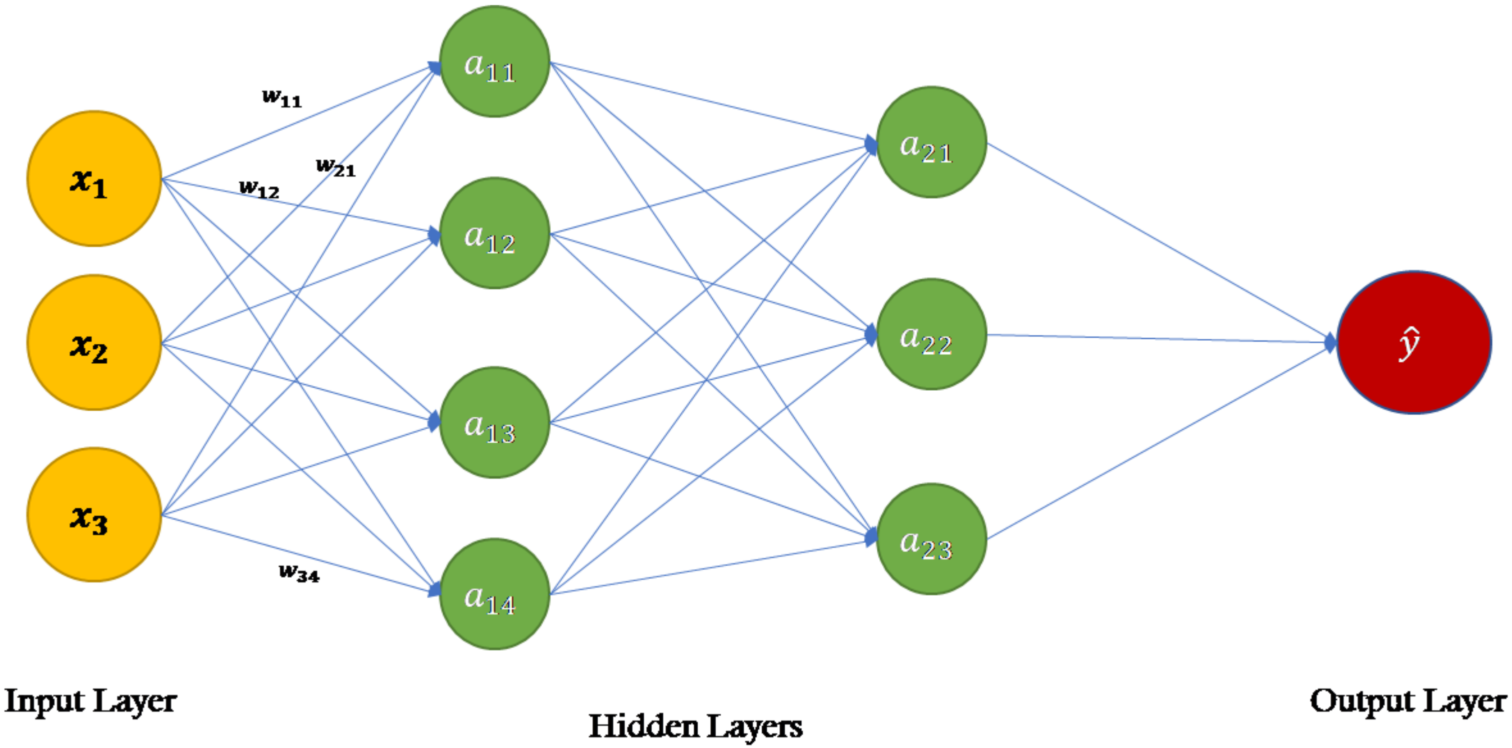
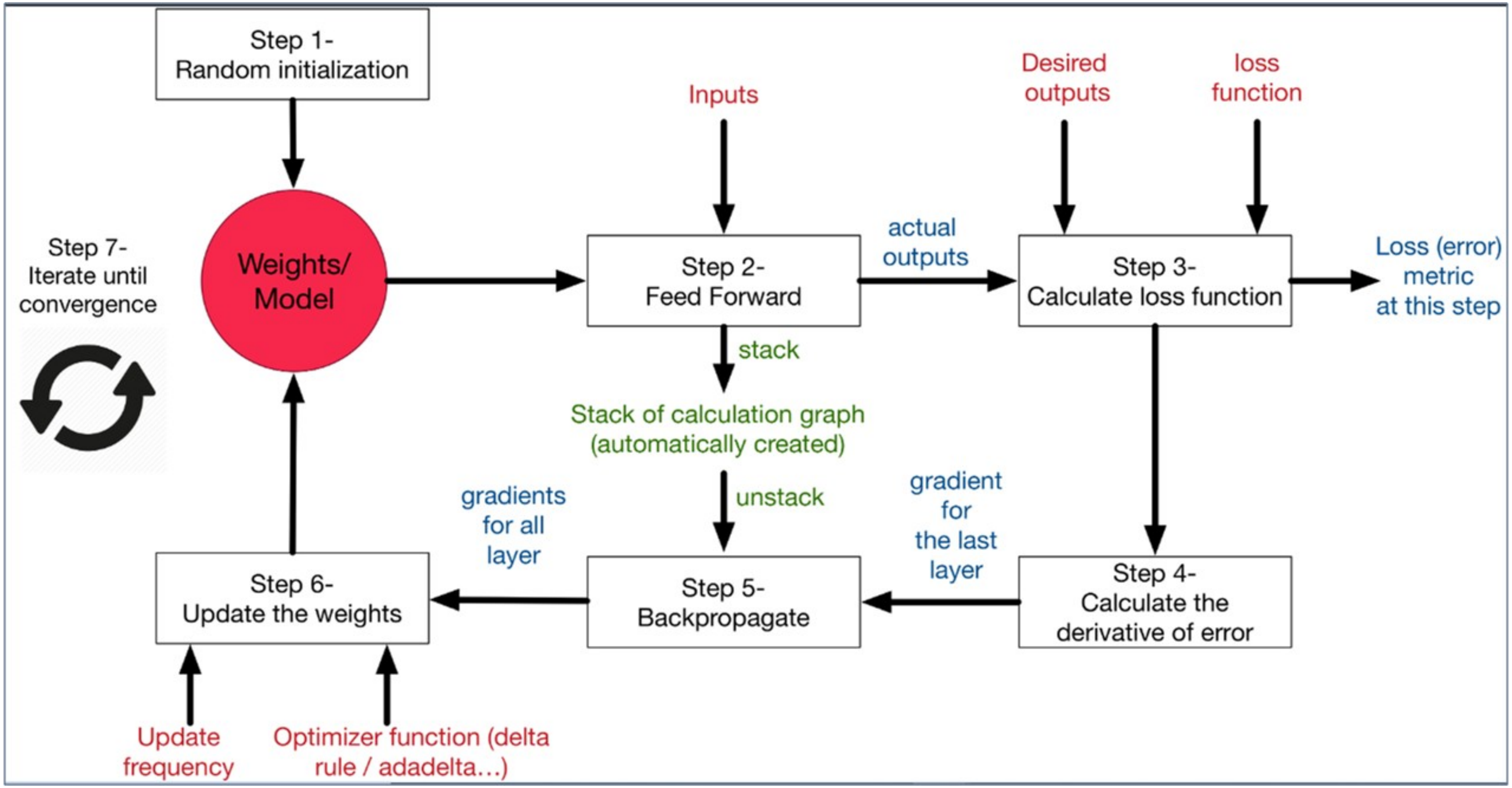
Physics-Informed Neural Networks for Many-Electron Time-Dependent Schrödinger Equation

Neural Networks

- A neural network is a function approximation that maps an input x to an output y, learning patterns from data.
- Mathematically:
  - Single layer model: $y=f(\sum(w_i x_i+b))$
  - Multilayer model: $a_{l+1}=f(W_l a_l+b_l)$
- Single-layer networks can only learn linear functions while deep networks (multiple hidden layers) enable learning complex, nonlinear mappings.
- Activation function: tanh $\rightarrow f(x)=\frac{e^x-e^{-x}}{e^x+e^{-x}}$

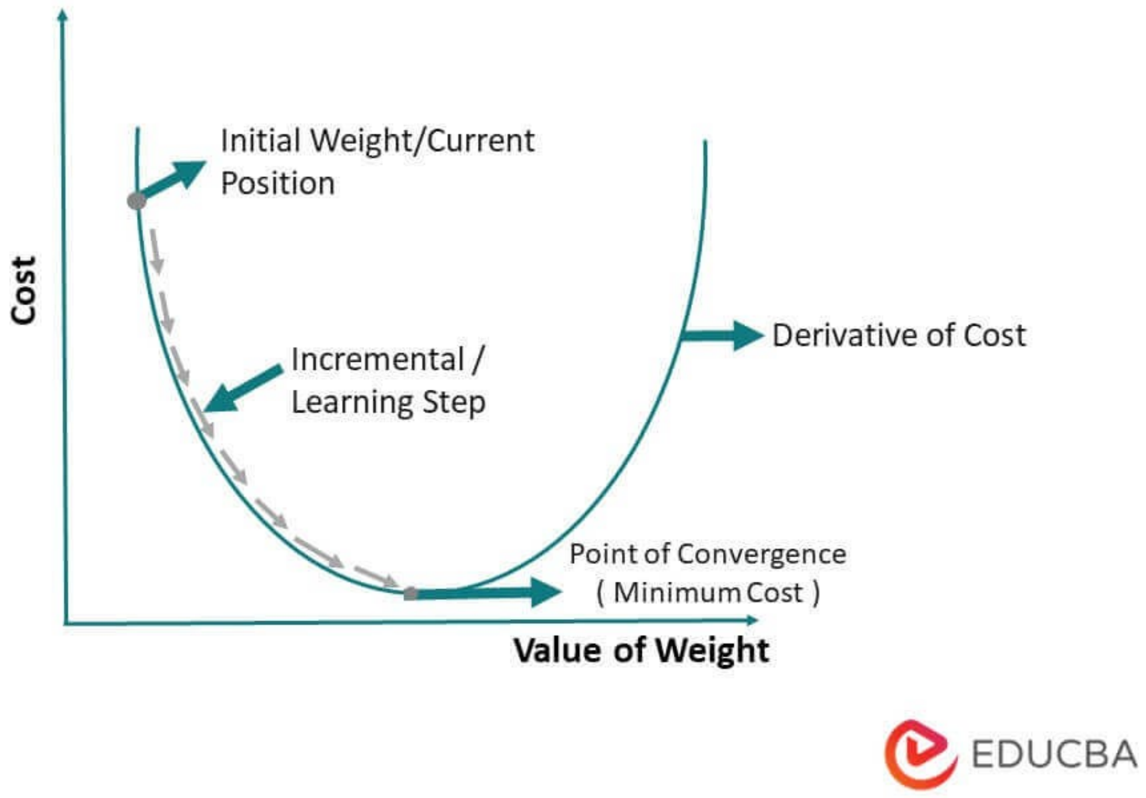


Multilayer Perceptron



Step 1-
Random initialization
Inputs
Desired
outputs
loss
function
Step 7-
Iterate until
convergence
Weights/
Model
Step 2-
Feed Forward
actual
outputs
Step 3-
Calculate loss function
Loss (error)
metric
at this step
stack
Stack of calculation graph
(automatically created)
unstack
gradients
for all
layer
gradient
for
the last
layer
Step 6-
Update the weights
Step 5-
Backpropagate
Step 4-
Calculate the
derivative of error
Update
frequency
Optimizer function (delta
rule / adadelta…)

Gradient Descent of Machine Learning



Physics-Informed Neural Networks (PINNs)

- PINNs are neural networks trained to solve partial differential equations (PDEs) by introducing physical laws into their loss function.
- Traditional numerical solvers can be computationally expensive for high-dimensional problems.
- Instead of training purely on data, PINNs ensure that solutions satisfy the governing equations (e.g., Schrödinger equation) via loss terms.

How Do PINNs Work?

- A neural network is trained to approximate the solution of a PDE.
- Loss functions:
  - Physics Loss (L_{PDE})
  - Initial condition loss ($L_{initial}$)
  - Boundary condition loss ($L_{boundary}$)
  - Total loss = $L_{PDE} + L_{initial} + L_{boundary}$

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

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

The PDE residual loss is computed as the mean squared error (MSE) of the residuals over all collocation points (x_i, z_i, t_i):

$$\mathcal{L}_{\text{PDE}} = \frac{1}{N_{\text{PDE}}}\sum_{i=1}^{N_{\text{PDE}}}\left[\left(\mathcal{R}_{\text{Re}}(x_i, z_i, t_i)\right)^2 + \left(\mathcal{R}_{\text{Im}}(x_i, z_i, t_i)\right)^2\right]$$

- N_{PDE}: Number of collocation points.
- **Collocation Points**: Points sampled throughout the spatial-temporal domain where the PDE is enforced.

Mathematical Formulation

Given the initial wave function $\psi_{\text{IC}}(x, z)$:

$$\psi(x, z, t = 0) = \psi_{\text{IC}}(x, z)$$

The initial condition loss is:

$$\mathcal{L}_{\text{IC}} = \frac{1}{N_{\text{IC}}} \sum_{i=1}^{N_{\text{IC}}} \left| \psi_{\text{NN}}(x_i, z_i, 0) - \psi_{\text{IC}}(x_i, z_i) \right|^2$$

- ψ_{NN}: Neural network's prediction at $t = 0$.
- N_{IC}: Number of initial condition points.

Physics-Informed Neural Networks for Many-Electron Time-Dependent Schrödinger Equation

- Develop and apply PINNs to solve the many-electron time-dependent Schrödinger equation (TDSE).
- Solving TDSE for many-electron systems is computationally intensive using traditional methods.

Research Project Roadmap

- Single-electron time-independent one-dimensional model
- Single-electron time-dependent one-dimensional model
    - Moving quantum dot
    - Double quantum dots with silicon charge qubits
- Single-electron three-dimensional model
- Many electron models
    - Quantum dot array

Single-Electron Time-Dependent 1D Model

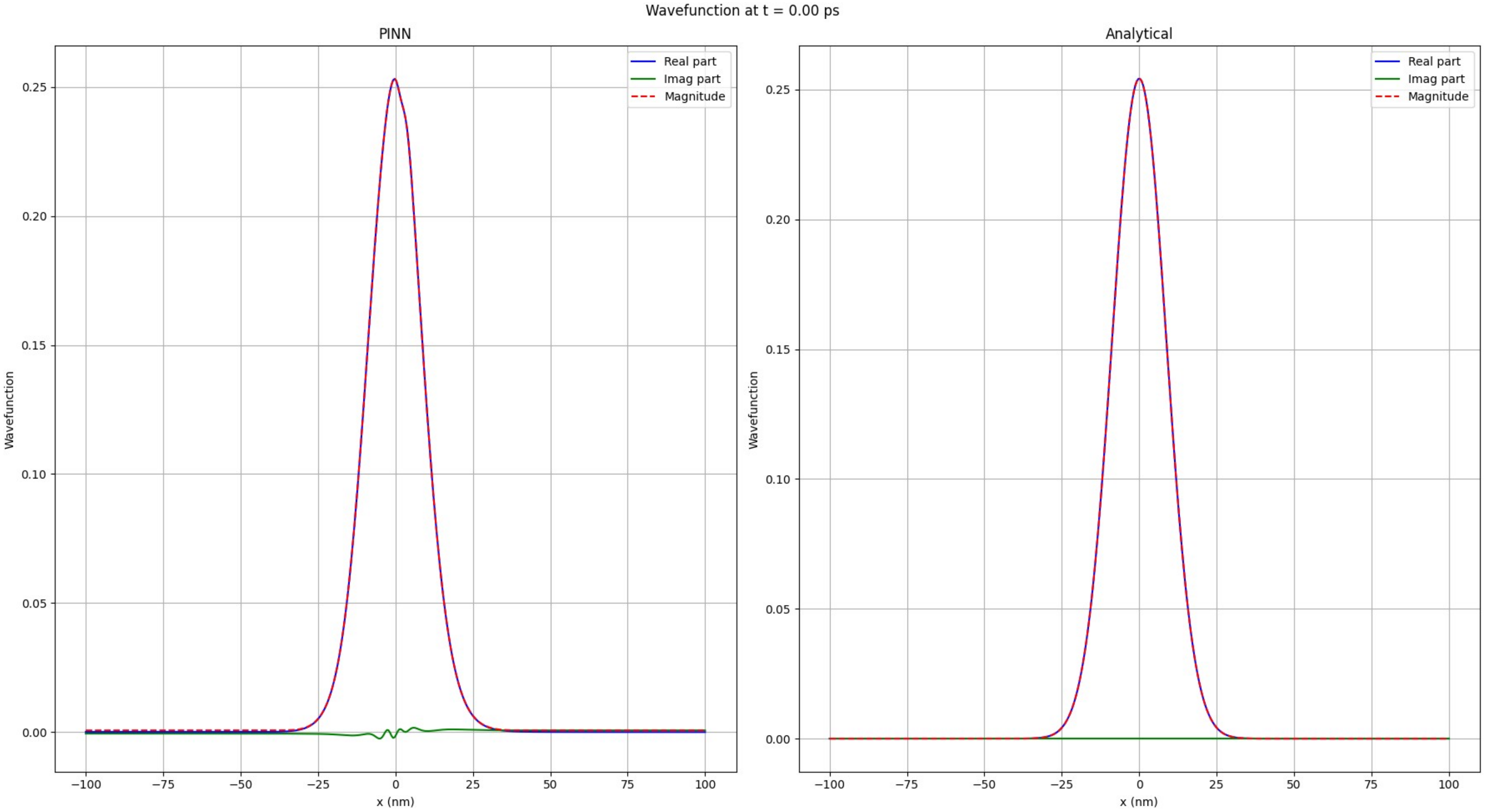
- Why start with single-electron models?
  - Serves as a benchmark for validating the PINN approach.
  - Easier to compare with analytical and numerical solutions.
  - Provides insights into training behaviour before scaling to multi-electron cases.
- Time-Dependent Schrodinger equation for a single electron in 1D harmonic potential:
  - $i\hbar \frac{\partial \psi(x,t)}{\partial t} = \frac{-\hbar^2}{2m} \frac{\partial^2 \psi}{\partial x^2} + \frac{1}{2} m \omega^2 x^2 \psi(x,t)$

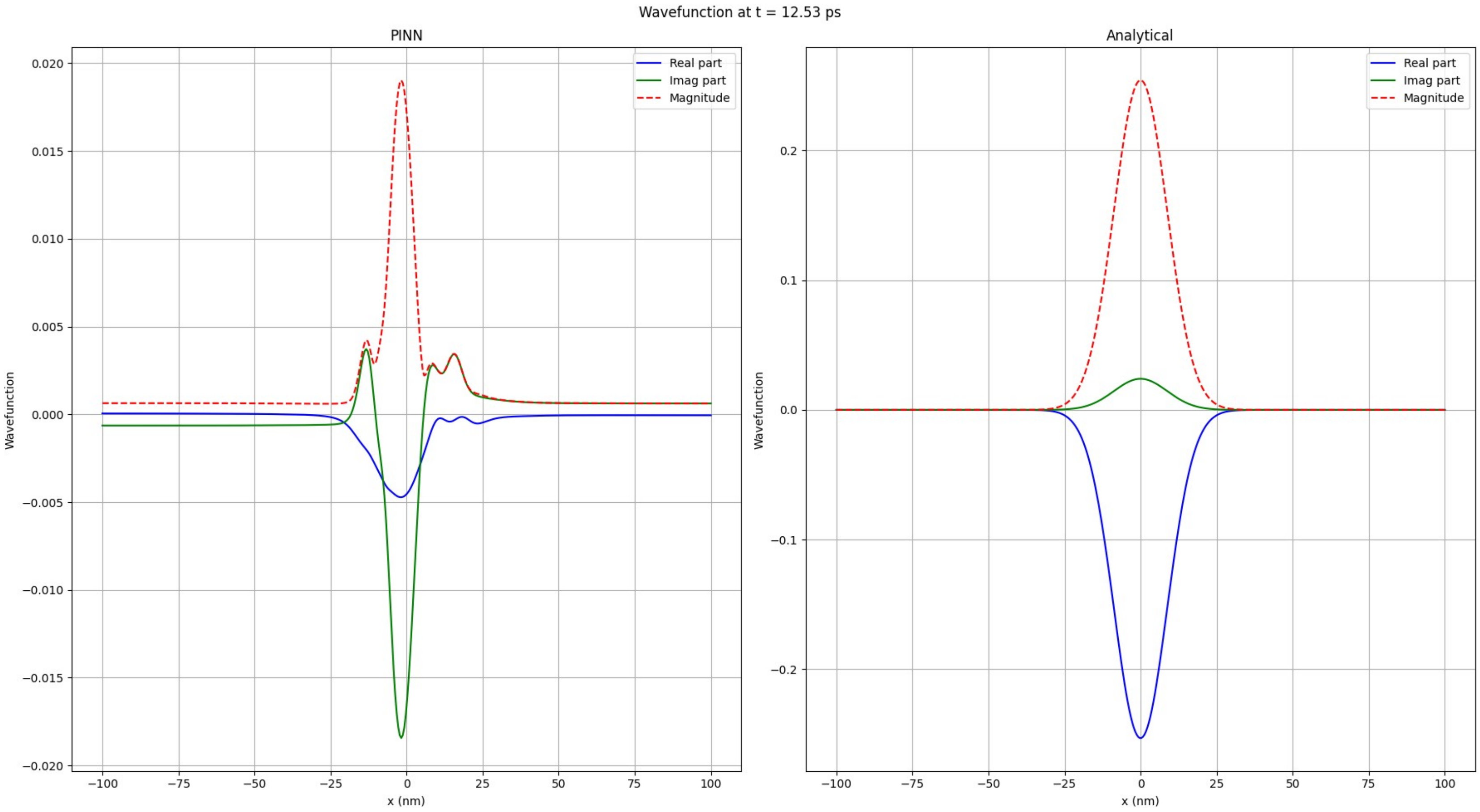
Neural Network Representation of the Schrödinger Equation

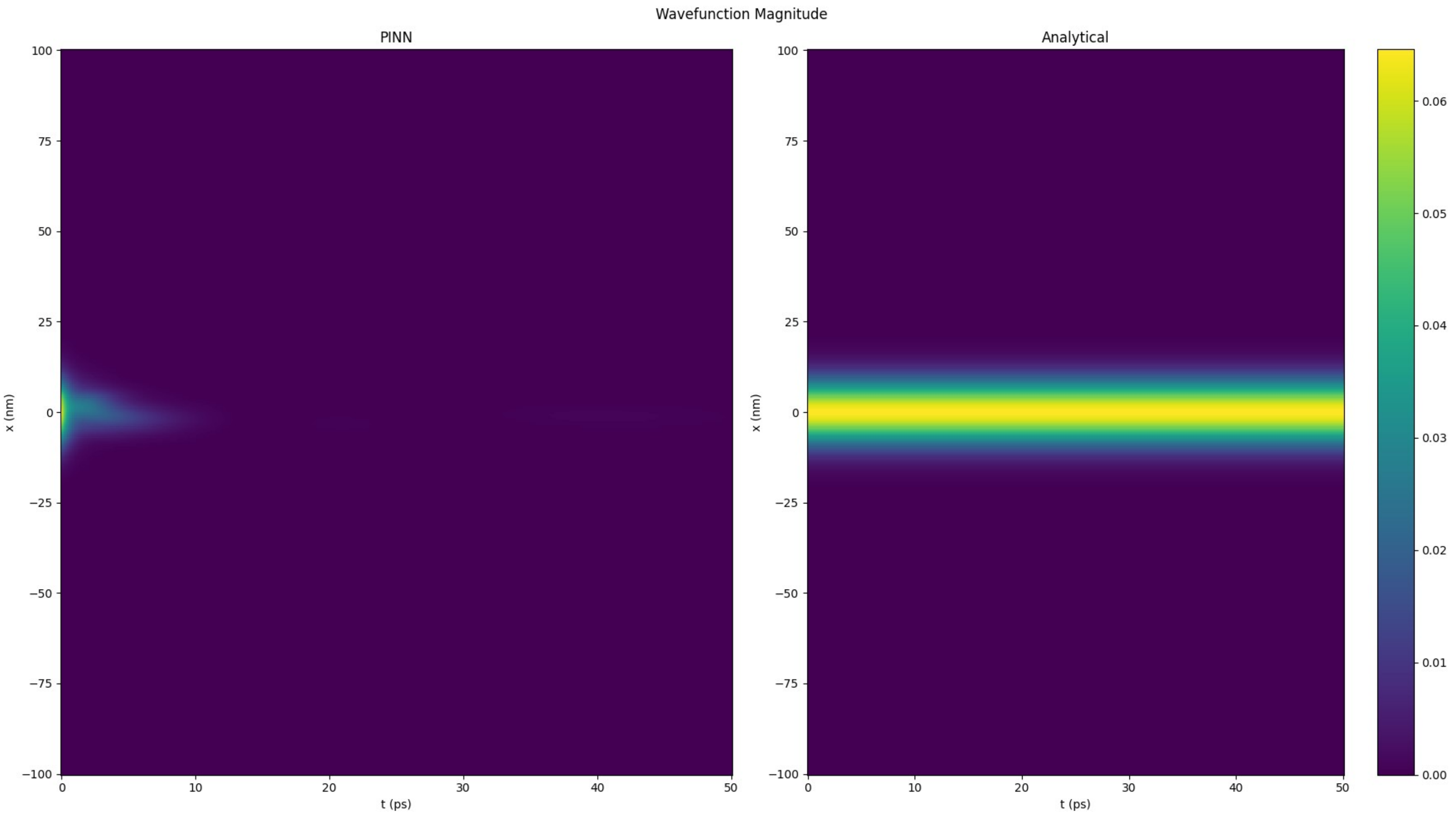
- Neural Network Setup:
  - Inputs: Position x and time t
  - Outputs: Real and Imaginary part of ψ
  - Activation functions: tanh
  - Hidden Layers: 64, 128, 128, 64
  - Optimiser: Adam

Neural Network Representation of the Schrödinger Equation

- Loss Function:
  - Physics loss:
    - MSE of $Residual = i\hbar \frac{\partial \psi(x,t)}{\partial t} + \frac{\hbar^2}{2m}\frac{\partial^2 \psi}{\partial x^2} - \frac{1}{2} m \omega^2 x^2 \psi(x,t)$
  - Initial condition loss:
    - MSE of $\psi_{nn}(x,t) - (\frac{m\omega}{\pi\hbar})^{\frac{1}{4}} e^{-(\frac{m\omega}{2\hbar}x^2)}$
  - Boundary condition loss: MSE of ψ at spacial boundaries = 0


Wavefunction at t = 0.00 ps
PINN
Analytical
Real part
Imag part
Magnitude
Wavefunction
x (nm)
0.00
0.05
0.10
0.15
0.20
0.25
−100
−75
−50
−25
0
25
50
75
100


Wavefunction at t = 12.53 ps
PINN
Analytical
Real part
Imag part
Magnitude
Wavefunction
x (nm)
0.020
0.015
0.010
0.005
0.000
−0.005
−0.010
−0.015
−0.020
0.2
0.1
0.0
−0.1
−0.2
−100
−75
−50
−25
0
25
50
75
100



Wavefunction Magnitude
PINN
Analytical
x (nm)
t (ps)
100
75
50
25
0
−25
−50
−75
−100
0
10
20
30
40
50
0.06
0.05
0.04
0.03
0.02
0.01
0.00

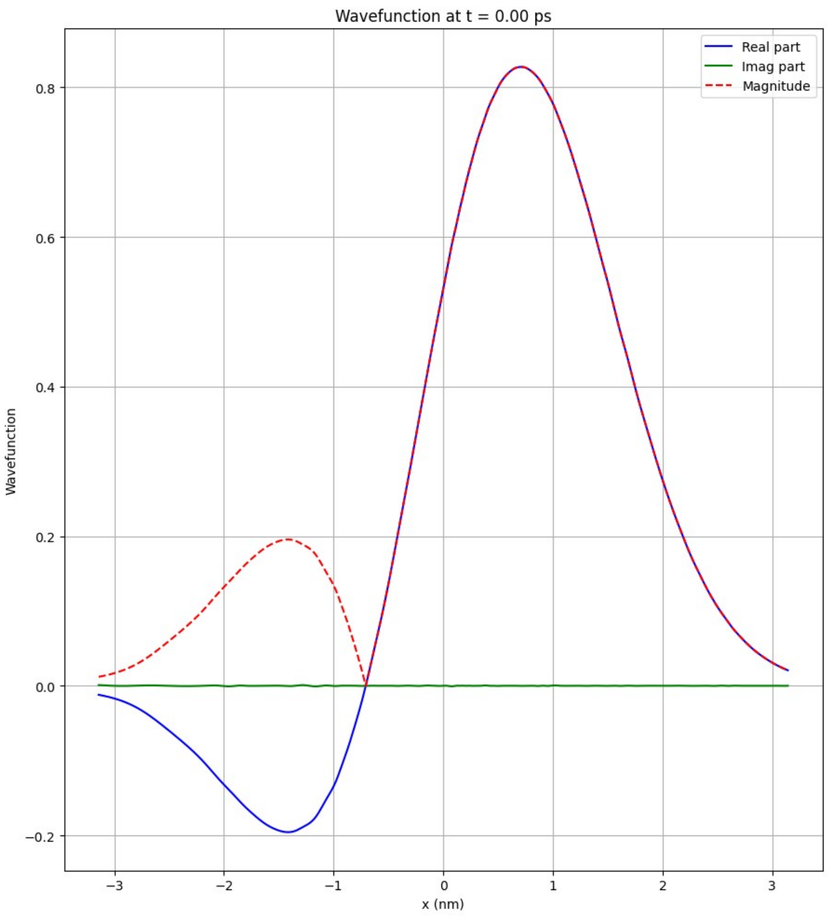
Reference Paper:PINN as Solvers for the Time-Dependent Schrödinger Equation

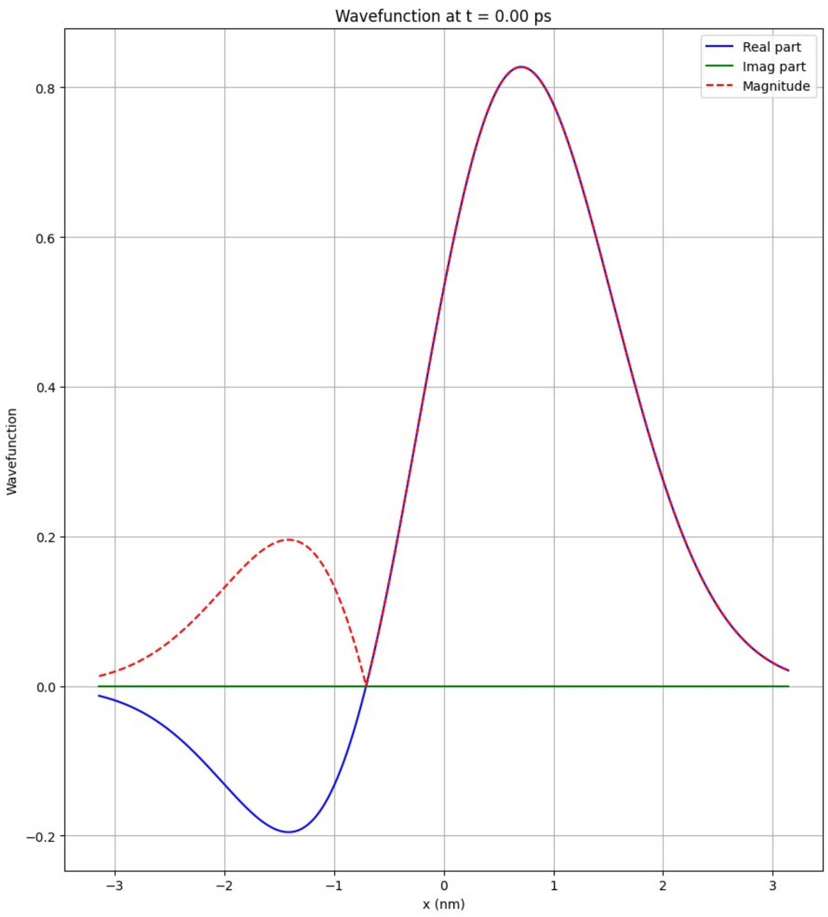
- Demonstrates the use of PINNs for solving the 1D time-dependent Schrödinger equation (TDSE) on quantum harmonic oscillator dynamics.
- Investigates generalisability, larger time domains, and higher energy states.
- Baseline results:
  - PINN accurately reconstructs superposition states of the harmonic oscillator.
  - Mean Squared Error (MSE) $\approx 10^{-5}$ for probability density.
- Their methodology and results might help guide my single-electron TDSE implementation.

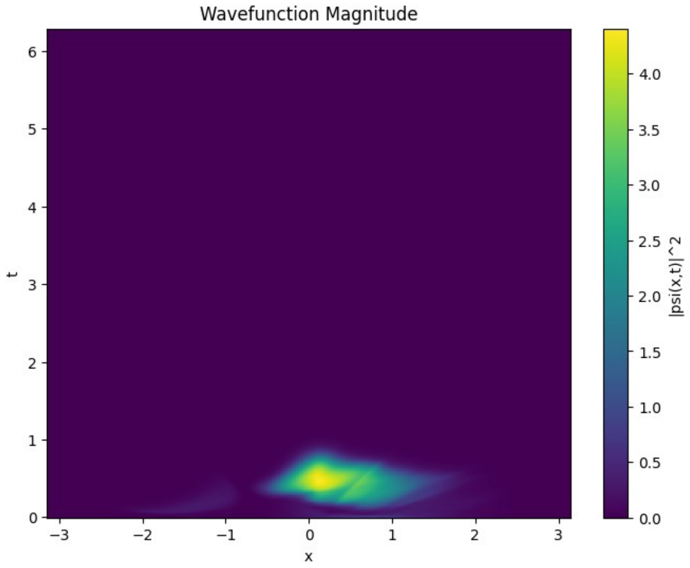
Reference Paper: Neural Network Architecture

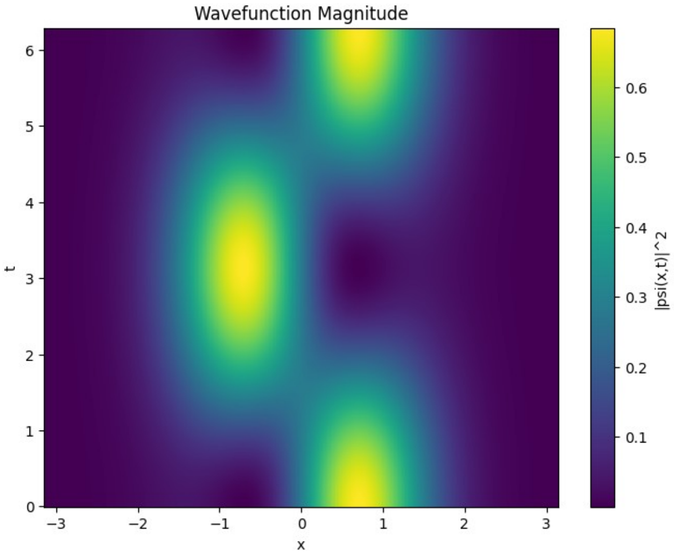
- They used a batch size of 3140 points for the interior, 200 points for the boundary conditions and 314 points for initial conditions.
- A fully-connected network with 6 layers consisting of 512 neurons each.
- We use the ADAM optimiser with ß1 = 0.09, ß2 = 0.999.
- The learning rate is initialised at α0 = 0.001 with exponential decay rate γ = 0.9 at decay steps t_γ = 2000 training steps with schedule $\alpha_t = \alpha_0\,\gamma^{t/t_\gamma}$
- The analytical solution for the baseline is

$$\phi_{0,1}(x,t) = \frac{1}{\sqrt{2}}\sqrt[4]{\frac{\omega}{\pi}}\exp\left(-\frac{\omega x^2}{2}\right)\left(exp\left(-i\frac{\omega}{2}t\right) + exp\left(-i\frac{3\omega}{2}t\right)\sqrt{2\omega}x\right).$$


Wavefunction at t = 0.00 ps
Real part
Imag part
Magnitude
Wavefunction
x (nm)
−3
−2
−1
0
1
2
3
−0.2
0.0
0.2
0.4
0.6
0.8


Wavefunction at t = 0.00 ps
Real part
Imag part
Magnitude
Wavefunction
x (nm)
−3
−2
−1
0
1
2
3
−0.2
0.0
0.2
0.4
0.6
0.8


Wavefunction Magnitude
t
x
|psi(x,t)|^2


Wavefunction Magnitude
t
x
|psi(x,t)|^2