DSECOP Module: Learning the Schrödinger Equation

Karan Shah
DSECOP Fellow
www.casus.science
Goals

• Relevant course: Quantum Mechanics 1 (Usually Junior year) (after students learn about Quantum Harmonic Oscillator)

• Physics goals:
  • Introduction to Time-Dependent Schrödinger Equation
  • Converting analytical solutions to code

• Machine learning goals:
  • Introduction to neural networks
  • Integrating physics domain knowledge into ML algorithms
Structure

• Lesson 1: Introduction to Neural Networks
• Lesson 2: Brief background on machine learning and applications to physics
• Lesson 3: Solving the Time-Dependent Schrödinger Equation for a Quantum Harmonic Oscillator, using machine learning

• Components:
  • In-built interactive demonstrations and exercises
  • Take-home reading and reference
  • Project ideas (trivial to ambitious)
Lesson 1

Introduction to Neural Networks
(with plumbing and colours)
Lesson 2

Broad introduction to machine learning

• Background for machine learning

• Brief explanation of:
  • Parts of ML workflow
  • Different ML models
  • Deep learning

• Applications to physics, and material to explore further (~70 references)
A PINN is constructed for the solution of the Time-Dependent Schrödinger Equation
\[ i \frac{\partial}{\partial t} \psi(x, t) - \hat{H} \psi(x, t) = 0 \]
in the domain \( x \in (-\pi, \pi), t \in (0, 2\pi) \).

The Hamiltonian is given by
\[ \hat{H} = -\frac{1}{2} \frac{\partial^2}{\partial x^2} + \frac{\omega^2}{2} x^2 \]

The analytical solution \( \psi_{mn}(x, t) \in \mathbb{C} \) is
\[ \psi_{mn}(x, t) = \frac{1}{\sqrt{2}} \left( e^{-iE_m t} \phi_m(x) + e^{-iE_n t} \phi_n(x) \right) \]
where \( \psi_{mn} \) is the wavefunction for a QHO consisting of the superposition of eigenstates \( \phi_m \) and \( \phi_n \) with \( E_i \) being the energy level of state \( \phi_i \).

The inputs of the PINN solver are \( x, t \) and \( \omega \), with the outputs being \( u, v \in \mathbb{R} \), where \( u = \text{Re}(\psi) \) and \( v = \text{Im}(\psi) \) for a QHO with frequency \( \omega \).
Lesson 3

$x - t$ snapshot for true and predicted values for $\psi_{0,1}$ with $\omega = 1.0$.

$MSE_u = 1.60e-5$, $MSE_v = 1.37e-5$
$x-t$ snapshot for true and predicted values for $\psi_{0,1}$ with $\omega = 1.0$.

$MAE_u = 1.60e-3, MAE_v = 1.37e-3$

$|\psi|^2$ snapshot for true and predicted values for $\psi_{0,1}$ with $\omega = 1.0$.

$MAE_u = 0.27, MAE_v = 0.49$
Lesson 3

For a system $f$, with solution $u(x, t)$, governed by the following equation

$$f(u) := \frac{\partial u}{\partial t} + \mathcal{N}[u; \lambda], \quad x \in \Omega, \quad t \in [T_0, T_f]$$

$$f(u) = 0$$

where $\mathcal{N}[u; \lambda]$ is a differential operator parameterised by $\lambda$, $\Omega \in \mathbb{R}^D$, $x = (x_1, x_2, \ldots, x_D)$

with boundary conditions

$$\mathcal{B}(u, x, t) = 0 \text{ on } \partial \Omega$$

and initial conditions

$$\mathcal{I}(u, x, t) = 0 \text{ at } T_0$$

PDE Loss

Backpropagation

PINN Architecture

Input

Neural Network (Trainable Parameters)

Output (PDE Solution)

PDE Loss

PDE
Lesson 3

We construct $u_{\text{net}}$, a surrogate model for the true solution $u$.

$$f_{\text{net}} = f(u_{\text{net}})$$

The constraints imposed by the system are encoded in the loss term $L$ for neural network optimisation.

$$L = L_f + L_{BC} + L_{IC}$$

where $L_f$ denotes the error in the solution within the interior points of the system. This error is calculated for $N_f$ collocation points.

$$L_f = \frac{1}{N_f} \sum_{i=1}^{N_f} \left| f_{\text{net}}(x_i^f, t_i^f) \right|^2$$

$$L_{BC} = \frac{1}{N_{BC}} \sum_{i=1}^{N_{BC}} \left| u(x_{BC}^i, t_{BC}^i) - u_i \right|^2$$

$$L_{IC} = \frac{1}{N_{IC}} \sum_{i=1}^{N_{IC}} \left| u(x_{IC}^i, t_{IC}^i) - u_i \right|^2$$

$L_{BC}$ and $L_{IC}$ represent the constraints imposed by the boundary and initial conditions, calculated on a set of $N_{BC}$ boundary points and $N_{IC}$ initial points respectively, with $u_i$ being the ground truth.
Lesson 3

TD SE Results

FCN: MAE (density): 3.8673

PINN: MAE (density): 0.0010
Lesson 3

- Advantages of PINNs:
  - Mesh free nature: Generate solutions for grids of arbitrary resolution
  - Hybrid workflow: Generate extremely fast coarse solutions, further polished by iterative numerical schemes
  - Automatic Differentiation: Well suited for integration into ML workflows
  - Generalisable across PDE parameters. Train once, solve a large class of PDEs

Disadvantages of PINNs:
- For low dimensional problems, numerical approaches are faster with theoretical guarantees
- Lack of interpretability / Black box algorithm
- Learning high-resolution higher-dimensional system is resource intensive. However, once learnt, inference is very quick on that domain
Lesson Plan

• Take home - RobotPlumber exercise (2 hours)
• In class - General discussion of machine learning, applications in physics (1-2 hours)
• In class - TD Schrodinger Equation and PINN theoretical background (1-2 hours)
• Take home - Go through notebook (2 hours)
• Project (2 - 8 hours depending on the scope)
Conclusion

- Module can be used for a Quantum Mechanics course
- Based on feedback, easy to add other potentials like infinite square well
- First two lessons can be used for general ML information, third application module can be adapted to any course with a differential equation

The module is available under the DSECOP GitHub repository
Link:
https://github.com/GDS-Education-Community-of-Practice/DSECOP/tree/main/Learning_the_Schrodingers_Equation
Thank you
Questions Comments Concerns?

Karan Shah
k.shah@hzdr.de

Feedback form: