# Particle In a Box

## Using Physics Informed Neural Networks

**Team 1**
Harsh Singh
Jaideep Nirmal A J
Sarang Gund

Spring 2025

# Problem Statement

Solve for the wavefunctions of a particle in a box of length L = 1 by making use of PINNs to solve

$$-\frac{\hbar^2}{2m}\frac{d^2\Psi}{dx^2} = E\Psi$$

1. Values of $E = \frac{n^2\pi^2\hbar^2}{2m}$, $n \in \mathbb{N}$ to get different Wavefunctions
2. If the energy eigenvalues were not known, how would the neural network have to be changed to find the eigenvalues also?

# Physics Informed Neural Networks

1. NN = Universal Function Approximator
2. Predicts solutions to differential equations
3. Trained to minimize data loss (if available)
4. Minimizes PDE residuals (physics loss)
5. Learns solution without needing mesh or full data

# Brief Working

1. Collocation points $\{(x_i)\}_{i=1}^{N_p}$ are sampled in the domain.
2. The network outputs $\Psi_\theta(x_i, t_i)$, which are plugged into the PDE to compute residuals.($\theta$ is parameter of the Neural Networks)
3. Boundary and/or initial conditions are included in the loss function as extra terms to consider them

# Model Architecture

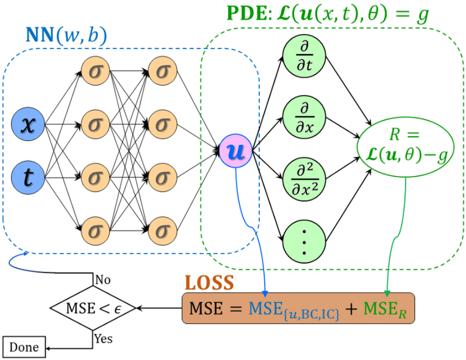


Figure: General Pinn architecture

# Model Architecture

1. 5 Sequential, Linear Layers
2. Coupled with activation function- Tanh(Experimental)
3. The eigen number **n** is globally defined
4. Input: $x$ space
5. Output: $\Psi_n(x)$
6. Model learns the form of a particular eigen function of the TISE and predicts the $\Psi_n(x)$ values on the domain of a given box ($x$ space)
7. To obtain solution for other $n$, we will retrain the model for the desired **n**

# Loss Functions

Four loss functions are defined

1. Boundary Condition Error (BCE)
2. Physics Loss
3. Norm Loss
4. Trivial Loss

$$Loss = \lambda_1 \mathcal{L}_{BCE} + \lambda_2 \mathcal{L}_{Physics} + \lambda_3 \mathcal{L}_{Norm} + \lambda_4 \mathcal{L}_{Trivial}$$

# MSE (Boundary Points)

$$\mathcal{L}_{MSE} = \frac{1}{2}\sum\left(\psi_{\text{true}}(x_i) - \psi_{\text{pred}}(x_i)\right)^2$$

- $X_i :=$ Boundary points

1. A Standard error function for NNs
2. Ensures the solution has the right *bounds*
3. We know $\Psi(x)$ only at two points - boundaries

# Physics Loss

$$\mathcal{L}_{phy} = \sum_{i=0}^{N} -\frac{\hbar^2}{2m}\frac{d^2}{dx^2}\hat{\Psi}_{pred}(x_i) - E\hat{\Psi}_{pred}(x_i)$$

1. The above term should ideally be 0
2. Ensures the solution is *physically behaving* as expected

# Norm Loss

$$\mathcal{L}_{Norm} = \left| \int_X \Psi_{pred}(x)\Psi^*_{pred}(x)dx - 1 \right|^2$$

Condition from Quantum Mechanics / Total Probability theorem

1. We use this to make sure the solution predicted by NN is a valid Quantum wavefunction
2. If we don't use this, the solution will be physically non-nonsensical

# Trivial Solution

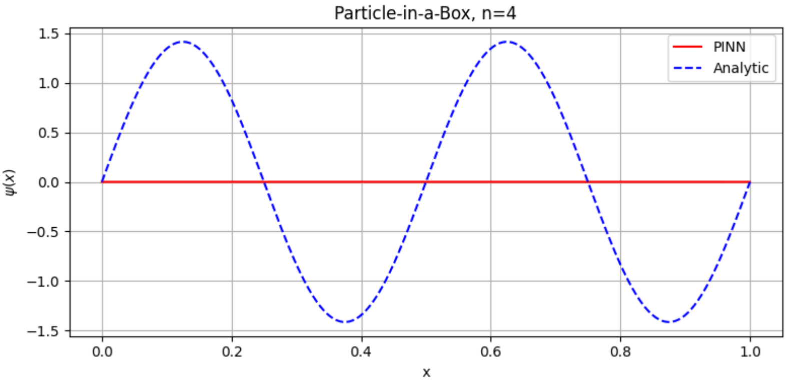


Figure: Trivial solution for the equation $-\frac{\hbar^2}{2m}\frac{d^2}{dx^2}\Psi = E\Psi$

# Trivial Loss

$$\mathcal{L}_{trivial} = \frac{1}{(< |\psi_{pred}(x)| > + 10^{-6})^2}$$

The trivial solution of the TISE is $\Psi(x) = 0 \; \forall \; x \in X$

1. We do not want the model to converge the solution to trivial
2. The NN can easily converge to a trivial solution
3. We penalise the model if it tries to converge to a trivial function
4. Trivial solution $\implies$ average of absolutes is 0
5. the $10^{-6}$ ensures a finiteness

# Curriculum Learning

1. A Strategy to learn a function by prioritising the loss functions
2. The loss weights $\lambda$s are changed in different epochs
3. Initially, weightage to boundary condition, Physics (PDE) Loss and Trivial loss is high
4. Once the solution is converged to an extent, we refine it by giving higher weightage to norm loss

# Performance Metrics

- **MSE (Mean Squared Error):** Measures average squared error between true and predicted values.

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(\Psi_{pred}(x_i) - \Psi_{true}(x_i))^2 \qquad \text{Ideal Value} \to 0$$

- **Energy Deviation:** Difference between predicted and true energy.

$$\Delta E = |E_{\text{true}} - \hat{H}\Psi_{pred}(x)| \qquad \text{Ideal Value} \to 0$$

- **Fidelity:** Overlap between predicted and true quantum states.

$$\mathcal{F} = \left|\int \psi_{\text{true}}(x)\,\psi_{\text{pred}}(x)dx\right|^2 \qquad \text{Ideal Value} \to 1$$

- **Correlation Coefficient ($\rho$):** Strength of linear correlation between true and predicted.

$$\rho_{\Psi_{pred},\psi_{true}} = \frac{\mathrm{Cov}(\Psi_{pred},\Psi_{true})}{\sqrt{\mathrm{Var}(\Psi_{true})}\cdot\sqrt{\mathrm{Var}(\Psi_{pred})}} \qquad \text{Ideal Value} \to \pm 1$$

# Model Performance

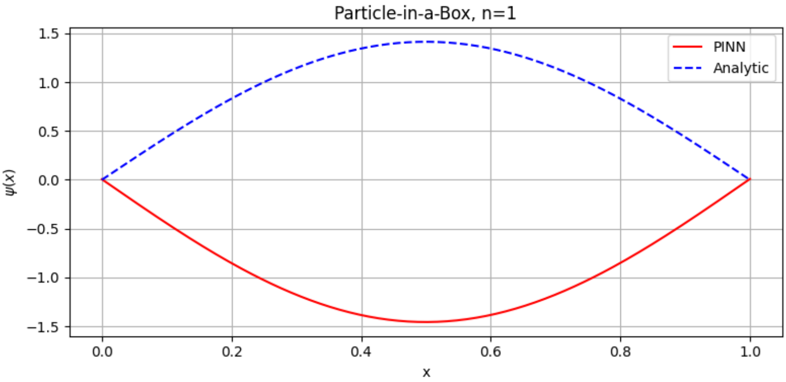


Figure: Predicted wave Function for $n = 1$

# Model Performance

| | Metric | Value |
|---|---|---|
| 0 | MSE | 4.110349 |
| 1 | Energy Deviation | 0.316992 |
| 2 | Fidelity | 1.057629 |
| 3 | Correlation Coefficient | -0.999998 |

Figure: Metrics for n = 1


Figure: Loss for n = 1

# Model Performance

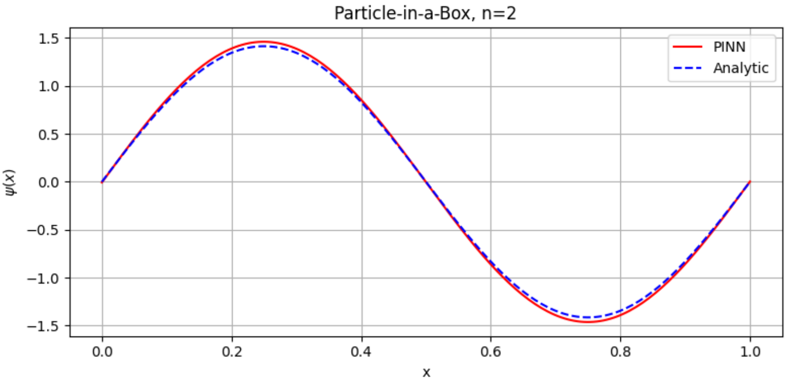


Figure: Predicted wave Function for n = 2

# Model Performance

| | Metric | Value |
|---|---|---|
| 0 | MSE | 0.001099 |
| 1 | Energy Deviation | 1.369095 |
| 2 | Fidelity | 1.067306 |
| 3 | Correlation Coefficient | 0.999999 |

Figure: Metrics for n = 2


Figure: Loss for n = 2

# Model Performance

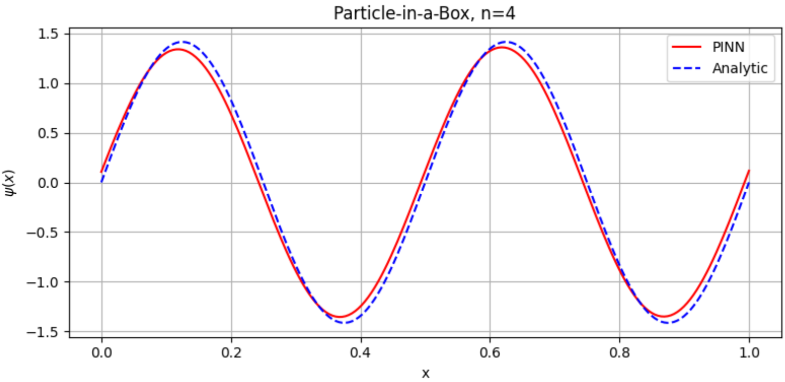


Figure: Predicted wave Function for $n = 4$

# Model Performance

| | Metric | Value |
|---|---|---|
| 0 | MSE | 0.008597 |
| 1 | Energy Deviation | 6.806664 |
| 2 | Fidelity | 0.906560 |
| 3 | Correlation Coefficient | 0.996536 |

Figure: Metrics for n = 4

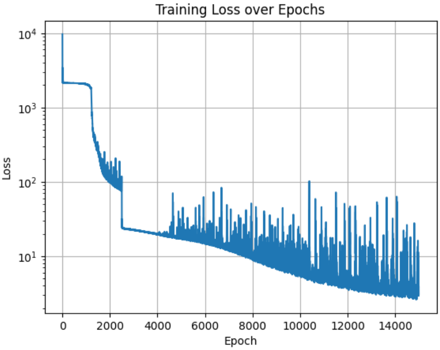


Figure: Loss for n = 4

# Utility of PINN

How is a Neural Network relevant here?

Regular approach

1. Find the non trivial solutions to $(H - \mathbb{I}\ E)\Psi = 0$
2. Usually done using QR Algorithm (‘scipy‘/LAPACK) in $O(n^3)$
3. To find eigen vector $n = 3$ for in domain $[a, b]$ for 3 cases :

- $A =$ `linspace(a, b, 1000)`|`B = linspace(a, b, 5000)`|$C =$ `linspace(a, b, 8000)`| we need to run the computation code again and again

# Utility of PINN

How is a Neural Network relevant here?
Neural Networks

1. Train the Model for a particular eigen state $n$ on a large $x$ space
2. The model *knows* the behaviour of eigen function very well
3. Pass the different $x$ space domains to the model predicts the eigen wave function
4. Running NN model $\implies$ (usually) a matrix multiplication, efficiently implemented in $O(n^{\log_2 7}) \approx O(n^{2.7})$ (Strassen's Algorithm)

So, NNs can give a speed up for large $N$ and repeated calculation of a specific eigen function.

# What if Energy Eigenvalues are Unknown?

1. Assume we do not know the formula for $E_n = \frac{n^2\pi^2\hbar^2}{2m}$.
2. We treat $E$ as a trainable parameter or explore it manually.
3. Strategy: Fix a trial value of $E$, let the PINN solve the differential equation.
4. The network converges to actual eigenvalue closest to guessed $E$
5. Output $\Psi(x)$ becomes the corresponding eigenfunction.

# Changes in PINN for Unknown $E$

1. Network Output: PINN now outputs both $\Psi(x)$ and $E$

$$\text{Output:} \quad (\Psi_\theta(x), \; E_\theta)$$

2. Modified Loss Function:
   - PDE loss becomes:

$$\mathcal{L}_{PDE} = \sum -\frac{\hbar^2}{2m}\frac{d^2\Psi_\theta}{dx^2} - E_\theta \Psi_\theta(x)$$

3. $E$ is updated along with network weights using backpropagation.
4. Therefore, the neural network will converge to a wavefunction and its corresponding eigenvalue

# Model Performance when E is unknown

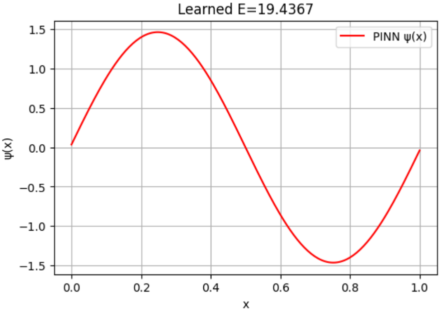


Figure: plot for Trial Energy $E$ close to 19

1. Energy Predicted: $E \approx 19.4367$
2. Closest actual eigenvalue corresponds to quantum number $n = 2$
3. Actual energy: $E_n = \dfrac{n^2\pi^2\hbar^2}{2mL^2}$, with $\hbar = 1,\ m = 1,\ L = 1$
4. So, $E_2 = \dfrac{4\pi^2}{2} \approx 19.7392$
5. Deviation: $\approx 0.3025$ (Relative Error $\approx 1.53\%$)

# Model Performance

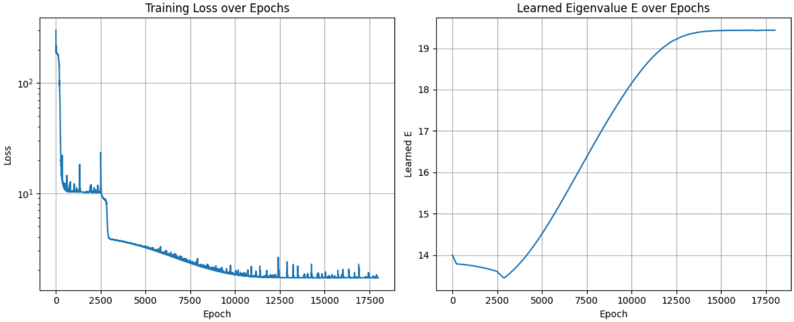


Figure: Loss and learned E for trial energy close to 19

# Model Performance when E is unknown

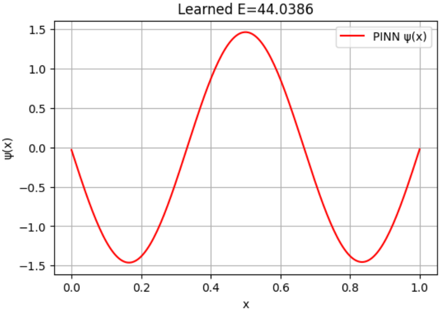


Figure: plot for Trial Energy $E$ close to 44

1. Energy Predicted: $E \approx 44.0386$
2. Closest actual eigenvalue corresponds to quantum number $n = 3$
3. Actual energy: $E_n = \frac{n^2\pi^2\hbar^2}{2mL^2}$, with $\hbar = 1,\ m = 1,\ L = 1$
4. So, $E_3 = \frac{9\pi^2}{2} \approx 44.41$
5. Deviation: $\approx 0.3746$ (Relative Error $\approx 0.84\%$)

# Model Performance

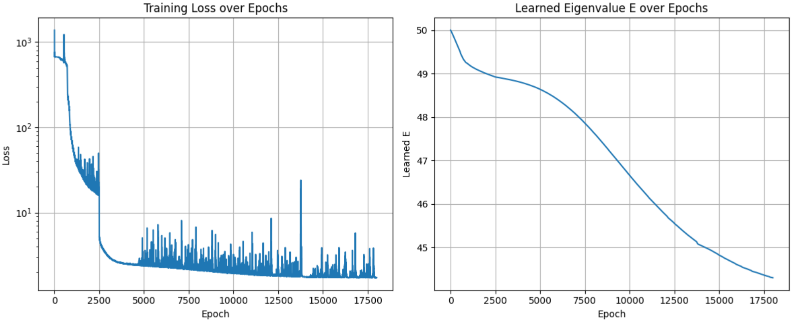


Figure: Loss and learned E for trial energy close to 44

# Scope of Improvement

1. Smoothening the curriculum learning weights - exponential decay instead of sudden jump
2. Loss Functions
   - Anchor Points - by QM properties, some points may only have a particular value
   - Rayleigh Loss - $|E_{true} - H\Psi|^2$
3. Better initialization - Xavier/Glorot Method
4. Adaptive Learning - model itself learns the weights
5. Better evaluation metrics