(t)$ was computed over time (Equation 8), which allowed to extract expectation values for each Pauli operator σ_x, σ_y, and σ_z:

$$\langle\sigma_x\rangle = \text{Tr}(\rho(t)\sigma_x), \quad \langle\sigma_y\rangle = \text{Tr}(\rho(t)\sigma_y), \quad \langle\sigma_z\rangle = \text{Tr}(\rho(t)\sigma_z, \tag{8}$$

These expectation values provide a trajectory of the qubit's state on the Bloch sphere over time as it evolves toward the target.

3.4 Dataset Composition and Structure

For each target unitary, the following information was stored:

- **Control fields** $u_x(t)$, $u_y(t)$, and $u_z(t)$: Optimized time-dependent control fields over the evolution period.
- **Expectation values** $\langle\sigma_x\rangle$, $\langle\sigma_y\rangle$, and $\langle\sigma_z\rangle$: Time-dependent expectation values of the Pauli operators.
- **Target angles** θ and ϕ: Rotation angles defining the target state on the Bloch sphere.

This dataset, consisting of control fields, expectation values, and target angles, is used as input-output pairs for training a PINN and the QubiNN to predict control fields and expectation values for new target states. The data generation process, based on QuTiP's 'cy_grape_unitary' and 'mesolve' functions, provides a rigorous basis for guiding the qubit to various target states through optimized control

The training dataset was carefully constructed to represent a diverse set of target states on the Bloch sphere, ensuring that the model learns to generate control fields for a wide range of qubit transformations. The data includes both specific basis states and a variety of randomly generated target states, which together form a comprehensive set of training examples.

The training data includes all primary basis states as well as a generation of superposition states. Each of these states represents a unique orientation on the Bloch sphere and is visualized in Figure 2:

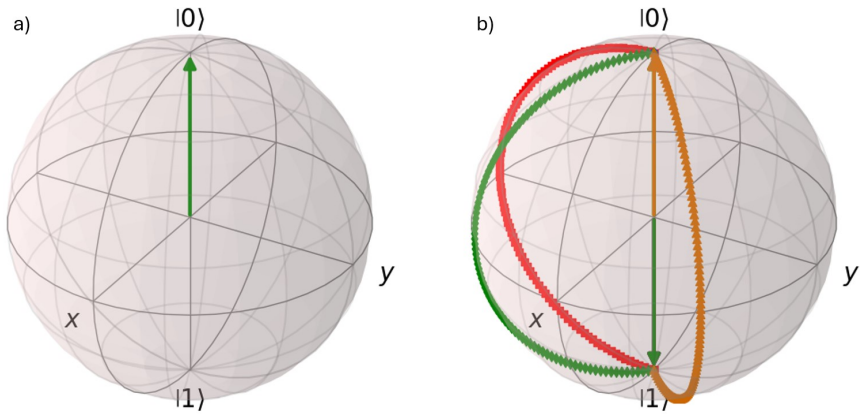

Figure 2: a) ψ_0 Initial state $|0\rangle$. b) Basis states and superposition with its generations across the x axis.

- $|0\rangle$: The qubit state pointing up along the z-axis, also the initial state before all simulations sttarted, $|\psi_0$.
- $|1\rangle$: The qubit state pointing down along the z-axis.
- $|+\rangle$: A superposition state generation across an the x-axis, defined as $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle+|1\rangle)$
- $|-\rangle$: A superposition state generation across an the -x-axis, defined as $|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$

The inclusion of these states provides reference points that guide the model in learning common transformations, especially those used in fundamental quantum operations. Figure 2 shows the Bloch sphere representations of these basis and superposition states, giving a visual reference of their spatial locations.

In addition to the basis and superposition states, 496 random target states distributed across the Bloch sphere, with their paths to reach each speficic state. Figure 3 shows a fraction (20%) of the generated target states and their paths. Each random state was specified by rotation angles θ and ϕ from an initial state $|0\rangle$ to the desired target orientation, effectively covering a wide range of qubit dynamics. These random target states provide diverse examples that enhance the model's generalization ability for arbitrary rotations on the Bloch sphere.

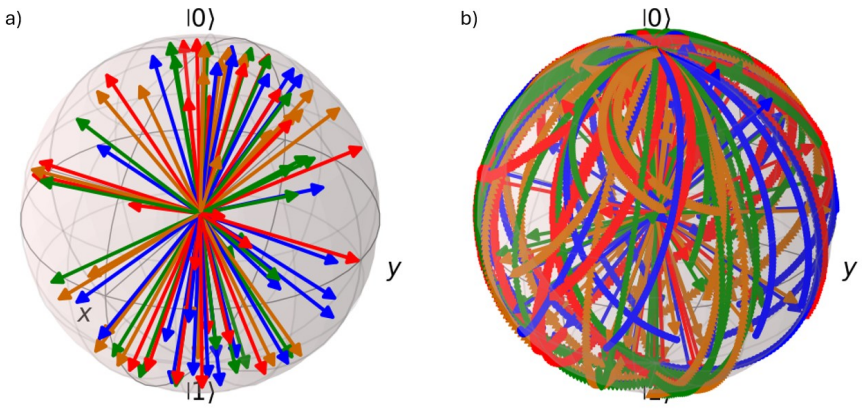

Figure 3: a) Vector states, b) all paths of to vectors from basis $|0\rangle$.

Histograms of the spherical coordinates (angles θ and ϕ) of the qubit vectors in the dataset are shown in Figure 4 . These histograms, display the density of states across different regions of the Bloch sphere. The polar angle θ measures the position relative to the z-axis. The histogram for θ values shows how states are spread between the north and south poles, providing insight into vertical state distribution. The azimuthal angle ϕ represents the angle around the z-axis. The histogram for ϕ values gives a view of how states are distributed horizontally around the Bloch sphere.

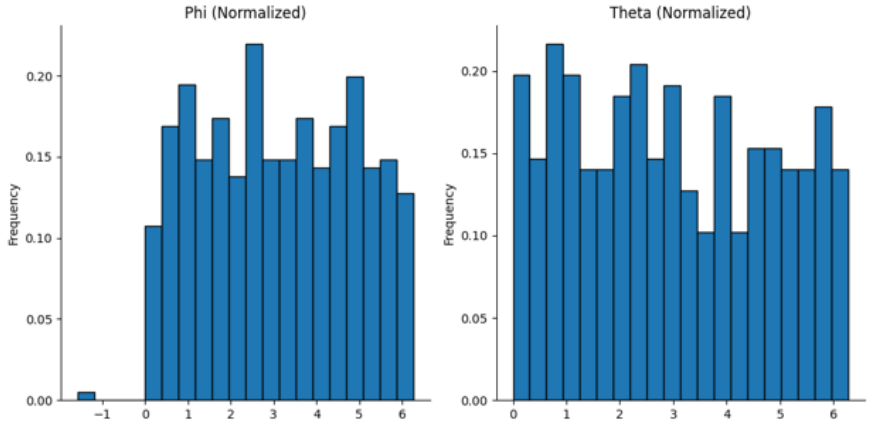

Figure 4: Angle histograms of θ and ϕ of all target states generated.

The histograms of the control fields u_z, u_y and u_z show the distribution of optimized field strengths across the training dataset, Figure 5. Each control field histogram provides insights into the relative intensity and variability of the fields applied along the x-, y-, and z-axes to drive the qubit toward various target states. These distributions indicate how often certain field strengths are utilized, reflecting the diversity and range of control efforts needed for different state transformations. The broader and more uniform distribution of field strengths suggests a rich variety of training examples, enabling the model to generalize well across different qubit dynamics.

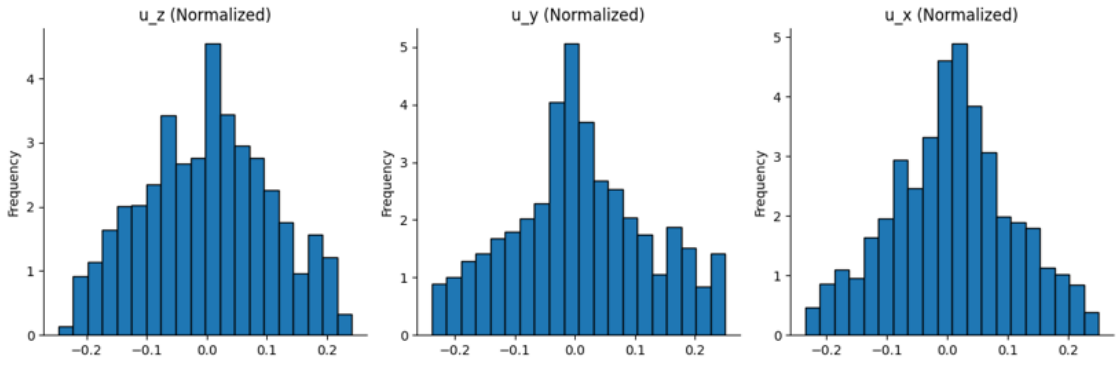

Figure 5: Histrograms of control field distributions normalized.

4 Physics-Informed Neural Network Architecture

The Physics-Informed Neural Network (PINN) is structured to simultaneously predict the control fields u_x, u_y, u_z and the expectation values of the Pauli operators $\langle\sigma_x\rangle, \langle\sigma_y\rangle, \langle\sigma_z\rangle$. These predictions ensure the accurate representation of the qubit's dynamics on the Bloch sphere, driven by both data and physics-based constraints.

The network receives as input three key parameters: the target state parameters θ and ϕ, which define the qubit's desired orientation on the Bloch sphere, and the normalized time t, representing the temporal evolution of the quantum state (Equation 9). It processes this input through a shared fully connected feed forward structure that learns a latent representation of the state evolution.

$$\mathbf{x} = \begin{bmatrix} \theta & \phi & t \end{bmatrix}, \tag{9}$$

The shared layer is followed by two specialized sub-networks. The first sub-network predicts the control fields u_x, u_y, u_z, while the second estimates the expectation values $\langle\sigma_x\rangle, \langle\sigma_y\rangle, \langle\sigma_z\rangle$. Each sub-network comprises three fully connected layers with ReLU activation functions. This modular design enables the PINN to simultaneously optimize both outputs while maintaining computational efficiency.

Equation 10 shows the output of the PINN is a six-dimensional vector:

$$\mathbf{y} = \begin{bmatrix} u_x & u_y & u_z & \langle\sigma_x\rangle & \langle\sigma_y\rangle & \langle\sigma_z\rangle \end{bmatrix}. \tag{10}$$

This representation captures the qubit's control and dynamical evolution at any given time.

To ensure the PINN adheres to the underlying physics, the training process employs a combined loss function. The data loss, defined as the Mean Squared Error (MSE) between the predicted and true values, captures the accuracy of the model's predictions relative to the training data. The physics-informed loss imposes the constraints of the Schrödinger equation on the time evolution of the expectation values. Specifically, the physics-based loss evaluates the residuals of the differential equations governing the dynamics seen in equations 11:

$$
\begin{aligned}
\frac{d}{dt}\langle\sigma_x\rangle &= -2\left(u_y\langle\sigma_z\rangle - u_z\langle\sigma_y\rangle\right), \\
\frac{d}{dt}\langle\sigma_y\rangle &= -2\left(u_z\langle\sigma_x\rangle - u_x\langle\sigma_z\rangle\right), \\
\frac{d}{dt}\langle\sigma_z\rangle &= -2\left(u_x\langle\sigma_y\rangle - u_y\langle\sigma_x\rangle\right).
\end{aligned} \tag{11}
$$

The physics loss is defined as the MSE of the residuals in Equation 12:

$$
\mathcal{L}_{\text{physics}} = \frac{1}{N}\sum_{i=1}^{N}\left[\left(\frac{d}{dt}\langle\sigma_x\rangle - \Delta\langle\sigma_x\rangle\right)^2 + \left(\frac{d}{dt}\langle\sigma_y\rangle - \Delta\langle\sigma_y\rangle\right)^2 + \left(\frac{d}{dt}\langle\sigma_z\rangle - \Delta\langle\sigma_z\rangle\right)^2\right]. \tag{12}
$$

The total loss function (Equation 13) combines the data loss and the physics loss:

$$
\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{data}} + \lambda_{\text{physics}}\mathcal{L}_{\text{physics}}, \tag{13}
$$

where λ_{physics} controls the weight of the physics-informed loss.

The PINN architecture, depicted in Figure 6, effectively predicts both the control fields and the expectation values across a broad range of target states, balancing accuracy and adherence to physical laws. This ensures robust performance in simulating quantum state dynamics.

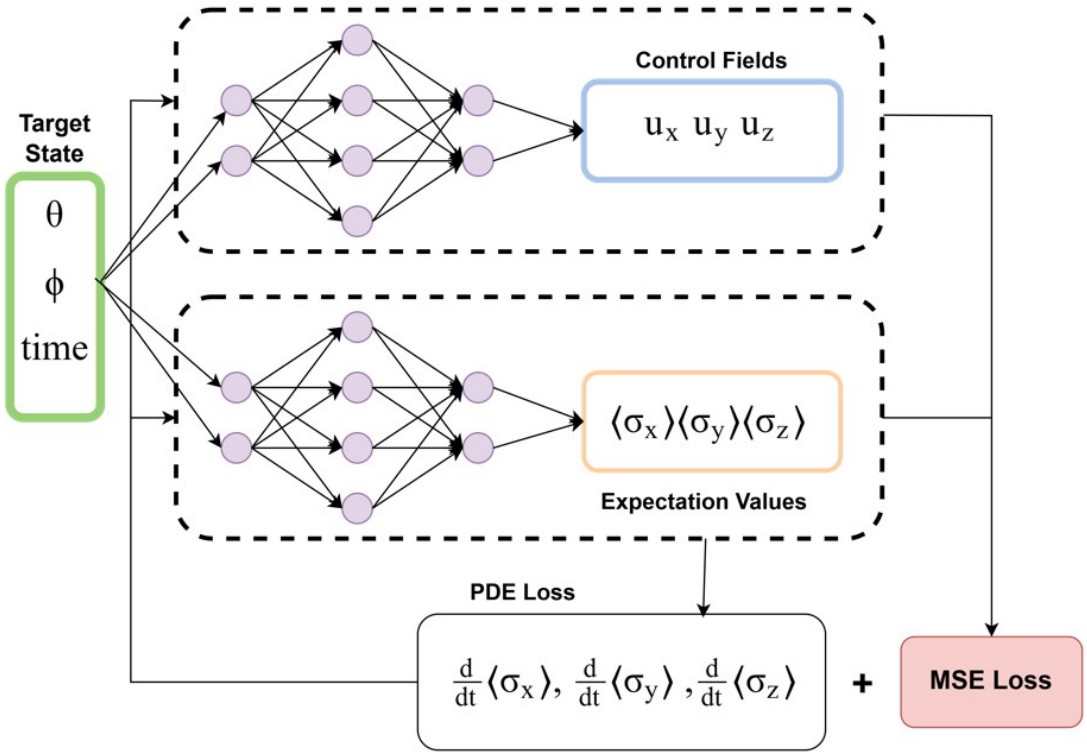

Figure 6: PINN Architecture

5 QubiNN: Qubit Neural Network Architecture

The QubiNN model is a fully connected neural network designed as a baseline to evaluate the effectiveness of the Physics-Informed Neural Network (PINN). Unlike the PINN, which incorporates physical laws into its training process, QubiNN relies entirely on data-driven learning. It serves as a purely numerical approach to predicting the dynamics of a qubit system, including the control fields and expectation values of the Pauli operators.

QubiNN is tasked with mapping the input parameters of the qubit system—(θ, ϕ, t), which specify the target state and the time evolution—to six output quantities: the control fields u_x, u_y, u_z and the expectation values $\langle\sigma_x\rangle, \langle\sigma_y\rangle, \langle\sigma_z\rangle$. The input to the network is a three-dimensional vector as seen in Equation 9 where θ and ϕ represent the target state parameters on the Bloch sphere, and t is the normalized time. The output is a six-dimensional vector (Equation 10) where the control fields dictate the external perturbations applied to the qubit, and the expectation values characterize the state evolution on the Bloch sphere.

The architecture consists of a feed forward fully connected structure with multiple layers. The input layer processes the three input features and propagates them through a sequence of hidden layers, each employing ReLU activation functions. These hidden layers allow the network to capture complex, non-linear relationships between the input and output spaces. The final output layer produces six numerical values corresponding to the control fields and expectation values.

This architecture, while simple in design, is highly flexible and capable of learning mappings directly from data without additional constraints, making it an ideal baseline for comparison with physics-informed methods.

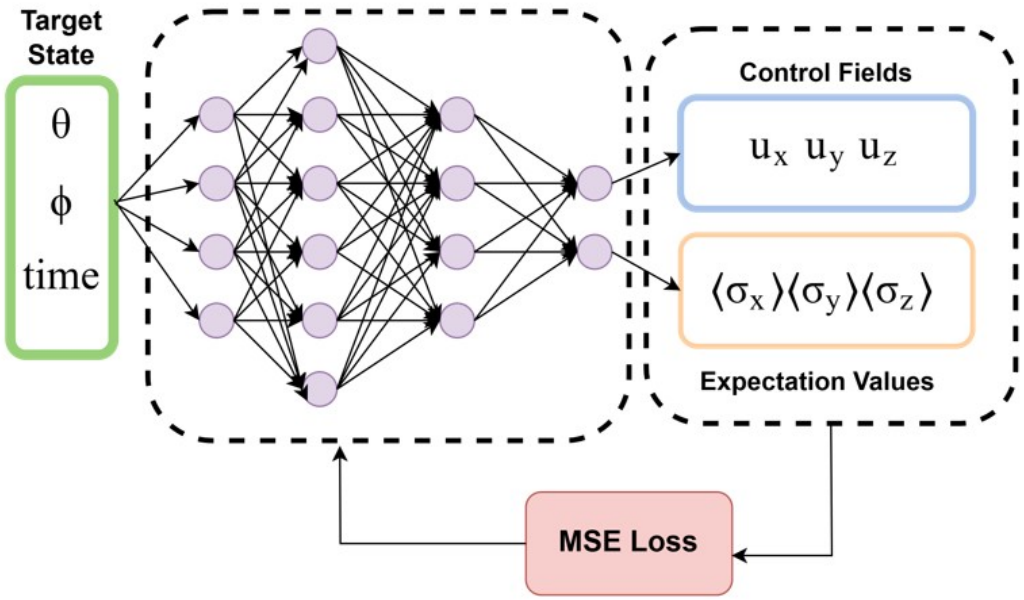

Figure 7: QubiNN Architecture

The QubiNN model was trained using supervised learning, with the training dataset

generated from QuTiP simulations of single-qubit dynamics. Each sample in the dataset consists of an input feature vector, $\mathbf{x}_i$, and the corresponding ground truth output vector, $\mathbf{y}_i^{\text{true}}$, derived from optimized GRAPE computations.

The objective of training is to minimize the error between the predicted output, $\mathbf{y}_i^{\text{pred}}$, and the true values, $\mathbf{y}_i^{\text{true}}$, Equation 14. This error is quantified using the mean squared error (MSE) loss function:

$$\mathcal{L}_{\text{data}} = \frac{1}{N} \sum_{i=1}^{N} \left\| \mathbf{y}_i^{\text{true}} - \mathbf{y}_i^{\text{pred}} \right\|_2^2, \tag{14}$$

where N is the total number of samples in the dataset. The MSE loss penalizes larger deviations more heavily, making it an effective measure for regression tasks involving continuous-valued outputs.

The model was optimized using the Adam optimizer, which dynamically adapts learning rates based on gradient magnitudes, enhancing the convergence rate. A learning rate of 0.001 was used, and the model was trained for multiple epochs to achieve stability in the loss function.

6 Results

This section presents the performance evaluation of the PINN and the baseline data-driven QubiNN against the QuTiP-generated values for three test target states: A, B, and C. The target states were selected to represent a diverse range of qubit orientations on the Bloch sphere, as shown in Figure 8. The benchmarks include comparisons of expectation values, control fields, fidelity, and computational efficiency.

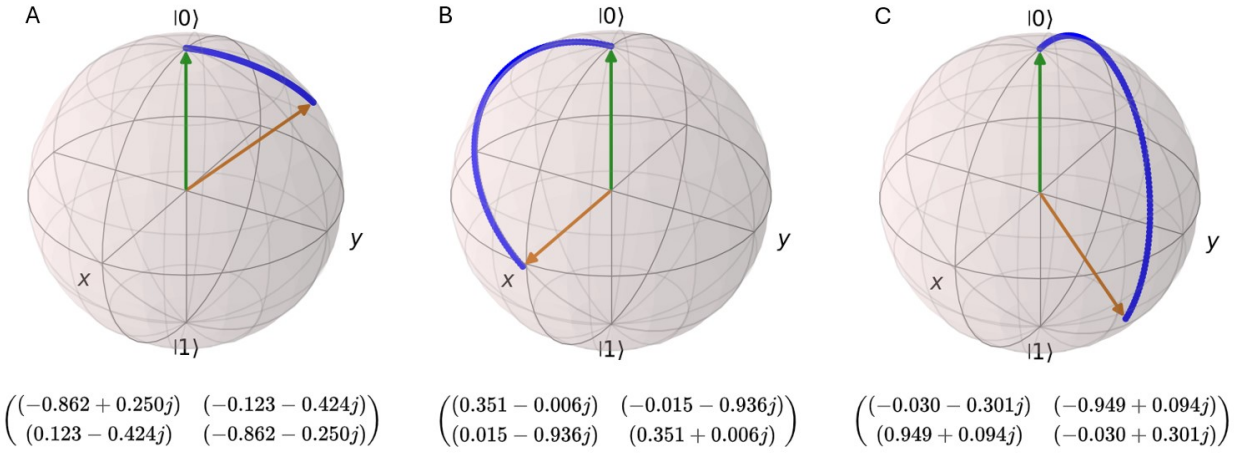

Figure 8: Tested states for benchmarking.

6.1 Visualizing Qubit Dynamics

Figure 8 illustrates the qubit's initial state ($\psi_0 = |0\rangle$), the target state vector, and the expected value trajectories on the Bloch sphere for each of the test states. The plots provide

an intuitive understanding of how the predicted and simulated trajectories evolve toward the target states. The PINN predictions show good alignment with the expected trajectories, capturing the overall dynamics effectively. QubiNN, while performing well in most cases, exhibits deviations in specific scenarios, particularly for state B, as discussed below.

6.2 Visual Comparison on the Bloch Sphere

To provide a visual comparison of the dynamics predicted by the QuTiP simulation, PINN, and QubiNN, a matrix of Bloch spheres is presented in Figure 9. Each row corresponds to a target state: the first row is state A, the second row is state B, and the third row is state C. Each column represents the method used: the first column shows the QuTiP simulation, the second column displays the PINN predictions, and the third column illustrates the QubiNN predictions.

Within each Bloch sphere, the initial state ($\psi_0 = |0\rangle$), the target state, and the predicted trajectory are depicted. This visualization highlights the ability of each model to approximate the quantum dynamics and approach the target state.

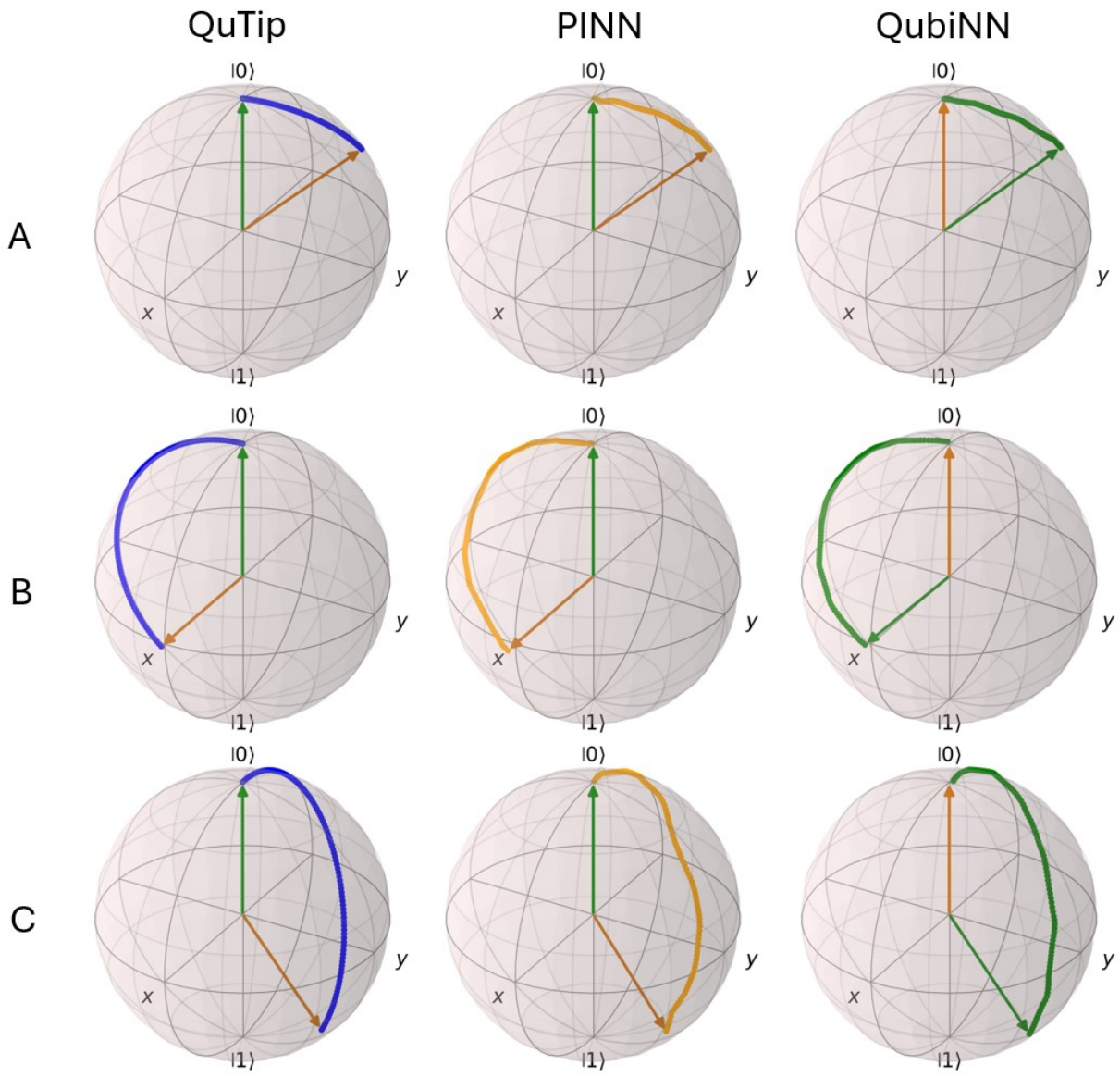

Figure 9: Comparison of state trajectories for target states A, B, and C on the Bloch sphere. Rows correspond to target states, while columns represent the simulation method (QuTiP, PINN, QubiNN). Each Bloch sphere displays the initial state ($\psi_0 = |0\rangle$), the target state, and the predicted trajectory.

The figure shows the comparison on the paths taken by each of the models against the actual calculated paths. Both the PINN and the QubiNN, struggle to maintain smooth path transitions but still achieve accurate state predictions.

6.3 Expectation Values and Control Fields

Figures 10 and 11 compare the predicted expectation values and control fields, respectively, for states A, B, and C. Each row corresponds to one target state, showing the dynamics of $\langle\sigma_x\rangle$, $\langle\sigma_y\rangle$, and $\langle\sigma_z\rangle$ or u_x, u_y, and u_z over time.

The QubiNN demonstrates consistently low MSE and high average fidelity in the expectation values across all three states, as shown in Tables 1 and 2. The PINN maintains a low MSE and high fidelity relative to the QubiNN, but performs better in the control field predictions.

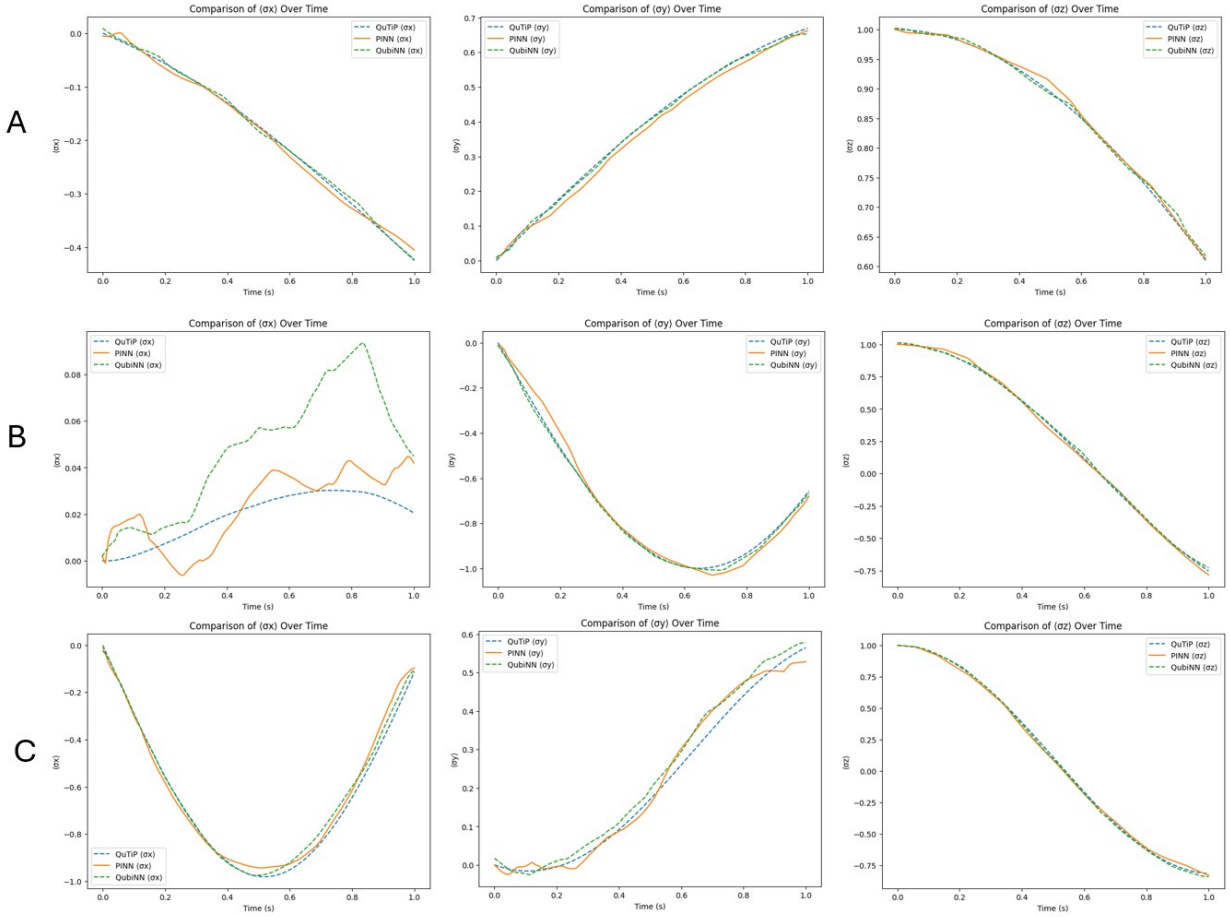

Figure 10: Comparison of expecation values over time for each of the tested states.

The PINN predictions for the control fields relatively match the QuTiP-generated values for states A and C, as reflected in the MSE values in Table 1. However, due to the sensitive nature of the control fields, these predictions do not reflect the optimized values. For state B, the PINN and QubiNN exhibit larger errors, with QubiNN showing significantly higher deviations in u_y and u_z.

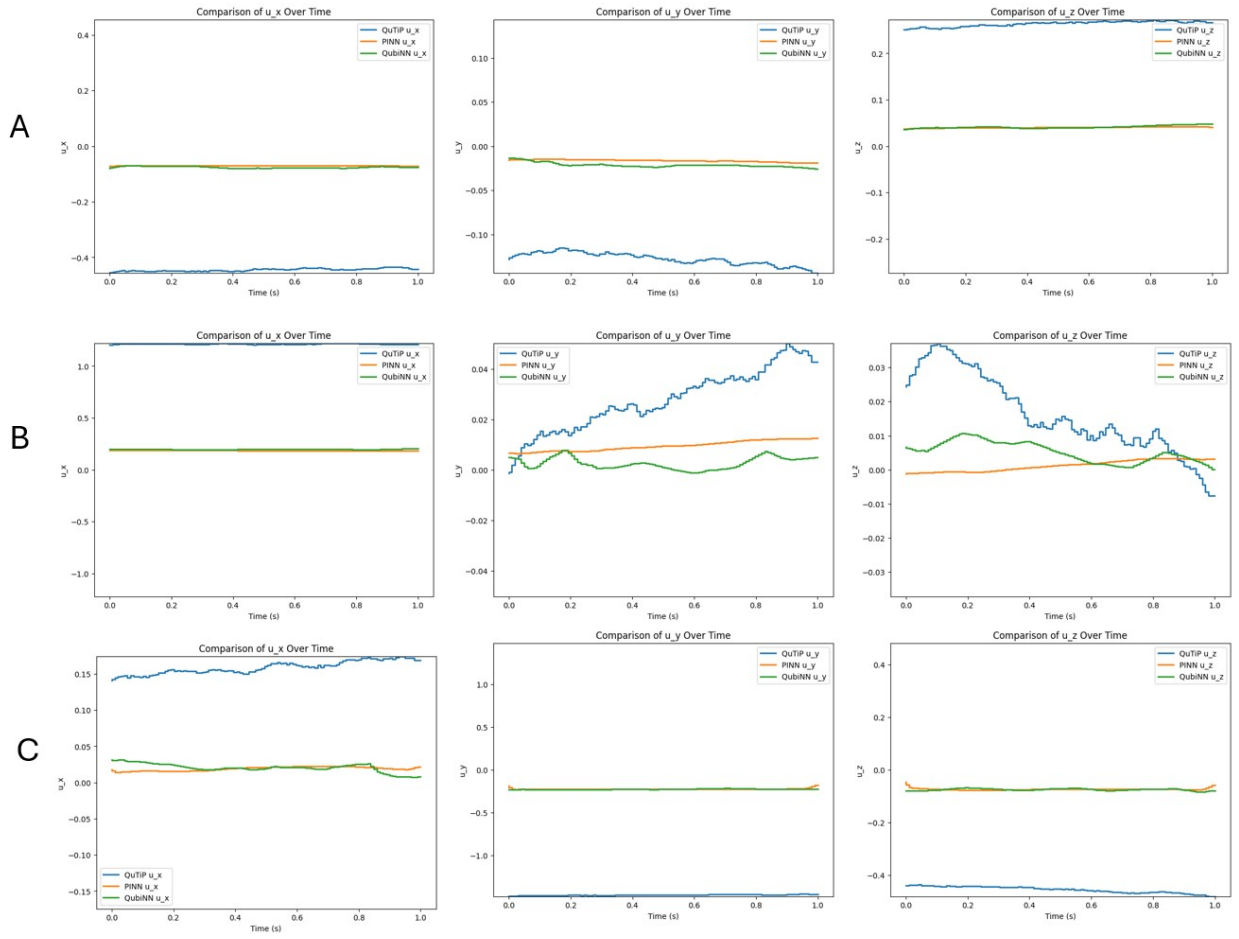

Figure 11: Comparison of control fields values over time for each of the tested states.

Table 1: Mean Square Error of all predicted values against QuTip simulated values.

State	model	$\langle\sigma_x\rangle$	$\langle\sigma_x\rangle$	$\langle\sigma_z\rangle$	u_x	u_y	u_z
A	PINN	0.000079	0.000300	0.000043	0.140113	**0.012371**	**0.049927**
	QubiNN	**0.000025**	**0.000036**	**0.000028**	**0.084466**	0.301498	0.376314
B	PINN	**0.000121**	0.001139	0.000356	**1.051941**	**0.000437**	**0.000399**
	QubiNN	0.001080	**0.000107**	**0.000129**	1.357086	0.655014	0.380808
C	PINN	0.001323	**0.000641**	0.000368	**0.019587**	**1.534023**	**0.145950**
	QubiNN	**0.000735**	0.000790	**0.000152**	0.693327	2.930499	0.707953

6.4 Fidelity Analysis

Table 2 provides the average fidelity of of the path taken by the PINN and QubiNN compared to the simulated expectation values. For states A and C, both models achieve high fidelity, demonstrating their effectiveness in approximating the target quantum state, with QubiNN having an slight edge over the PINN. However, for state B, the fidelity values of the control fields show that both models struggled to predict the control fields for this particular target state.

Table 2: Fidelity of all predicted values against QuTip simulated values.

State	model	$\langle\sigma_x\rangle$	$\langle\sigma_x\rangle$	$\langle\sigma_z\rangle$	u_x	u_y	u_z
A	PINN	0.998492	0.999415	0.999959	**0.999791**	**0.998892**	**0.999855**
	QubiNN	**0.999585**	**0.999871**	**0.999964**	0.997990	0.987809	0.996269
B	PINN	0.865281	0.998151	0.999245	**0.999863**	**0.947039**	0.006356
	QubiNN	**0.976205**	**0.999908**	**0.999678**	0.999815	0.475345	**0.858737**
C	PINN	0.997771	0.993886	0.999440	**0.988137**	0.998877	0.995969
	QubiNN	**0.999045**	**0.997671**	**0.999701**	0.904886	**0.999628**	**0.997634**

Table 3 shows the computed fidelity of target state achieved by each model. With the QubiNN model having a slight better accuracy than the PINN model which only achieved a better fidelity for C state.

Table 3: Fidelity of the target state against the simulated target state.

State	PINN	QubiNN
A	0.997149	**0.998331**
B	1.009622	**0.996837**
C	**0.997326**	1.006842

6.5 Computational Efficiency

The computational efficiency of the PINN and QubiNN compared to the QuTiP simulation is summarized in Table 4. All simulations and predictions in this study were conducted using a single CPU instance provided by Google Colab notebooks. This setup ensures a consistent computational environment for evaluating both the QuTiP simulations and the neural network predictions. Both neural networks exhibit significant speedup, with the PINN achieving up to 4990x faster computation for state A and comparable performance for the

other states. QubiNN demonstrates slightly higher speedup due to its simpler architecture but at the cost of reduced fidelity and accuracy in some control field cases.

Table 4: Speedup

State	Time	QuTip	PINN	QubiNN
A	Total Time (s)	10.351861	0.002074	**0.001971**
	Speedup		4990.10x	**5252.07x**
B	Total Time	11.900153	0.003630	**0.002495**
	Speedup		3278.13x	**4768.14x**
C	Total Time	11.918799	0.003930	**0.002484**
	Speedup		3032.15x	**4796.69x**

7 Conclusions

This study evaluated the performance of Physics-Informed Neural Networks (PINNs) and data-driven neural networks (QubiNNs) in solving single-qubit quantum control problems. Both models were benchmarked against QuTiP simulations across three diverse target states, with a focus on accuracy in predicting expectation values and control fields, as well as computational efficiency.

The results indicate that the QubiNN consistently performed slightly better than the PINN in predicting expectation values, achieving lower mean squared error (MSE) and higher fidelity in most cases. This highlights the strength of QubiNN in learning data-driven mappings when the training data adequately represents the system's dynamics. In contrast, the PINN provided improvements in predicting control fields, but both models struggled overall, reflecting the inherent complexity of accurately approximating optimized control pulses.

From a computational perspective, the QubiNN achieved superior speedup compared to the PINN, making it a highly efficient solution for real-time quantum control simulations. However, its reliance solely on data-driven learning limits its generalization to unseen or physically complex scenarios. While the PINN incorporates physics-informed constraints through the Schrödinger equation, its added complexity did not translate into significant improvements in this study, particularly for control field predictions.

These findings emphasize the trade-offs between the two approaches. QubiNN demonstrates high accuracy and computational efficiency in data-rich environments but lacks the physical grounding to handle out-of-distribution dynamics. The PINN offers a framework for integrating physical laws, yet further refinement is required to fully exploit its potential, particularly for accurately modeling control fields.

Future work should explore hybrid models that combine the strengths of data-driven and physics-informed learning. Additionally, expanding this framework to multi-qubit systems or improving the representation of control fields could provide further insights into the applicability of neural networks in quantum control.

References

[1] Terrence J Sejnowski. "The unreasonable effectiveness of deep learning in artificial intelligence". In: *Proceedings of the National Academy of Sciences* 117.48 (2020), pp. 30033–30038.

[2] Francesco Regazzoni, Luca Dede, and Alfio Quarteroni. "Machine learning for fast and reliable solution of time-dependent differential equations". In: *Journal of Computational physics* 397 (2019), p. 108852.

[3] Lu Lu et al. "DeepXDE: A deep learning library for solving differential equations". In: *SIAM review* 63.1 (2021), pp. 208–228.

[4] Jiequn Han, Arnulf Jentzen, and Weinan E. "Solving high-dimensional partial differential equations using deep learning". In: *Proceedings of the National Academy of Sciences* 115.34 (2018), pp. 8505–8510.

[5] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations". In: *Journal of Computational physics* 378 (2019), pp. 686–707.

[6] Zhiping Mao, Ameya D Jagtap, and George Em Karniadakis. "Physics-informed neural networks for high-speed flows". In: *Computer Methods in Applied Mechanics and Engineering* 360 (2020), p. 112789.

[7] Shengze Cai et al. "Physics-informed neural networks for heat transfer problems". In: *Journal of Heat Transfer* 143.6 (2021), p. 060801.

[8] Corey Trahan, Mark Loveland, and Samuel Dent. "Quantum Physics-Informed Neural Networks". In: *Entropy* 26.8 (2024), p. 649.

[9] George Em Karniadakis et al. "Physics-informed machine learning". In: *Nature Reviews Physics* 3.6 (2021), pp. 422–440.

[10] Stefano Markidis. "On physics-informed neural networks for quantum computers". In: *Frontiers in Applied Mathematics and Statistics* 8 (2022), p. 1036711.

[11] Henry Jin, Marios Mattheakis, and Pavlos Protopapas. "Physics-informed neural networks for quantum eigenvalue problems". In: *2022 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2022, pp. 1–8.

[12] Priya Batra and TS Mahesh. "Physics-informed neural network for quantum control of NMR registers". In: *arXiv preprint arXiv:2407.00444* (2024).

[13] Bjarni Jónsson, Bela Bauer, and Giuseppe Carleo. "Neural-network states for the classical simulation of quantum computing". In: *arXiv preprint arXiv:1808.05232* (2018).

[14] Paul Nation. *QuTip: Quantum Toolbox in Python*. https://qutip.org/. 2010–2024.

[15] Felix Bloch. "Nuclear induction". In: *Physical review* 70.7-8 (1946), p. 460.