# Physics-Informed Neural Networks to Solve the Many Electron Time-Dependent Schrödinger Equation

Presented by,
Afthash Sahal Ubaid Puzhakkal
a1913863

# Introduction

- **The Schrödinger equation:** Fundamental equation in quantum mechanics that describes how the wavefunction, $\Psi$, evolves over time
- **Traditional Methods:** Analytical methods for simple systems or numerical techniques like Crank-Nicholson method for complex systems
- **Project Goal**: Solve the **Time-Dependent Schrödinger equation (TDSE)** using **Physics-Informed Neural Networks (PINNs)**
- **PINN:** They are a type of neural networks that learns to solve **Partial Differential Equations (PDEs)** by embedding the relevant equations into its loss function

# Introduction

- Why PINNs?
  - Grid-free solution
  - No need for large labelled datasets for training: Model learn by satisfying the underlying physical equation using input points
- **Research Problem:** Solving TDSE is challenging and computationally expensive for higher dimensions when using numerical methods like Crank-Nicholson
- **Research Question:** Can PINNs efficiently and accurately solve the TDSE?

# Dataset

- No labelled data
- Use random generated data for inputs, x and t
- Time (t): Uniform distribution with the problem’s time domain
- Space (x): Uniform distribution for the baseline model and a normal distribution centred around the position of the quantum dot at the given time

# Tasks

- Solve **Schrödinger equation** for:
  - Baseline Model: Electron inside a 1D harmonic oscillator
  - Quantum Dot Model: Electron inside a time-dependent moving quantum dot
  - 2D Quantum Dot: An experiment to check how both Crank-Nicholson and PINN handle higher dimensions
- Training Strategies:
  - Causal Training
  - Normalization Enforcement

# Methodology

- Problem Setup
- PINN Model Architecture
- Loss Function
- Training Strategies
- Evaluation Metrics ?

# Problem Setup

- The **Schrödinger equation:** $i\hbar \frac{\partial \psi(x,t)}{\partial t} = \frac{-\hbar^2}{2m} \frac{\partial^2 \psi(x,t)}{\partial x^2} + V(x,t)\psi(x,t)$
- Where,
  - For baseline model: $V(x) = \frac{1}{2} m \omega^2 x^2$
  - For moving quantum dot model: $V(x,t) = \frac{1}{2} m \omega^2 (x - x_{qd}(t))^2$
- **Goal:** Predict the wavefunction, Ψ(x, t) using PINNs

# PINN Architecture

- Input: x, t
- Output: **u, v**, where **Ψ = u + iv**
- Fully connected feed forward neural networks:
  - Baseline Model: 6 HL, 512 neurons each
  - Moving Quantum Dot Model: 12 HL, 512 neurons each
- Activation Function:
  - First layer: tanh
  - Rest: SiLU

# Loss Function

- Total loss, $L_{total} = \lambda_{pde} L_{pde} + \lambda_{ic} L_{ic} + \lambda_{bc} L_{bc}$
- Physics Loss (PDE loss), $L_{pde}$:
  - Residual, $R_{pde} = i\hbar \frac{\partial \psi(x,t)}{\partial t} + \frac{\hbar^2}{2m} \frac{\partial^2 \psi(x,t)}{\partial x^2} - V(x,t)\psi(x,t)$
  - Model output: u, v
  - Physics loss, $L_{pde} = \frac{1}{N_{colloc}} \sum^{N_{colloc}} (R_{pde}^{real})^2 + (R_{pde}^{img})^2$
- Initial Condition Loss, $L_{ic}$
- Boundary Condition Loss, $L_{bc}$

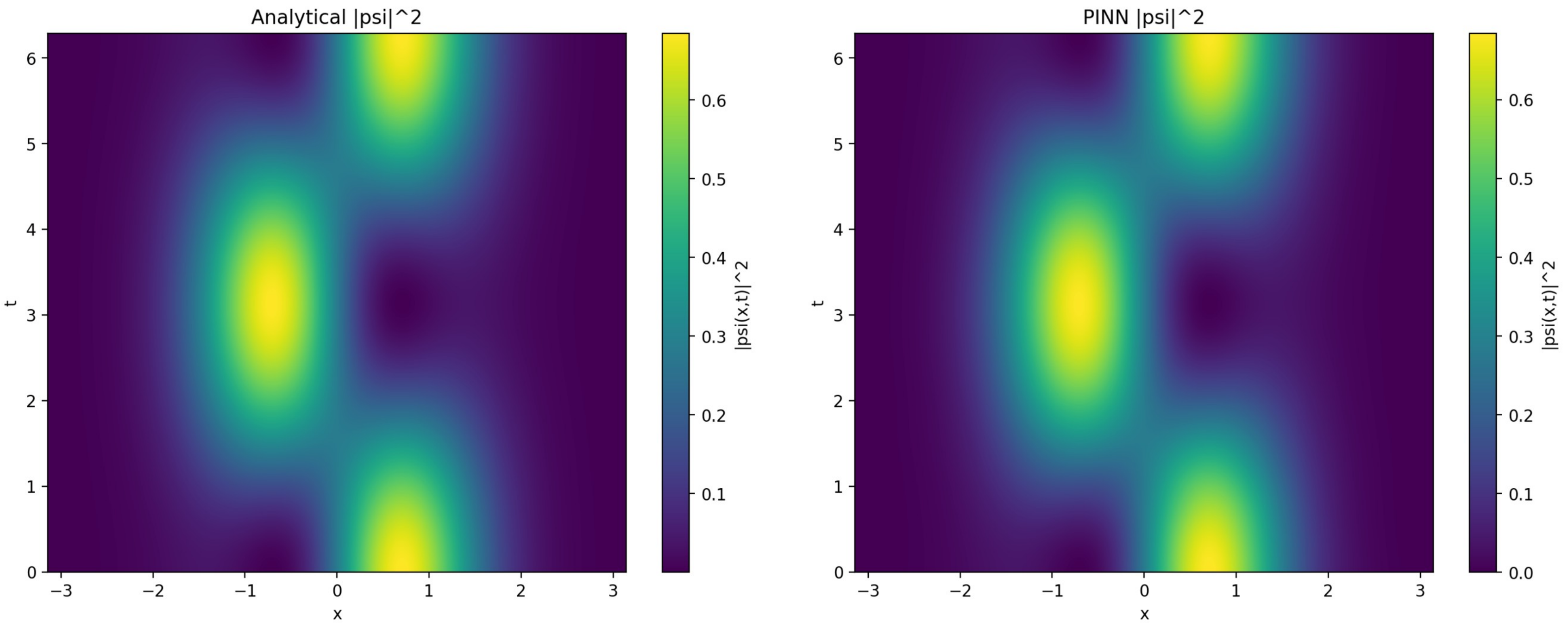
# Training Strategies

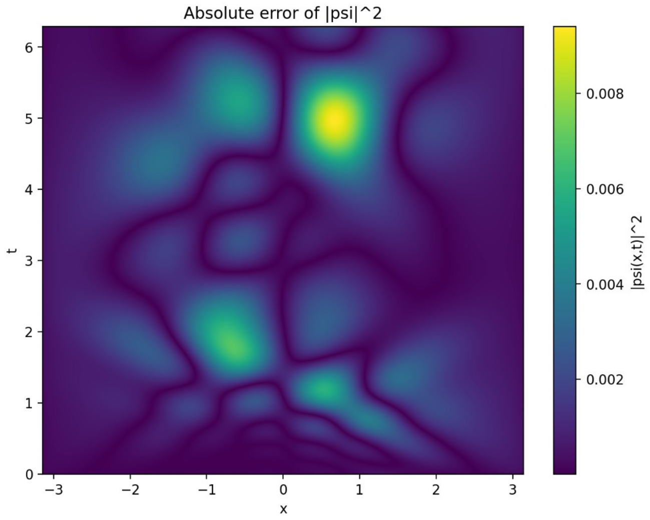
- Causal Training:
  - Ensures the model learn in a time ordered manner
- Normalization Enforcement:
  - Force the wavefunction to stay normalized during training
- Both helped improve the accuracy of the moving quantum dot model

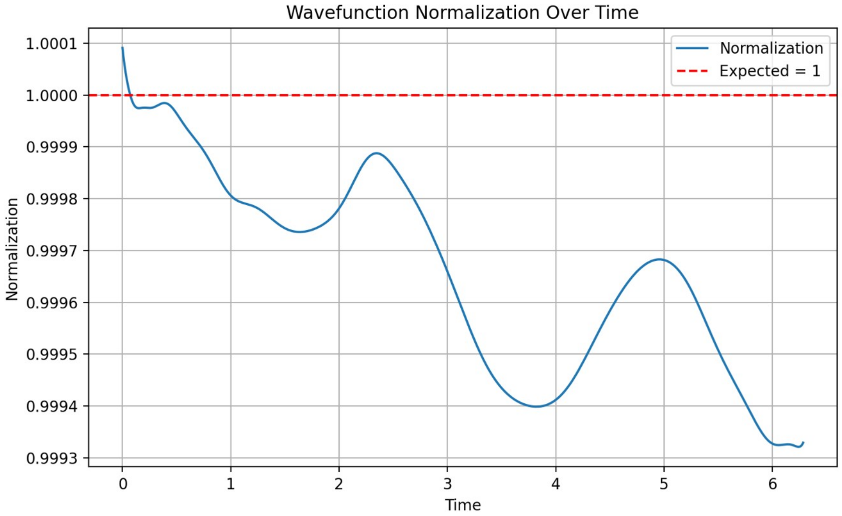
# Results

- Baseline
- Moving Quantum Dot
  - Vanilla model
  - Causal model
  - Normalization model
- Time-Dependent Model
- 2D Quantum Dot Model

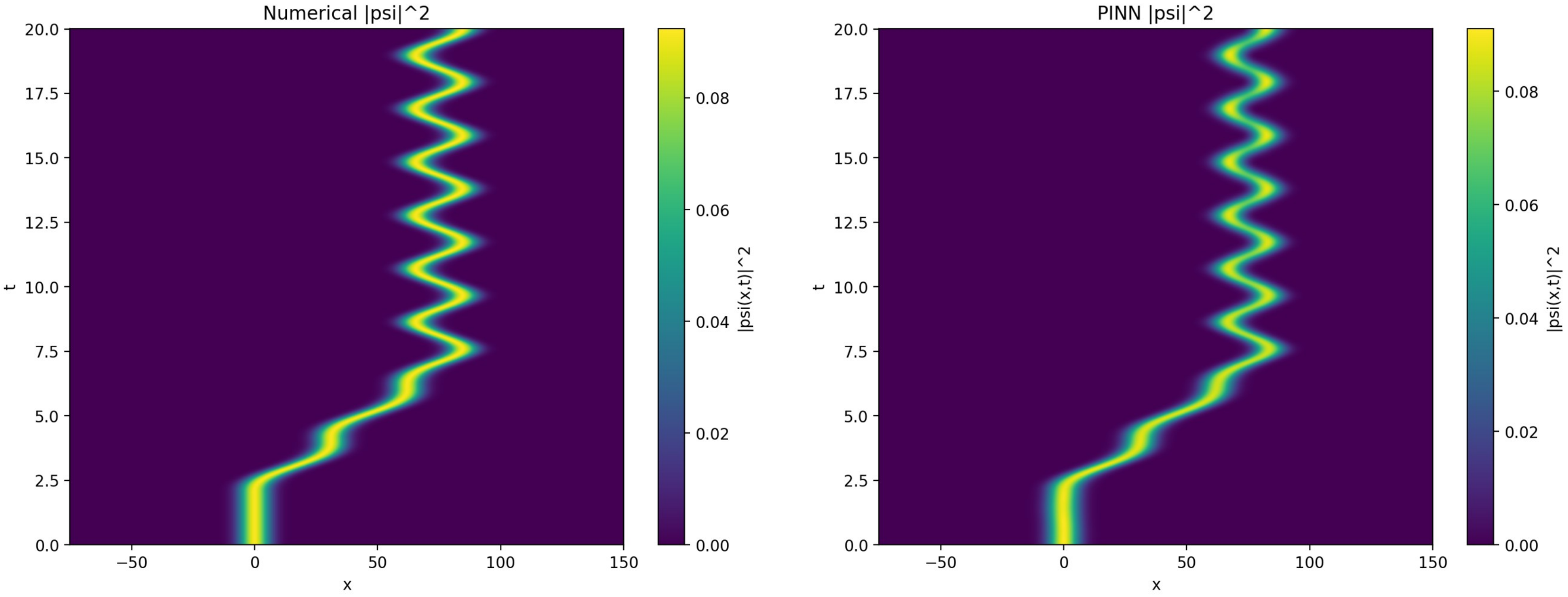
# Baseline

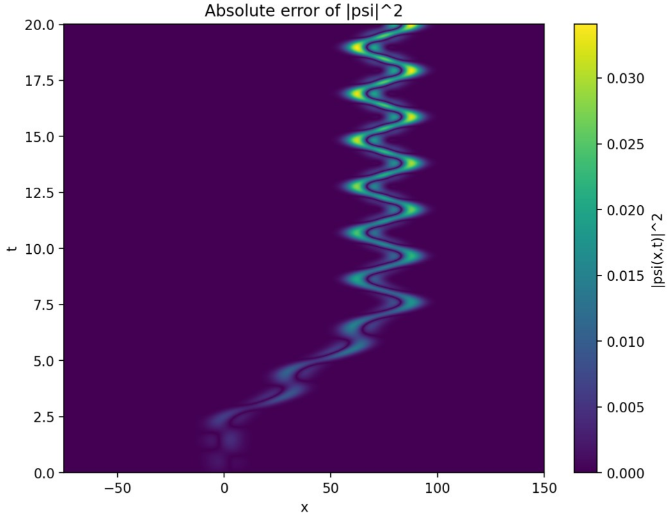

Analytical |psi|^2
PINN |psi|^2
t
x
|psi(x,t)|^2
0
1
2
3
4
5
6
−3
−2
−1
0.0
0.1
0.2
0.3
0.4
0.5
0.6


Absolute error of |psi|^2
t
x
|psi(x,t)|^2
0.008
0.006
0.004
0.002

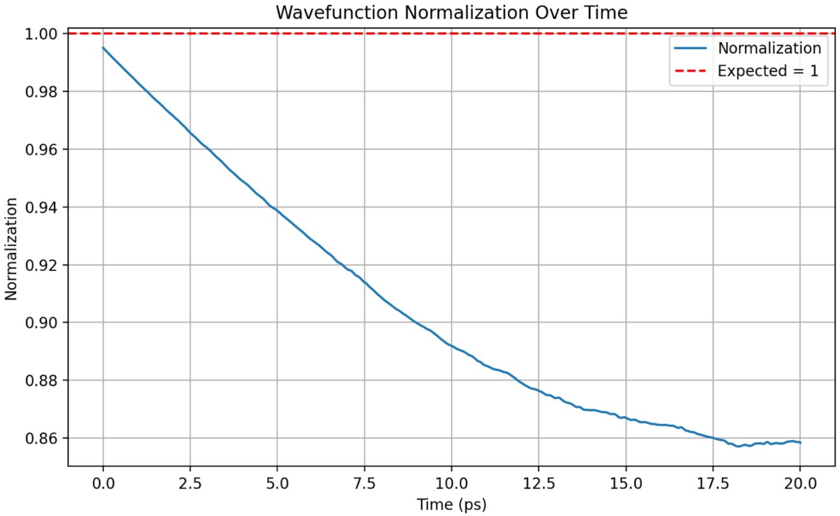

Wavefunction Normalization Over Time
Normalization
Expected = 1
Time

# Moving Quantum Dot



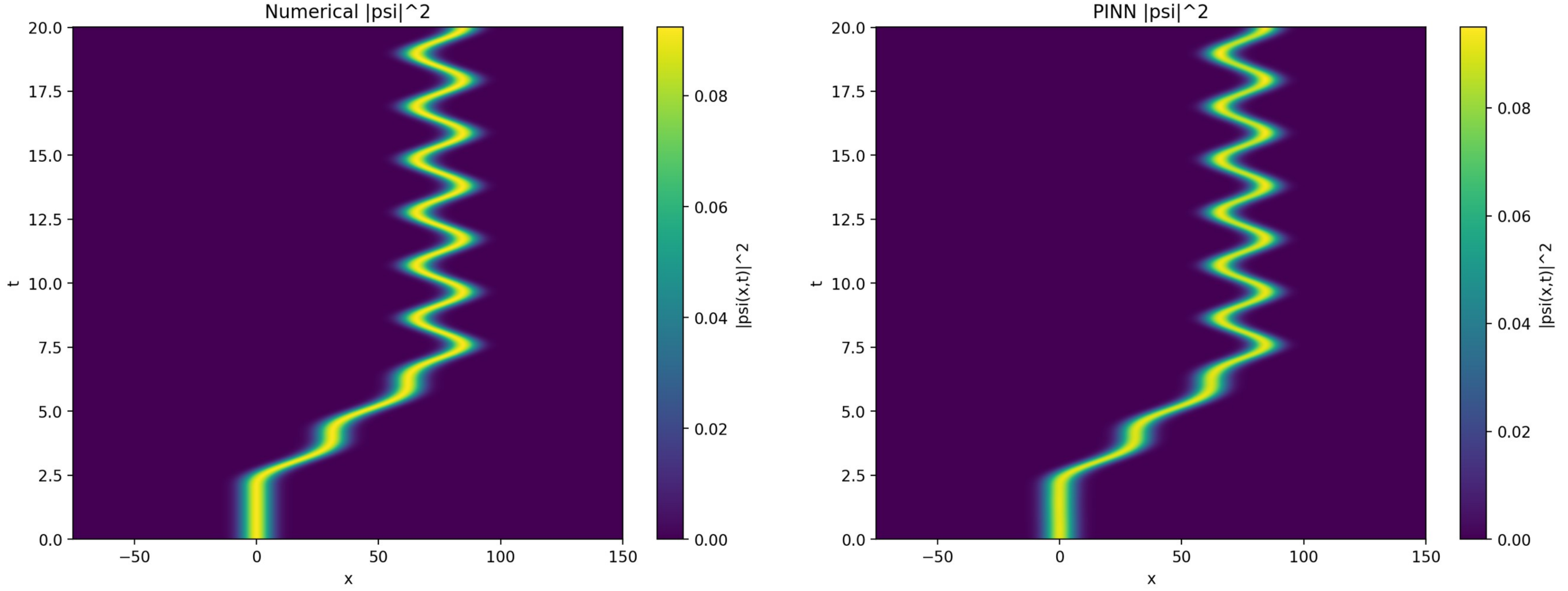


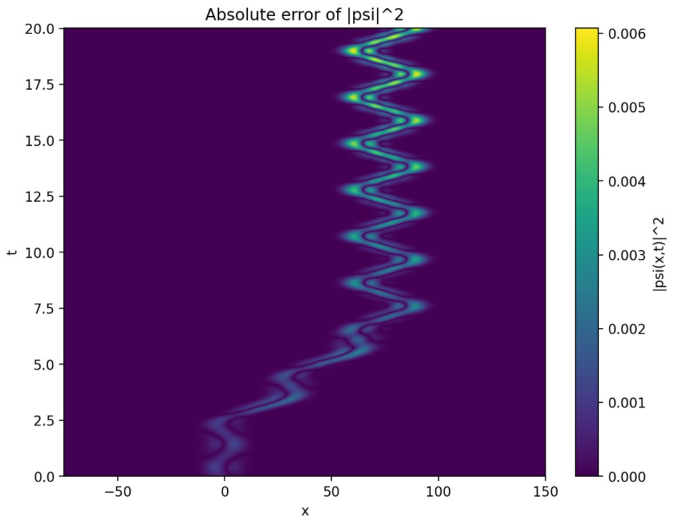
Absolute error of |psi|^2
t
x
|psi(x,t)|^2

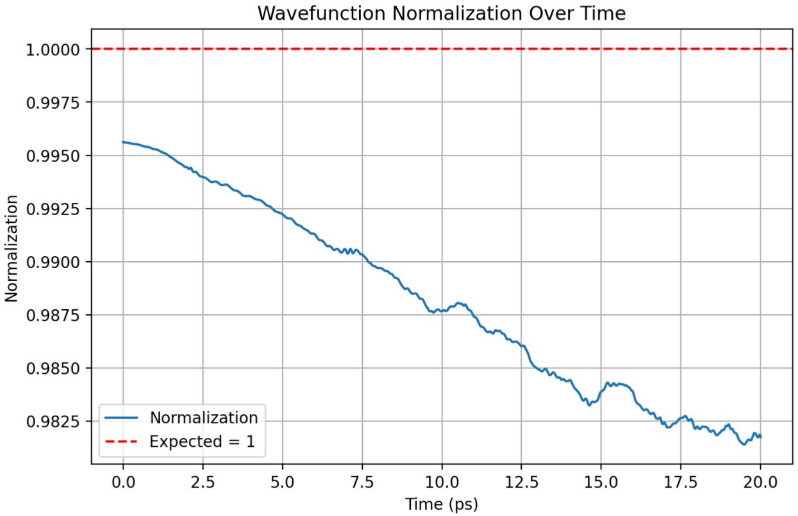
Wavefunction Normalization Over Time
Normalization
Expected = 1
Normalization
Time (ps)

# Causal Training



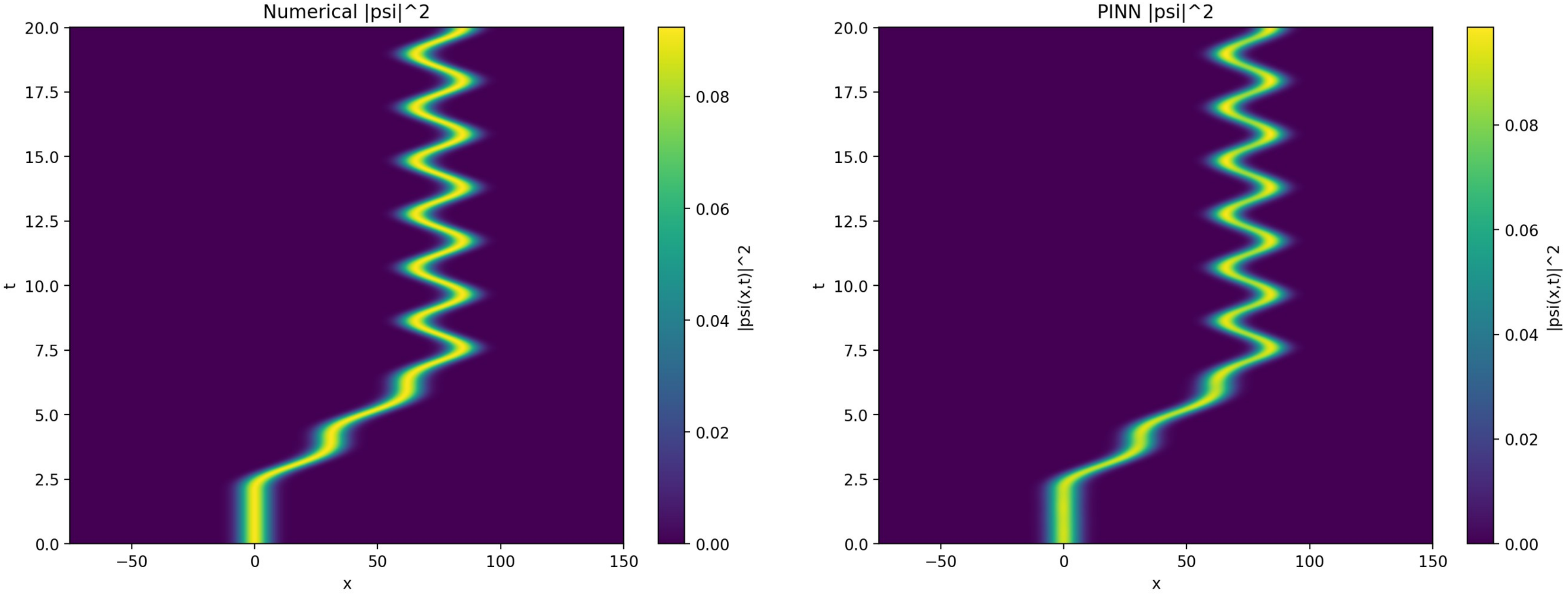


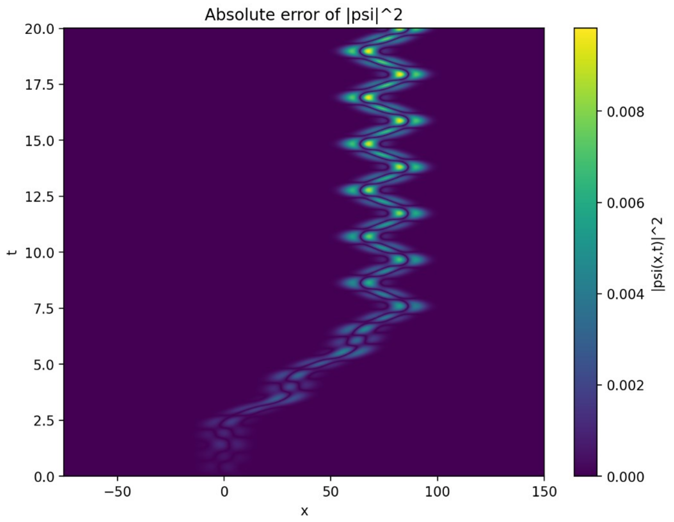
Absolute error of |psi|^2
t
x
|psi(x,t)|^2

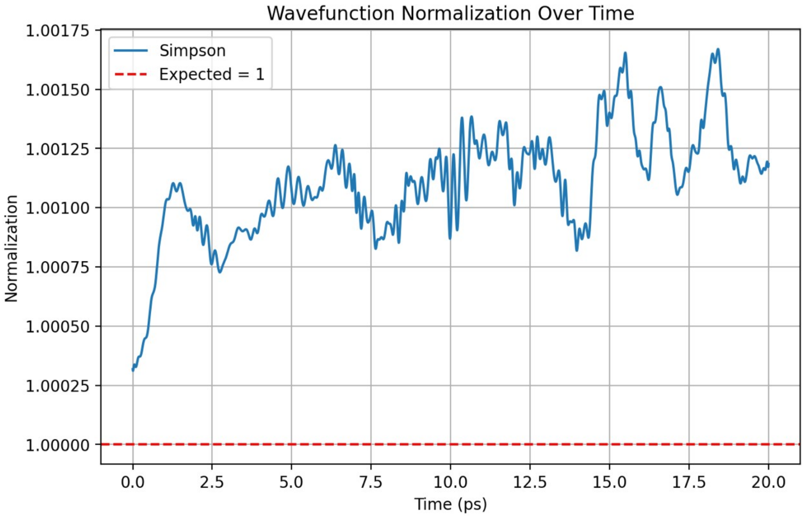



Wavefunction Normalization Over Time
Normalization
Time (ps)
Normalization
Expected = 1

# Normalization Enforcement



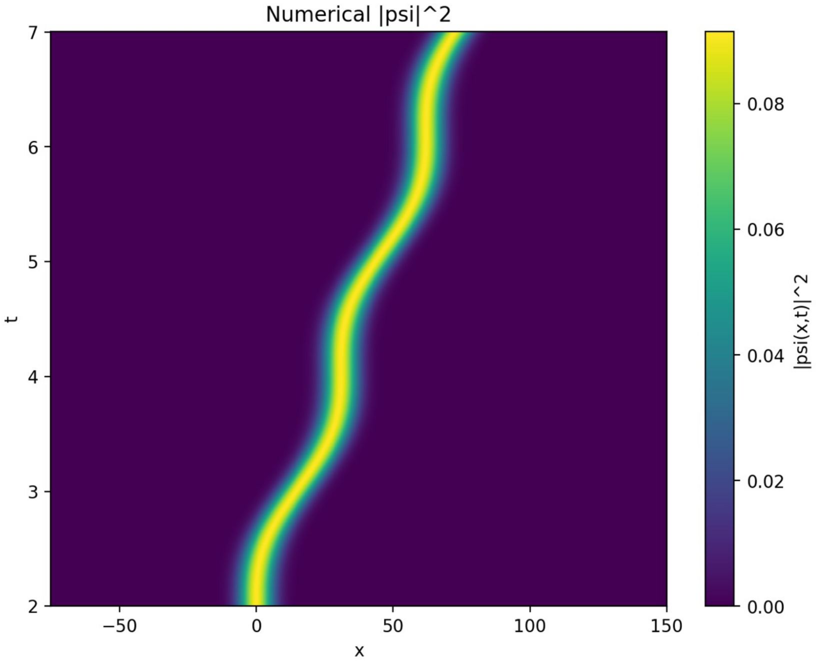

Absolute error of |psi|^2
t
x
|psi(x,t)|^2
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
−50
0
50
100
150
0.008
0.006
0.004
0.002
0.000

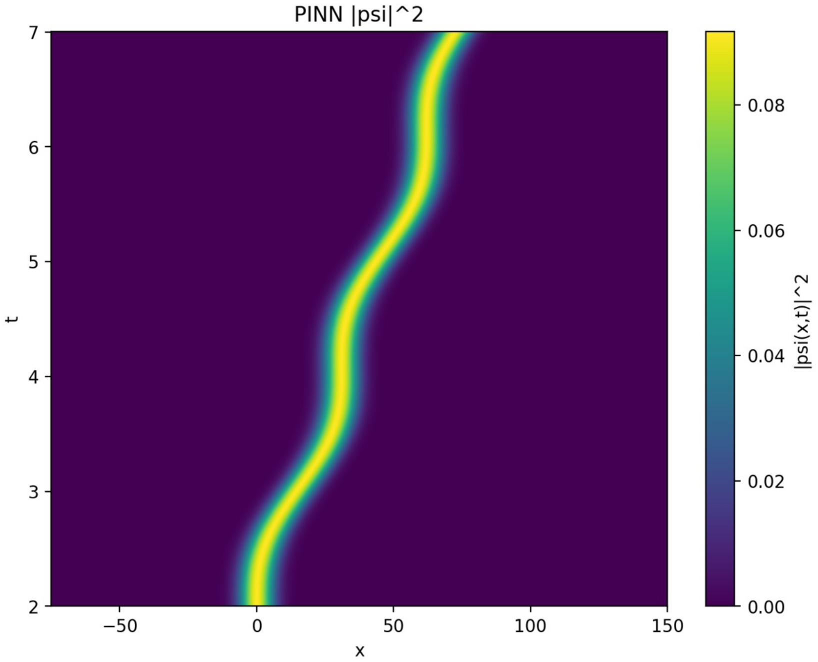

Wavefunction Normalization Over Time
Simpson
Expected = 1
Normalization
Time (ps)
1.00175
1.00150
1.00125
1.00100
1.00075
1.00050
1.00025
1.00000
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0

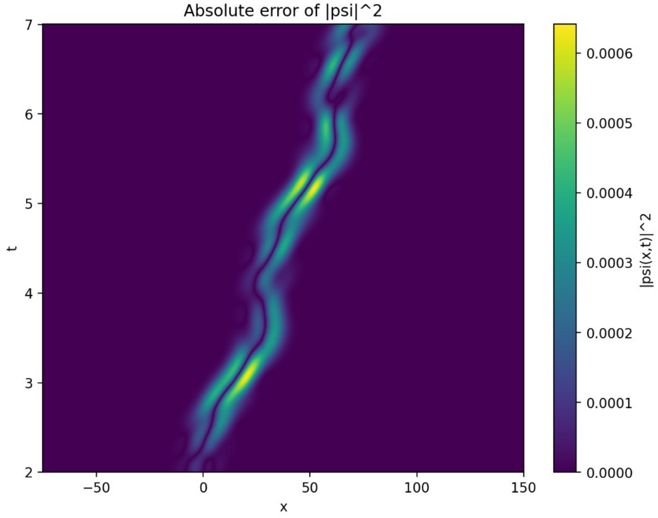
# Result Comparison

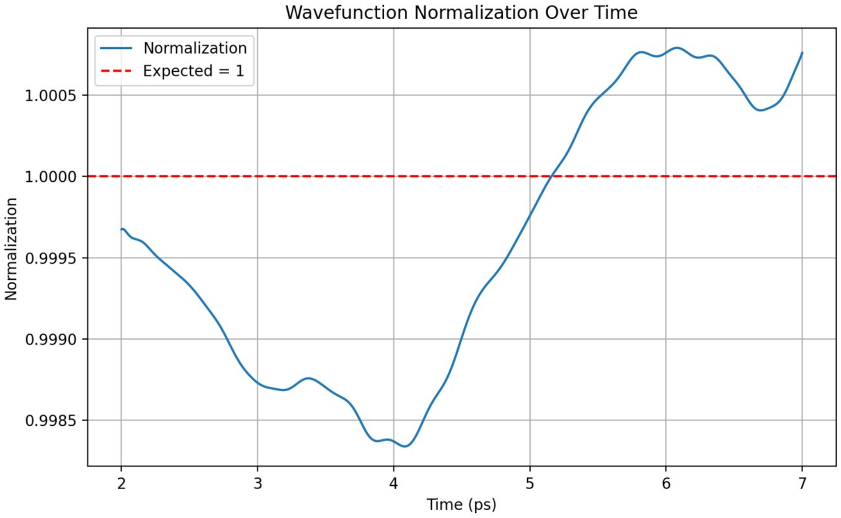
| Moving QD Models | $\Delta\Psi^2$ - max | Wavefunction Normalization | | Training Time (minutes) |
|---|---|---|---|---|
| | | Mean | Standard deviation | |
| Vanilla Model | 0.03409 | 0.90468 | 0.04191 | 394 |
| Causal Model | 0.00607 | 0.98813 | 0.00447 | 381 |
| Norm Model | 0.00982 | 1.00111 | 0.00023 | 492 |

# Time-Dependent Model

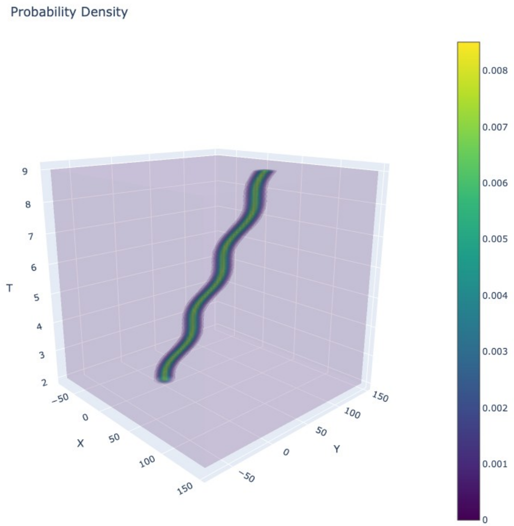


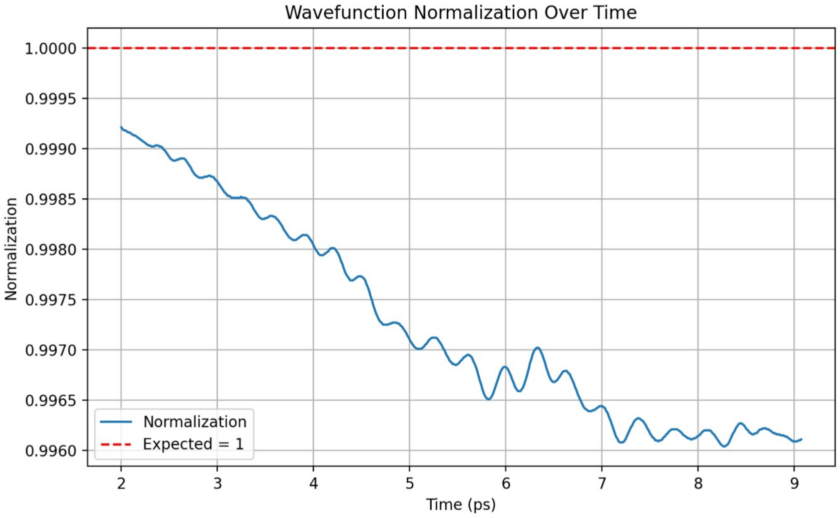




Absolute error of |psi|^2
t
x
|psi(x,t)|^2
7
6
5
4
3
2
−50
0
50
100
150
0.0006
0.0005
0.0004
0.0003
0.0002
0.0001
0.0000


Wavefunction Normalization Over Time
Normalization
Expected = 1
Normalization
Time (ps)
1.0005
1.0000
0.9995
0.9990
0.9985
2
3
4
5
6
7

# 2D Quantum Dot



# Discussion

- PINNs are flexible but they need improvements in handling longer time evolution
- Training strategies such as causal training and normalization enforcement improve the vanilla model
- 2D model showed promise for higher-dimensional problems
- Challenges:
  - Instability over time
  - Training sensitivity to model architecture
  - High computational cost for deeper networks

# Conclusion

- Summary: PINNs are capable of solving TDSE
- Real-world use:
  - Quantum computing
  - Material design
- Future works:
  - Hybrid solvers: PINN + Crank-Nicholson
  - Generalization: $V_{QD}$, $\omega$, $V(x, t)$ etc as inputs

Thank You