Title: PINN Applied for Solving the Van der Pol Oscillator
Date: 9-11-2024

Introduction

The Van der Pol oscillator is a second-order nonlinear ordinary differential equation that models systems with damping characteristics. It is frequently applied in fields such as electrical circuits and biological systems. Traditional methods to solve this equation include both analytical and numerical solutions, which can be computationally intensive, particularly for systems with strong nonlinearities. This approach utilizes a Physics-Informed Neural Network (PINN) to approximate the solution to the Van der Pol oscillator, potentially reducing computational requirements while offering solution flexibility.

Mathematical Formulation

Differential Equation:
The Van der Pol equation is given by:

$$d2y/dt2-\mu(1-y2)dy/dt+y=0$$

where μ is a scalar parameter that influences the system's nonlinearity and damping characteristics. Initial conditions are generally set by specifying values for y(0) and dy/dt(0).

Analytical or Classical Approach:
The classical approach to solving this equation typically employs numerical solvers, such as MATLAB's ode45 and SciPy's solve_ivp, which use adaptive step sizes to enhance numerical accuracy. However, with strong nonlinearities (high μ), this discretization process can become complex.

Assumptions:
The model assumes a constant μ value during training and does not account for external forces. Additionally, simplifications were made to the initial and boundary conditions to stabilize training.

PINN Architecture Design

Neural Network Architecture
The PINN architecture used in this work is a fully connected network with three layers:

- **Input layer:** Takes three inputs: time t, position y, and velocity v.
- **Hidden layers:** Consists of two hidden layers with 128 and 64 neurons, respectively, each with ReLU activation functions.
- **Output layer:** Outputs a single value representing d2y/dt2.

Training Procedure:
The network was trained using the Adam optimizer with a learning rate of 0.001 over 10,000 epochs. Training data consisted of time values within the initial condition range, and synthetic data generation was applied to enforce constraints.

Implementation

Tools and Libraries Used:

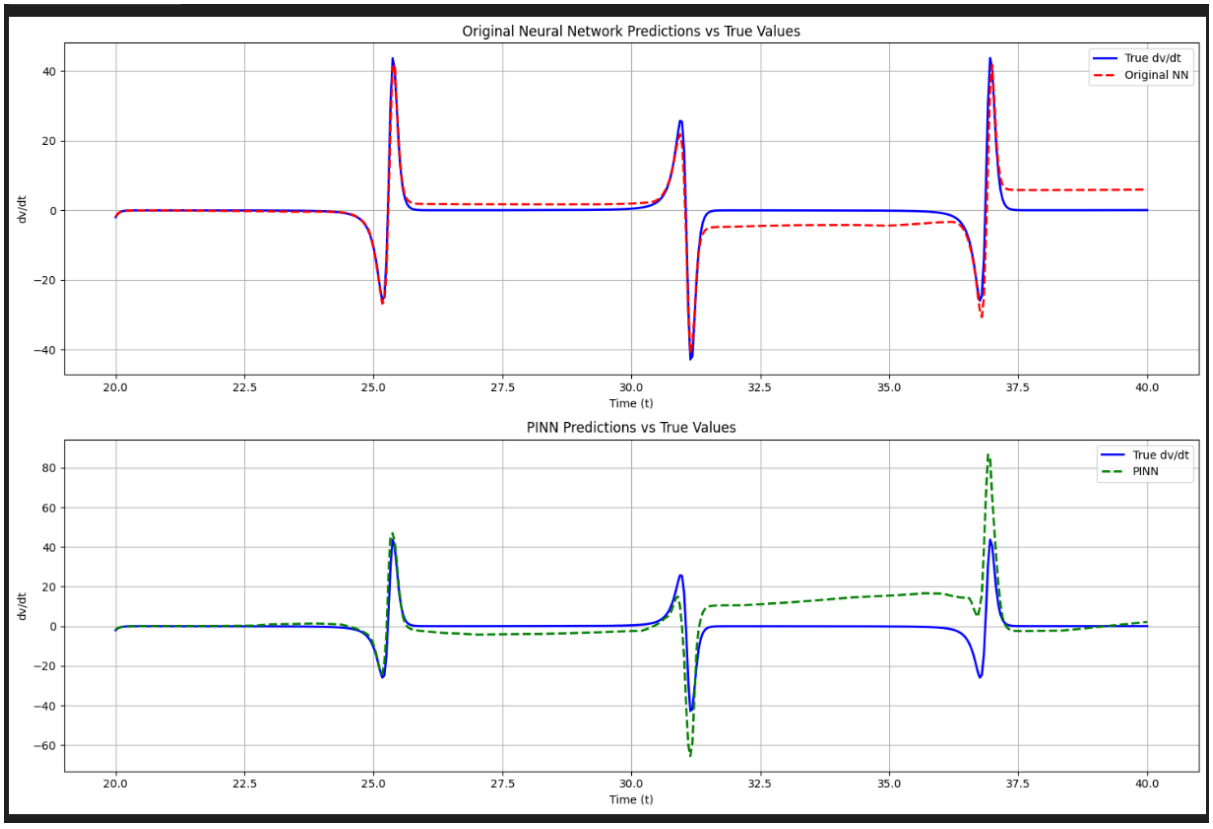
- **PyTorch** for neural network modeling and training
- **NumPy** for data manipulation

- **SciPy** for comparison with classical solvers
- **Matplotlib** for visualizing results

Code Description:

1. **Data Preparation:** Generated synthetic data for t, y, and v.
2. **Model Definition:** Defined a PINN class with layers and a forward pass to compute d2y/dt2 .
3. **Loss Function:** Calculated the physics loss to satisfy the Van der Pol differential equation.
4. **Training Loop:** Optimized the model to minimize the loss, with frequent updates and loss tracking.

Difficulties Encountered:
The main challenge was stabilizing the training process due to the equation's sensitivity and strong nonlinearity. Initial attempts showed poor convergence, which was improved by adjusting the learning rate and increasing the number of training epochs.



Results

The PINN was able to approximate the dynamics of the Van der Pol oscillator effectively for moderate μ values. Graphs comparing the outputs from the PINN and traditional solvers aligned well, though discrepancies were observed at higher μ values, where traditional solvers displayed greater stability.

Conclusion

The PINN provided a reasonable solution for the Van der Pol oscillator, demonstrating the viability of using PINNs to solve differential equations without explicit data. While the model performed well for

moderate nonlinearity, its performance declined at higher nonlinearity levels, highlighting a limitation of the current PINN approach in handling extreme conditions.

Future Improvements:

- Adaptive loss weighting to better capture high-nonlinearity effects.
- Experimentation with alternative architectures or more complex activation functions.
- Use of hybrid approaches, where the PINN is combined with traditional solvers to improve stability and accuracy.