Solving Time-Dependent Schrodinger Equation using Physics Informed Neural Networks

Deepak Bhatter
B22EE022
b22ee022@iitj.ac.in

Prajjwal Dixit
B22ES002
b22es002@iitj.ac.in

Rahul Sharma
B22EE051
b22ee051@iitj.ac.in

https://github.com/deepak5512/PINNs

Abstract

In this project, we employ **Physics-Informed Neural Networks (PINNs)** to solve **Partial Differential Equations (PDEs)**, beginning with a standard PDE to validate the methodology. After establishing the effectiveness of PINNs on simpler equations, we extend the approach to solve the **Time-Dependent Schrödinger Equation (TDSE)**, a core equation in quantum mechanics. By embedding the physical laws directly into the training process, PINNs offer a mesh-free, data-efficient alternative to traditional numerical solvers. Our results demonstrate the capability of PINNs to model complex physical systems accurately.

1 Introduction

Partial Differential Equations (PDEs) play a central role in modeling complex physical systems across various domains such as fluid mechanics, electromagnetics, and quantum physics. However, solving these equations—especially in high dimensions or with complex boundary conditions—remains computationally intensive and challenging. This is particularly true for equations like the Schrödinger Equation, which is foundational to quantum mechanics and describes how quantum states evolve over time. Recent advances in machine learning, particularly deep learning, have opened up new possibilities for data-driven approaches in scientific computing. One such approach is Physics-Informed Neural Networks (PINNs), which fuse neural networks with the laws of physics to approximate solutions of PDEs in a more flexible and efficient manner.

1.1 Overview of PDEs in Physics

PDEs describe how physical quantities vary with respect to multiple variables such as space and time. Solving a PDE typically involves satisfying both the equation itself and relevant boundary and initial conditions. Classical methods—like finite difference or finite element methods—require discretization of the domain, often leading to high computational costs and limitations in scalability.

1.2 Need for Data Driven Approaches

Many physical systems are either too complex for analytical solutions or lack sufficient data for traditional numerical modeling. Moreover, real-world data often comes with noise or is incomplete. In such scenarios, data-driven models guided by physical laws offer a powerful alternative. PINNs, in particular, stand out because they do not require large labeled datasets and inherently ensure the physical validity of the learned solution.

Submitted as a project report for the course **Deep Learning (CSL7590)** at IIT Jodhpur. Do not distribute.

1.3 Introduction to Physics-Informed Neural Networks (PINNs)

PINNs are neural networks that incorporate physical constraints—represented by differential equations—into the training process. They work by minimizing a loss function composed of two parts: one that penalizes deviations from known data (data loss), and another that penalizes violations of the governing PDE (physics loss). By using automatic differentiation, PINNs efficiently compute the PDE residuals, enabling accurate and mesh-free solutions to differential equations.

1.4 Objectives of the Project

The primary goals of this project are as follows:

- To understand and implement PINNs by first solving a standard, well-understood PDE.
- To extend the PINN framework to solve the Time-Dependent Schrödinger Equation (TDSE).
- Evaluate the performance of PINNs in capturing quantum dynamics, comparing them against known analytical or numerical benchmarks.
- To analyze the effect of neural architecture, loss design, and optimization strategies on the quality of PDE solutions.

2 Literature Review

The application of neural networks to solve partial differential equations (PDEs) has become extremely popular in recent years, particularly following the introduction of Physics-Informed Neural Networks (PINNs). First introduced by Raissi et al. (2019), PINNs leverage the physics of the problem of interest—captured in differential equations—to guide the learning process of neural networks. Compared to traditional data-driven approaches, PINNs embed the governing equations as soft constraints in the loss function, thus increasing their ability to address problems with limited or partially available labeled data.

In quantum mechanics, the Schrödinger Equation is used to describe the quantum state evolution. Classical numerical techniques, including finite difference time domain (FDTD) methods, Crank-Nicolson schemes, and spectral techniques, have been employed to solve the Time-Dependent Schrödinger Equation (TDSE). These techniques, while being generally valid, can be very computationally intensive and can be plagued by stability and discretization error issues, particularly for multidimensional systems.

Recent research has shown that PINNs are capable of efficiently solving the Time-Independent and Time-Dependent Schrödinger Equations. For example, Zhang et al. (2020) used PINNs to calculate the energy eigenvalues and wavefunctions of 1D quantum wells. The model showed comparable accuracy to analytical solutions. Likewise, Lu et al. (2021) applied this framework to multi-dimensional quantum systems and also investigated adaptive loss weighting strategies to speed up convergence. In particular, the use of PINNs in TDSE opens the door to modeling complex quantum systems where analytic solutions are infeasible. By embedding the TDSE into the network's loss function, the model learns to approximate both the real and imaginary components of the wave function over space and time. This approach has been shown to offer smooth, continuous approximations that preserve important quantum properties such as probability conservation.

Additionally, studies have examined the integration of boundary and initial condition data (data loss) as well as physics loss to improve accuracy. Supervised learning-based hybrid models with learned data points and physics-informed constraints have been shown to perform well at stabilizing training processes as well as reducing errors.

In conclusion, the intersection of deep learning and physics with PINNs offers an exciting new approach to solving quantum mechanical equations. The literature is full of overwhelming evidence that supports the viability of applying PINNs to model TDSE-governed wave functions, so it is justified to use them in this project.

3 Problem Statement

3.1 Mathematical Formulation of the Schrödinger Equation

The Schrödinger Equation is one of the most fundamental equations in quantum mechanics, governing the behavior of quantum systems. It describes how the quantum state of a physical system changes over time and space.

3.1.1 Time-Independent Schrödinger Equation (TISE)

The Time-Independent Schrödinger Equation is used when the system's potential does not vary with time. It is given by:

$$\hat{H}\psi(x) = E\psi(x) \tag{1}$$

where:

- $\hat{H}$ is the Hamiltonian operator, usually:

$$\hat{H} = -\frac{\hbar^2}{2m}\frac{d^2}{dx^2} + V(x) \tag{2}$$

- $\psi(x)$ is the wave function,
- E is the energy eigenvalue.

The TISE is typically used for finding stationary states and quantized energy levels.

3.1.2 Time-Dependent Schrödinger Equation (TDSE)

In many quantum systems, the state evolves over time. The Time-Dependent Schrödinger Equation is written as:

$$i\hbar\frac{\partial\psi(x,t)}{\partial t} = \hat{H}\psi(x,t) \tag{3}$$

In one spatial dimension, this becomes:

$$i\hbar\frac{\partial\psi(x,t)}{\partial t} = \left(-\frac{\hbar^2}{2m}\frac{\partial^2}{\partial x^2} + V(x)\right)\psi(x,t) \tag{4}$$

where:

- $\psi(x,t)$ is the complex-valued wave function,
- $\hbar$ is the reduced Planck's constant,
- $V(x)$ is the potential energy.

This form governs the full dynamics of a quantum system over time and space.

3.1.3 Nonlinear Schrödinger Equation

In certain physical systems (e.g., nonlinear optics or Bose-Einstein condensates), a nonlinear version of the Schrödinger Equation appears. It generally includes a term like $|\psi|^2\psi$, leading to:

$$i\frac{\partial\psi}{\partial t} + \frac{1}{2}\frac{\partial^2\psi}{\partial x^2} + |\psi|^2\psi = 0 \tag{5}$$

This project primarily focuses on the linear TDSE; however, extending to nonlinear cases is a possible future direction.

3.2 Boundary and Initial Conditions

To obtain a well-posed solution to the TDSE, suitable boundary and initial conditions must be defined:

Initial Condition

The wave function at time $t = 0$ is usually specified as:

$$\psi(x, 0) = \psi_0(x) \tag{6}$$

Boundary Conditions

Depending on the problem, common boundary conditions include:

- **Dirichlet:** $\psi(a, t) = \psi(b, t) = 0$ (e.g., infinite potential well),
- **Periodic:** $\psi(a, t) = \psi(b, t)$,
- **Neumann:** $\frac{\partial \psi}{\partial x}(a, t) = \frac{\partial \psi}{\partial x}(b, t) = 0$

The selection of these conditions significantly affects the behavior and solvability of the TDSE. In the current project, we are using Dirichlet conditions as Boundary Conditions.

3.3 Goal of the Project

The main goals of this project are:

- To solve the Time-Dependent Schrödinger Equation using Physics-Informed Neural Networks (PINNs).
- To first validate the approach on a simpler PDE to understand the PINN framework.
- To design and train a neural network that can approximate the solution $\psi(x, t)$ over a specified domain.
- To incorporate the physics of the TDSE into the training via the PDE residual.
- To assess the effectiveness of PINNs by comparing results with analytical solutions (where available) or traditional numerical solvers.

4 Proposed Method

4.1 Why are PINNs required?

PINNs leverage neural networks to approximate solutions to PDEs while incorporating physical laws as constraints. Instead of relying purely on data to learn relationships, PINNs leverage the structure of known physics to guide their solutions. This hybrid approach balances data and physics, ensuring robustness and accuracy, even with sparse or noisy data.

4.2 Comparison: PINNs vs. Data-Only Neural Networks

To empirically evaluate the advantage of incorporating physics into the learning process, we compared the performance of a Physics-Informed Neural Network (PINN) with a traditional neural network trained solely on data points generated from the analytical solution of the heat equation.

4.2.1 Experimental Setup

- **PDE Used:** 1D Heat Equation
- **Analytical Solution:** Known and used to generate supervised data
- **Data:** A subset of spatial-temporal points with corresponding solution values
- **Loss Metric:** PDE residual loss (Physics loss) + Data Loss
- **Evaluation:** Visual comparison with the ground truth and PDE residual magnitude

4.2.2 Results

1. PINNs

- Trained using both data points and the PDE residual enforced through collocation points
- **PDE Loss:** $\approx 10^{-3}$
- **Observation:** The predicted solution closely matches the true analytical solution, even at regions between known data points (better generalization)

2. Data-Only Neural Network

- Trained solely on the data using standard MSE loss
- **PDE Loss:** ≈ 0.51
- **Observation:** The predicted solution fits the training data but diverges in regions with no data, showing overfitting and lack of physical consistency

4.2.3 Visual Comparison

The following graphs depict the predictions of both models compared to the ground truth.

- **PINN Solution vs True Solution**

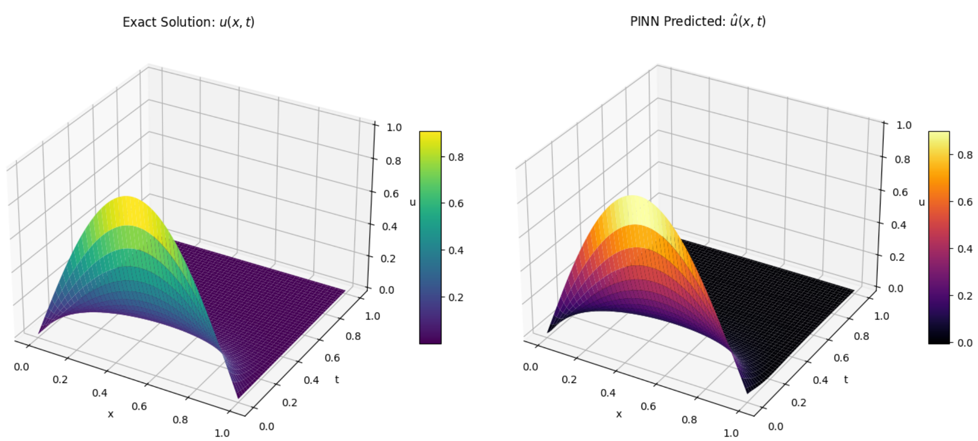

Figure 1: Exact Solution vs PINNs Solution

- **PDE Loss Comparison**
  - PDE Loss for PINNs = $1.80 * 10^{-3}$
  - PDE Loss for Data-Only Neural Network = $5.17 * 10^{-1}$

4.2.4 Discussion

From both the visualizations and the PDE loss metrics, it is evident that PINNs outperform traditional neural networks trained only on data. The core reason is that PINNs leverage the underlying physics (PDE) as an additional form of supervision, enabling them to learn solutions that are physically consistent across the domain—even in areas with sparse or no data. In contrast, a standard neural network lacks any notion of physical laws and hence performs poorly in extrapolating or interpolating beyond the training data, especially in physics-governed systems.

4.3 PDE Residual

A key concept in PINNs is the PDE residual, which measures how well a candidate solution satisfies the governing equations. **Example:** For the heat equation:

$$\frac{\partial u}{\partial t} - \alpha \frac{\partial^2 u}{\partial x^2} = 0 \tag{7}$$

the residual is defined as:

$$f(t, x; \theta) = \frac{\partial u}{\partial t} - \alpha \frac{\partial^2 u}{\partial x^2} \tag{8}$$

Here, $u(t, x; \theta)$ is the neural network approximation with parameters θ. Ideally, $f(t, x; \theta) = 0$, indicating that the approximation satisfies the PDE. During training, the network minimizes the residual across the domain, progressively refining its approximation.

4.4 Loss Function Design

PINNs use a composite loss function that combines data consistency and physics adherence.

1. Data Loss

Penalizes errors between network predictions and observed data:

$$\text{MSE}_u = \frac{1}{N_u} \sum_{i=1}^{N_u} |u(t_i, x_i) - u_i|^2 \tag{9}$$

where (t_i, x_i, u_i) are data points.

2. Physics Loss

Penalizes deviations from the PDE residual at collocation points:

$$\text{MSE}_f = \frac{1}{N_f} \sum_{i=1}^{N_f} |f(t_i, x_i; \theta)|^2 \tag{10}$$

The total loss:

$$\text{Loss} = \text{MSE}_u + \text{MSE}_f \tag{11}$$

ensures that the solution adheres to both the data and the physics.

4.5 Requirement of Data in PINNs

One of the key advantages of Physics-Informed Neural Networks (PINNs) is their ability to learn without relying on traditional labeled datasets. When the governing physical laws—typically in the form of partial or ordinary differential equations—and initial/boundary conditions are well-defined, PINNs can be trained solely by minimizing the residuals of these equations at sampled points in the domain. In such cases, no real-world data is required. However, in practical applications where:

- The system is partially observed
- The governing equations are incomplete or unknown
- There is experimental data or noisy observations

PINNs can incorporate this available data into the training process by extending the loss function to include a data mismatch term. This hybrid approach ensures that the network respects both the known physics and the empirical evidence.

Thus, while data is not a strict requirement for training a PINN, it can significantly enhance performance and applicability in real-world scenarios.

4.6 Collocation Points

Collocation points are randomly sampled points in the problem domain where the PDE residual is evaluated. They ensure that the neural network respects the physics globally, not just at observed data points.
For example, in a rod with length L = 10 and time interval T = 1 , collocation points might be randomly sampled in [0, 1] x [0, 10] .

4.7 Training PINNs

Training involves minimizing the total loss with respect to the network parameters θ. This process ensures that the solution satisfies both the data and the physics.

PINNs rely on automatic differentiation (AD) to compute derivatives from the neural network. **Automatic Differentiation** computes derivatives using computational graphs, ensuring high precision, avoiding errors associated with numerical differentiation.

4.8 PINNs for Time-Dependent Schrodinger Equation

This section presents the implementation of a Physics-Informed Neural Network (PINN) framework to solve the 1D time-dependent Schrödinger equation (TDSE). The equation governs the evolution of a quantum wavefunction $\psi(x, t)$ over time and is given in natural units ($\hbar = m = 1$) as:

$$i\frac{\partial \psi}{\partial t} = -\frac{1}{2}\frac{\partial^2 \psi}{\partial x^2}. \tag{12}$$

The implementation of the PINN for this PDE consists of the following components:

4.8.1 Complex-Valued Representation

Since ψ is complex-valued, the neural network outputs two channels corresponding to Re(ψ) and Im(ψ). These are trained jointly using a composite loss.

Residual Loss Computation (Physics-Informed)

Automatic differentiation is used to calculate the derivatives required to compute the PDE residual. The real and imaginary parts of the residual are formulated separately to enforce compliance with the TDSE across the domain.

4.8.2 Boundary and Initial Condition Enforcement

The loss function incorporates both initial and boundary condition penalties using a known analytical solution (plane wave). These are treated as supervised training points, while the interior domain uses unsupervised collocation points with PDE loss enforcement.

4.8.3 Loss Function Structure

The total loss is a weighted combination of:

- Initial condition loss: L_{init}
- Boundary condition loss: L_{bdry}
- Physics (residual) loss: L_{pde}

This structure enables the network to learn physically consistent solutions even in regions where direct data is unavailable.

4.8.4 Training Strategy

The model is trained using the Adam optimizer and a scheduled learning rate decay over 5000 epochs. Loss metrics are logged periodically for convergence analysis.

4.8.5 Model Architecture

- **Input Layer:**
  - 2 neurons: x (space), t (time)
- **Input Normalization Layer:**
  - A Lambda layer scales both inputs from $[x_{\min}, x_{\max}]$, $[t_{\min}, t_{\max}]$ to the range $[-1, 1]$
- **Hidden Layers:**

- **4 hidden layers** (customizable via `num_layers`)
- Each layer has **20 neurons** (customizable via `neurons_per_layer`)
- Activation function: `Tanh`
- Weight initializer: **Glorot normal (Xavier initialization)**

- **Output Layer:**
  - 2 neurons:
    * Output 1: $\text{Re}(\psi)$ (real part of wave function)
    * Output 2: $\text{Im}(\psi)$ (imaginary part of wave function)
  - No activation function (i.e., linear)

5 Experiments & Results

5.1 Experimental Setup

The experiments are designed to test the PINN's ability to learn the exact quantum dynamics of a free particle under a known analytical solution:

- **Domain:**
  - Space: $x \in [0, 1]$
  - Time: $t \in [0, \pi]$
- **Exact Solution Used:**

$$\psi(x,t) = \cos(kx - \omega t) + i \sin(kx - \omega t), \quad \text{with } k = 1,\, \omega = \frac{1}{2}$$

- **Data Used for Training:**
  - 50 initial condition points
  - 50 boundary condition points
  - 10,000 collocation points inside the domain

5.2 Quantitative Results

Loss Convergence

Over 5000 epochs, the total loss and its individual components (initial, boundary, physics) decrease significantly.

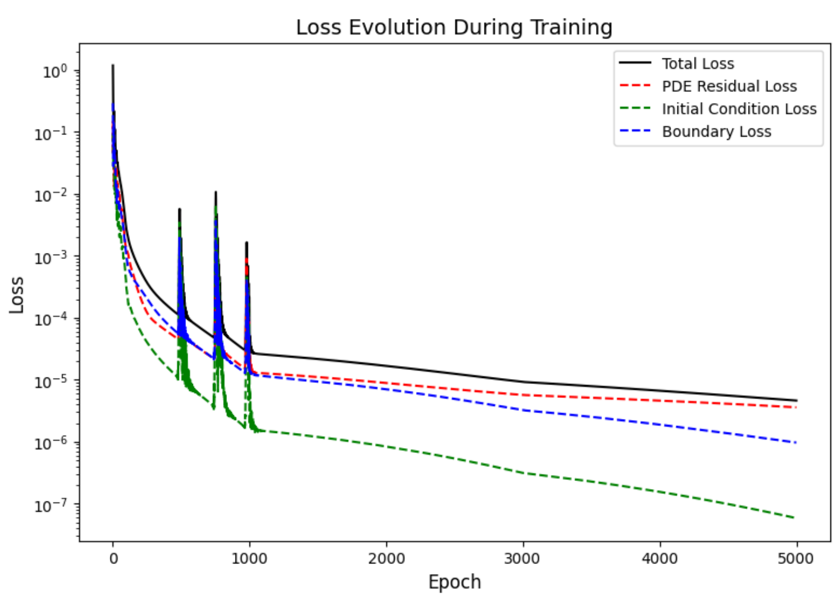

Figure 2: Semilog plot of total loss, PDE residual loss, initial loss, and boundary loss vs. epochs.

Error Norms

On randomly sampled test points, L_1 and L_2 errors are computed periodically.

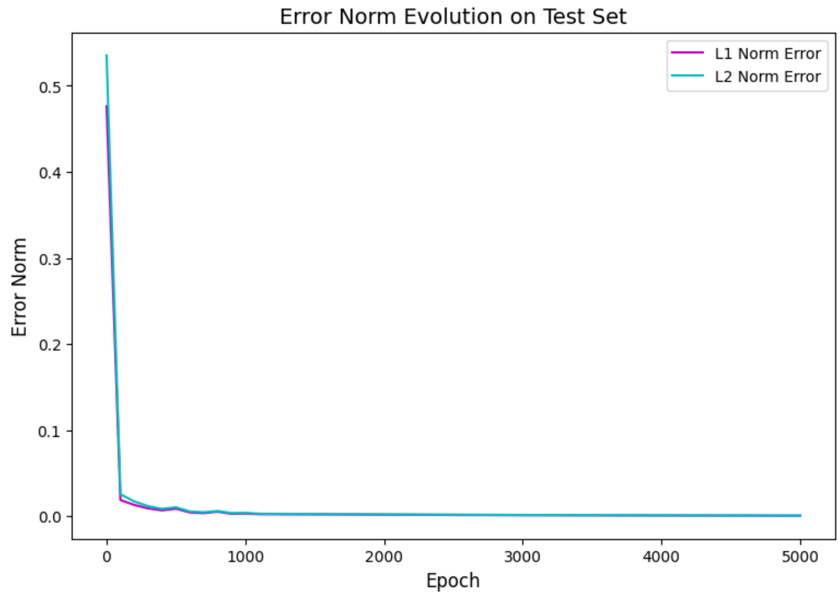

Figure 3: L_1 and L_2 norm errors over training epochs, showing convergence to near-zero values.

PDE Residual Heatmaps

Residual values of the real and imaginary parts of the TDSE across the domain are visualized.

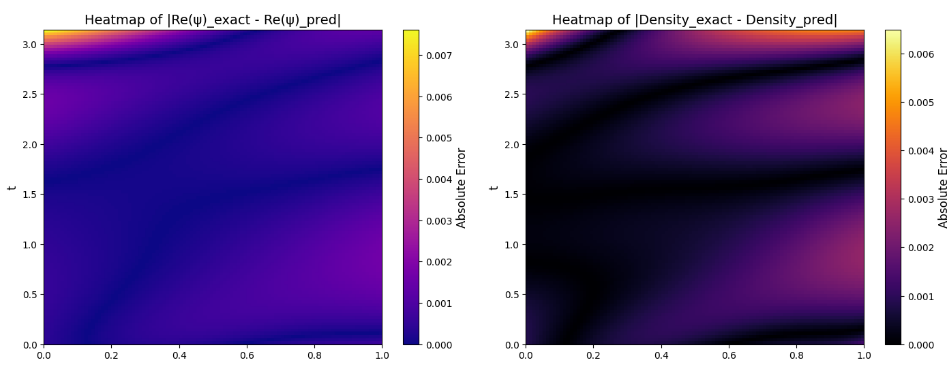

Figure 4: 2D heatmaps of absolute error in Re(ψ) and $|\psi|^2$ over the space-time domain.

5.3 Qualitative Results

Initial and Boundary Conditions

The model's output is evaluated at $t = 0$ and compared to the analytical initial condition. Similarly, predictions at $x = 0$ and $x = 1$ over time are plotted and compared to the expected boundary behavior.

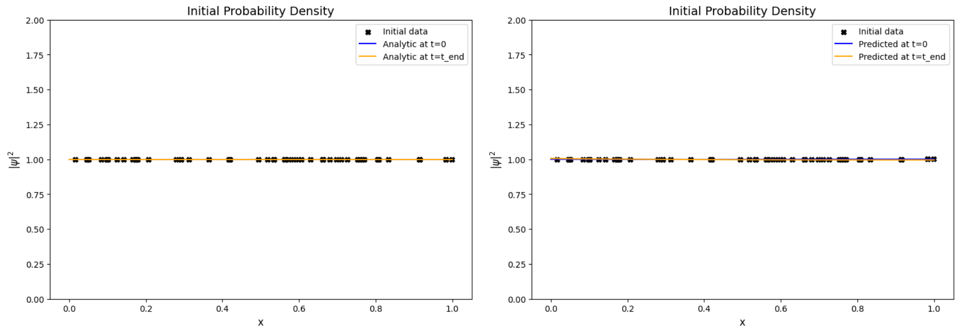

Figure 5: Density plots at $t = 0$ using the exact equation vs trained model

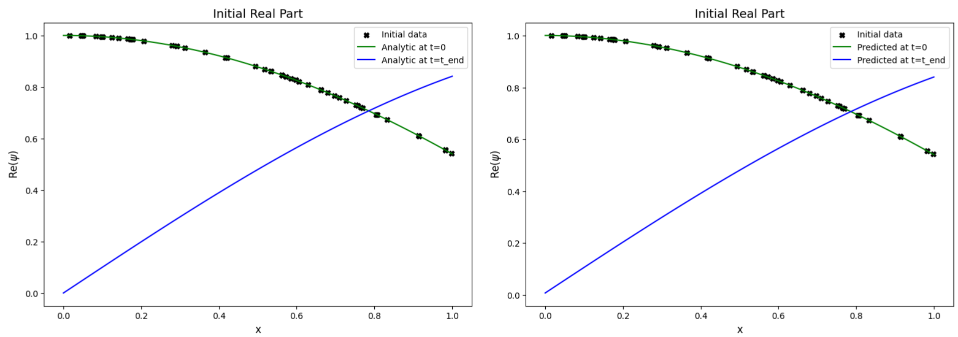

Figure 6: Real plots at $t = 0$ using the exact equation vs trained model

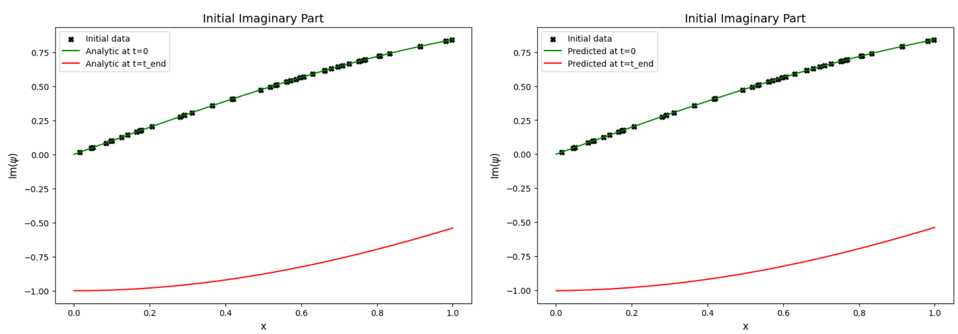

Figure 7: Imaginary plots at $t = 0$ using the exact equation vs trained model

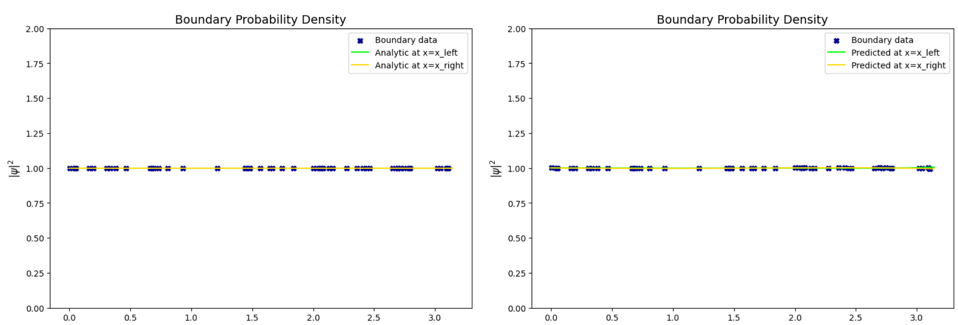

Figure 8: Boundary value predictions for $|\psi|^2$ vs. time at $x = 0$ and $x = 1$.

3D Visualization of the Wavefunction

A surface plot of $\mathrm{Re}(\psi)$ over (x, t) showcases the smooth temporal-spatial evolution learned by the network.

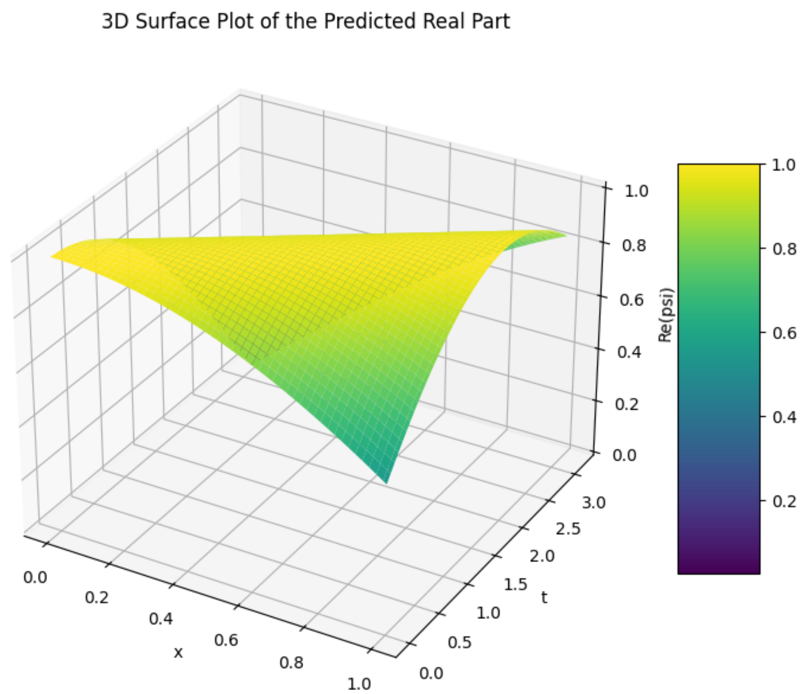

Figure 9: 3D surface plot of predicted $\mathrm{Re}(\psi(x, t))$ from the trained model.

5.4 Physical Property Evaluation

Probability Conservation

The integral $\int |\psi(x, t)|^2 dx$ is computed at various time snapshots to verify normalization.

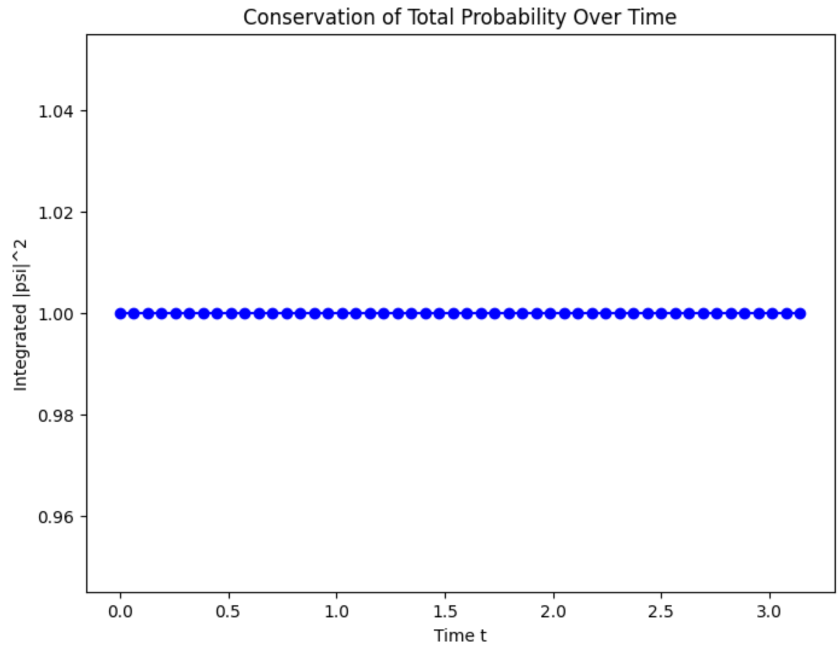

Figure 10: Plot of total probability vs. time showing near-perfect conservation.

Kinetic Energy Profile

Using the learned gradient of ψ, the kinetic energy density is approximated and plotted for a fixed time slice.

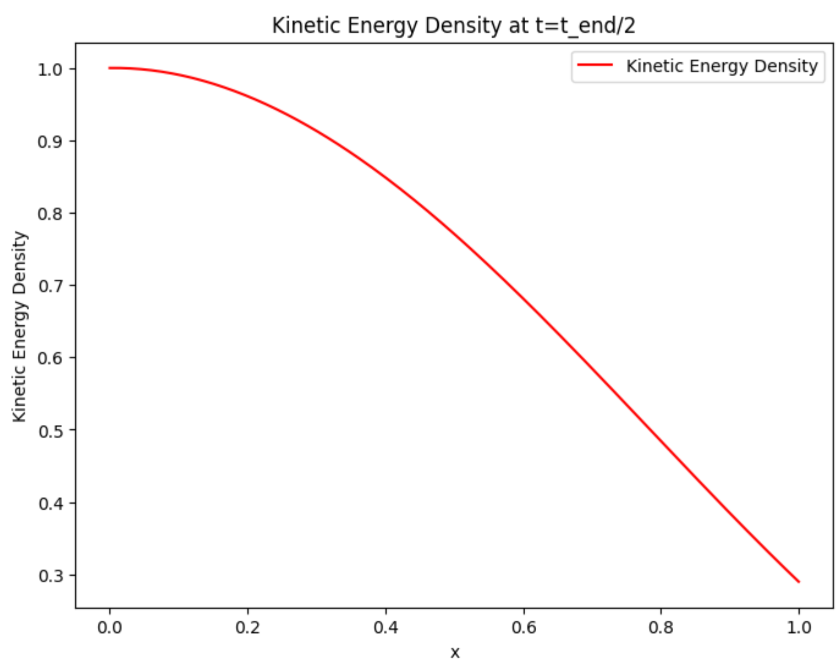

Figure 11: Spatial plot of kinetic energy density at $t = \pi/2$, validating physical consistency.

6 Conclusions

- The PINN successfully learns the complex-valued dynamics of the Schrödinger equation using minimal supervision.
- Predictions are highly accurate across the entire domain, matching well with the analytical solution.
- The model respects conservation principles and captures key quantum behaviors, making it a promising tool for solving more advanced quantum PDEs without relying on traditional meshing or discretization.

References

1 Medium Article (Physics-Informed Neural Networks (PINNs))

2 Medium Article (Understanding Physics-Informed Neural Networks (PINNs) — Part 1)

3 YouTube Video

4 Raissi, Maziar, 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."

5 Zhang, D., Guo, L., & Karniadakis, G. E. (2020). Learning in the presence of unknown unknowns: Robust physics-informed neural networks. Journal of Computational Physics, 405, 109109.

6 Lu, L., Meng, X., Mao, Z., & Karniadakis, G. E. (2021). DeepXDE: A deep learning library for solving differential equations. SIAM Review, 63(1), 208–228.