# Smart Reactor : Machine Learning (V2)

*Undergraduate project - Semester 2024-25-I*

---


## Mentees:

**Ankit Kumar (220159)**
**Komal Kumari (220540)**


---

## Contents

# 1 Overview

The main objective of our project was to develop a machine learning model capable of predicting the kinetic behavior of a reaction using experimental data.

- Throughout the project, we explored the fundamentals of neural networks
- Gained experience in solving forward PINNS
- Learned implementation of inverse problems with PINNs.
- We applied this knowledge to the Belousov-Zhabotinsky (BZ) reaction, which was provided to us as a case study.

However, the actual task presented a considerable challenge, as it involved a higher degree of stiffness and complex equations beyond what we had practiced. This required us to adapt our approach and deepen our understanding of working with PINNs in handling stiff and intricate systems.

# 2 PINNs

Physics-Informed Neural Networks (PINNs) solve differential equations by embedding physical laws into a neural network's loss function. The loss combines residuals from the governing equations and boundary/initial conditions, ensuring adherence to the physics. PINNs are efficient, requiring minimal data, and versatile, handling forward and inverse problems across fields like fluid dynamics, biology, and quantum mechanics. They excel in solving complex, high-dimensional systems where traditional methods may struggle.

# 3 Regression Problem using TensorFlow

We have used Artificial Neural Network (ANN) to perform regression on a synthetic dataset. The goal is to model a quadratic relationship between the input variable x and the output y, given by the equation:

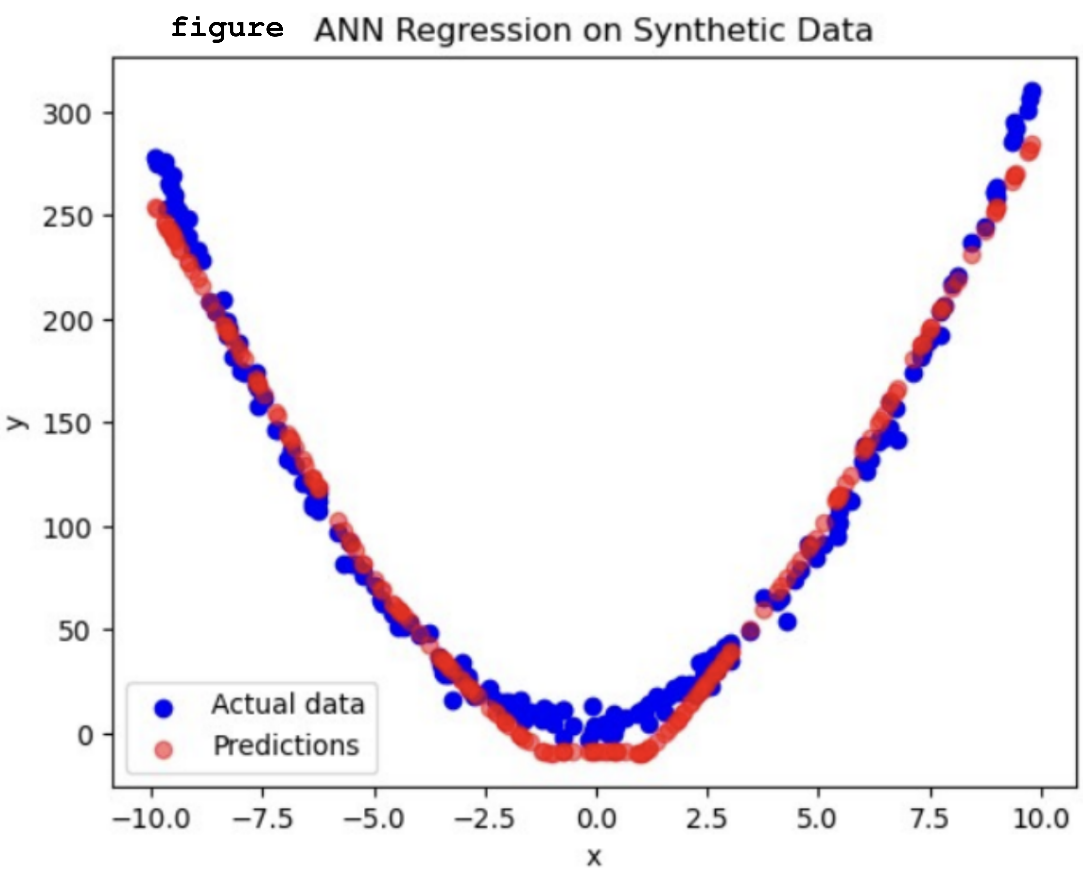
$$y = 3x2 + 2x + 5 + \epsilon \tag{1}$$

where:

- $x$ is randomly sampled from a uniform distribution in the range $[-10, 10]$.

- $\epsilon$ is random noise sampled from a normal distribution $N(0,5)$ to simulate real-world data variability.

**figure**



# 4 Forward Problem Using PINNs

PINNs solve forward problems of differential equations by embedding the governing physics directly into the neural network's loss function. In a forward problem, the objective is to find unknown functions that satisfy given differential equations with specific initial and boundary conditions.

- **Define the Domain and Differential Equations:** Specify the problem's spatial/temporal domain and the differential equations that govern it.
- **Neural Network Setup:** Create a neural network that takes domain points (e.g., time or spatial coordinates) as inputs and outputs the approximated solution.
- **Physics-Based Loss Function:** solution. Physics-Based Loss Function: Incorporate three loss terms|differential equation residuals, initial conditions, and boundary conditions|into a single loss function. Automatic differentiation is used to compute derivatives for the differential equation residuals.
- **Training:** Optimize the network's weights to minimize the total loss, ensuring that the network output satisfies the differential equations and conditions.

- **Solution Extraction** After training, the network approximates the solution across the domain.

### 4.1 Example: Sine and Cosine Functions

**Functions:**

$$y_1(x) = \sin(x)$$
$$y_2(x) = \cos(x)$$

**ODE System:**

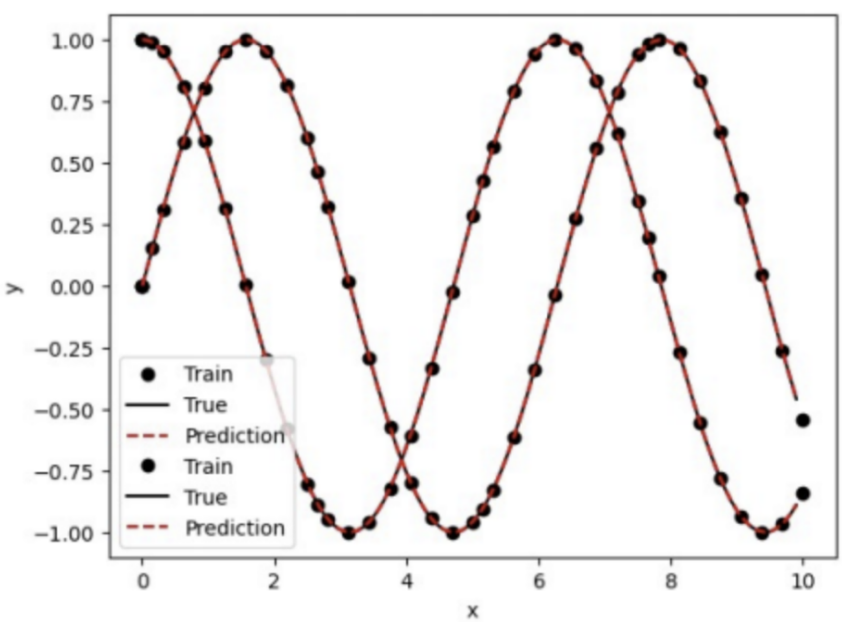
$$\frac{dy_1}{dx} = y_2$$
$$\frac{dy_2}{dx} = -y_1$$

**Description:**

In this problem, we aim to solve a system of ordinary differential equations (ODEs) representing simple harmonic oscillators. The functions involved are sine and cosine, which are solutions to the ODE system. Specifically, we are trying to model two functions that satisfy the above differential equations.

The initial conditions at $x = 0$ are:

$$y_1(0) = 0, \quad y_2(0) = 1$$



## 4.2 Example: Damped Harmonic Oscillator

### System Description

The damped harmonic oscillator is described by the following second-order differential equation:

$$\frac{d^2 y}{dt^2} + 2\gamma \frac{dy}{dt} + \omega^2 y = 0$$

Where:

- ( y(t) ) is the displacement as a function of time.
- ( \gamma ) is the damping coefficient, which represents the strength of the damping (friction).
- ( \omega ) is the natural frequency of the system.

In this example, we rewrite the second-order ODE as a system of two first-order ODEs by defining:

$$y_1(t) = y(t)$$

$$y_2(t) = \frac{dy}{dt}$$

The system of ODEs then becomes:

$$\frac{dy_1}{dt} = y_2$$

$$\frac{dy_2}{dt} = -2\gamma y_2 - \omega^2 y_1$$

### Initial Conditions

For this example, we set the initial conditions as:

$$y_1(0) = 1 \quad \text{(initial displacement)}$$

$$y_2(0) = 0 \quad \text{(initial velocity)}$$

### Target Solution (Damped Oscillations)

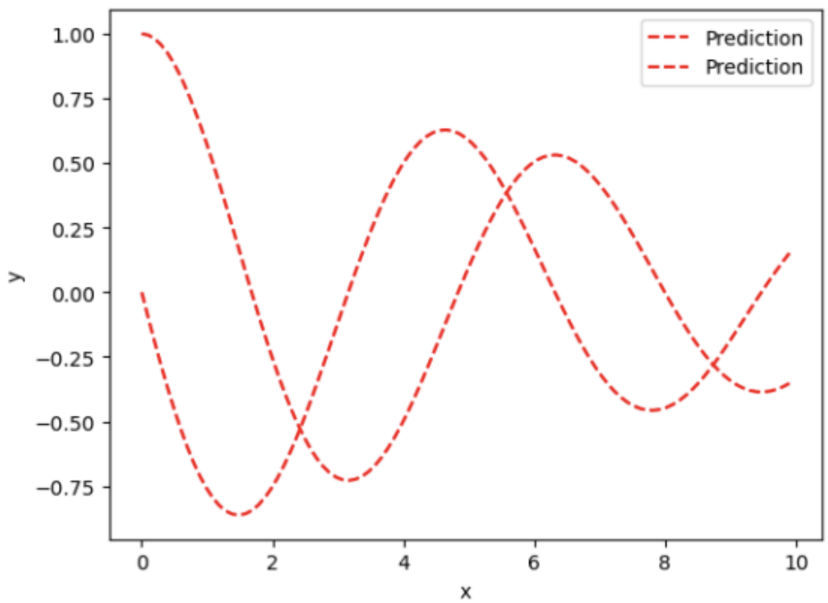
The analytical solution to this system is given by:

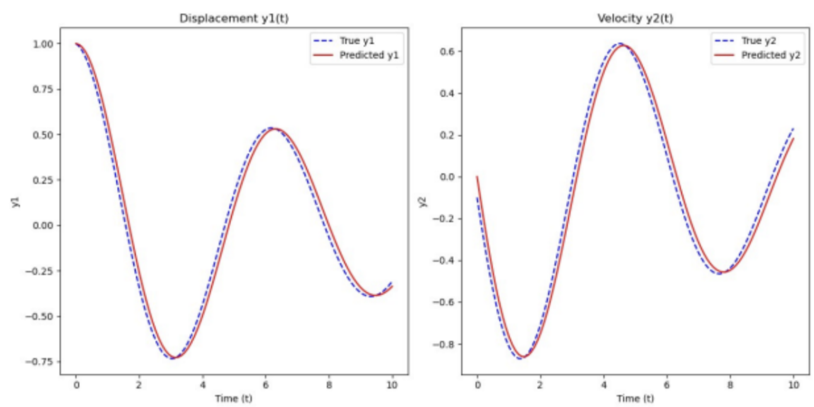
$$y(t) = e^{-\gamma t} \cos(\omega t)$$

Where:

$$y_1(t) = e^{-\gamma t} \cos(\omega t)$$

$$y_2(t) = -\gamma e^{-\gamma t} \cos(\omega t) - \omega e^{-\gamma t} \sin(\omega t)$$




# 5 Forward PINNs on BZ Reaction

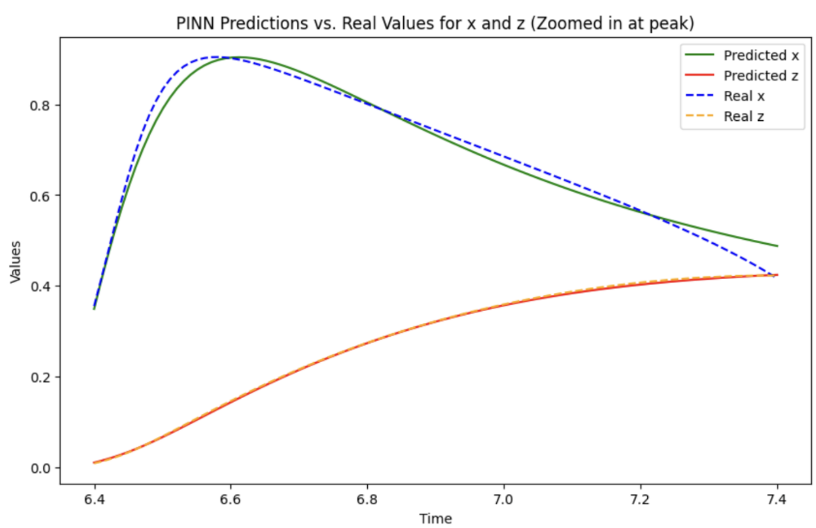
## Description

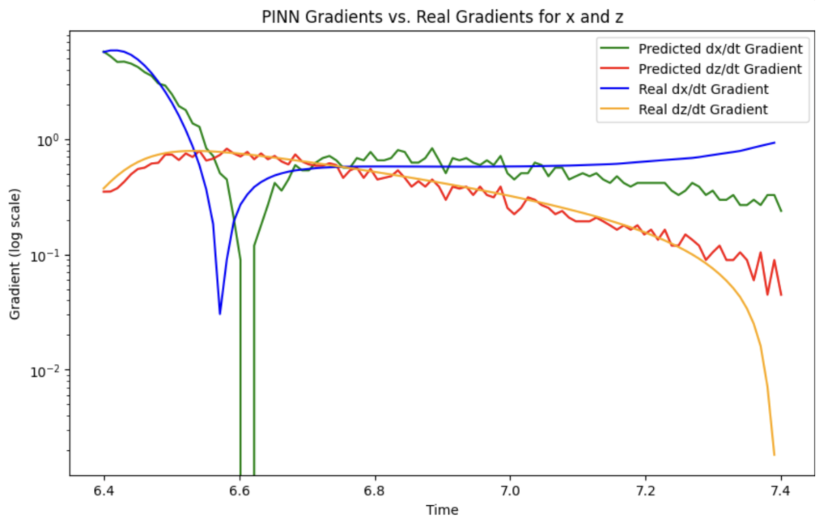
The Belousov-Zhabotinskii (BZ) reaction is a well-known oscillating chemical reaction, where periodic changes in concentration cause visible color changes. This behavior arises from the periodic oscillation of reactants between different oxidation states. Due to its complexity, the reaction is often modeled using the Oregonator, a simplified system of nonlinear differential equations. These equations are stiff, making analysis difficult. By nondimensionalizing and reducing the system, researchers study a simplified 2x2 model that exhibits relaxation oscillations. The goal is to solve this reduced system using Physics-Informed Neural Networks (PINNs) to analyze its oscillatory behavior and compare it with experimental data.

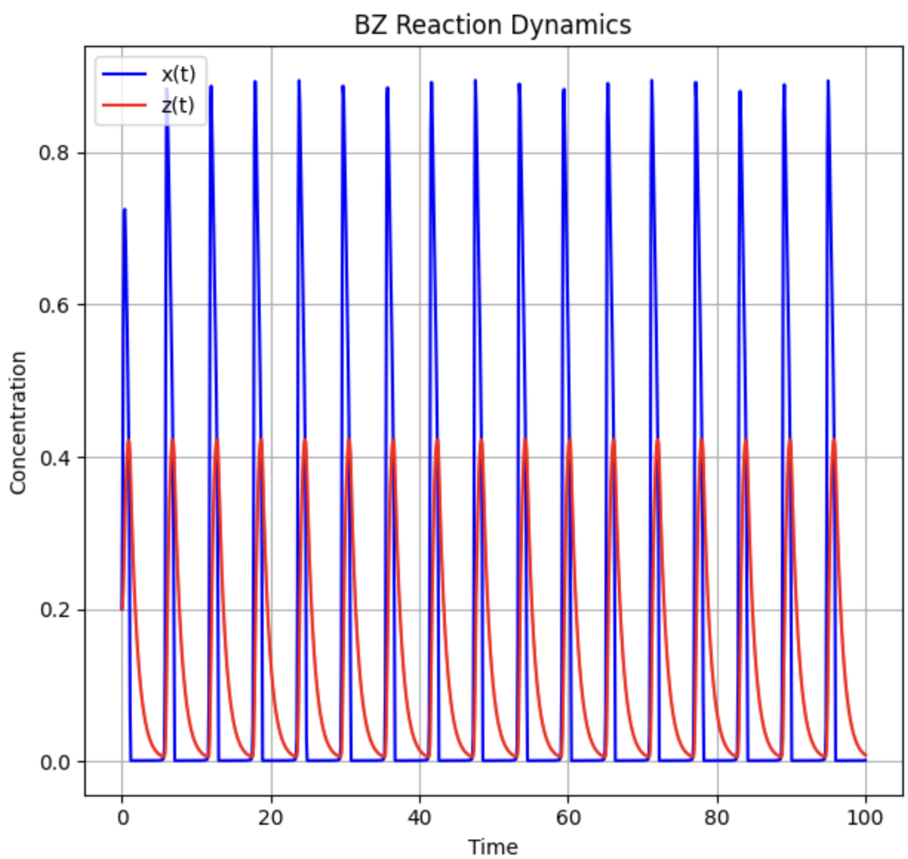
$$\varepsilon \frac{dx}{d\tau} = x(1-x) + \frac{f(q-x)}{q+x} z \equiv g(x,z),$$
$$\frac{dz}{d\tau} = x - z \equiv h(x,z).$$

Figure 1: Oregonator Model for BZ Reaction

The reduced set of equations are (f=0.67, $\epsilon$=0.04, $q$=8e-04).




PINN Gradients vs. Real Gradients for x and z
Predicted dx/dt Gradient
Predicted dz/dt Gradient
Real dx/dt Gradient
Real dz/dt Gradient
Gradient (log scale)
Time


BZ Reaction Dynamics
x(t)
z(t)
Concentration
Time

# 6 Inverse Problem on PINNs

Physics-Informed Neural Networks (PINNs) solve inverse problems by combining physical laws (encoded as PDEs/ODEs) with observed data to infer unknown parameters or conditions. A neural network approximates the solution and unknowns, while a loss function enforces both the governing equations (physics loss) and data consistency (data loss). During training, the network learns to satisfy the physical model and match observed data, enabling the estimation of unknowns and the reconstruction of the system's state. PINNs are efficient, flexible, and handle sparse or noisy data effectively.

Steps in Solving Inverse Problems with PINNs:

- **Define the Domain and Observations:** Specify the spatial/temporal domain and use available data points.
- **Neural Network Setup:** Build a neural network that incorporates the unknown parameters.
- **Loss Function for Parameter Estimation:** Include terms for equation residuals and fit to observational data.
- **Training:** Optimize both network weights and parameters to minimize the loss.

## 6.1 Example: Diffusion Equation

Consider the following diffusion equation with a source term:

$$\frac{\partial u}{\partial t} = C\frac{\partial^2 u}{\partial x^2} + e^{-t}\left(\sin(\pi x) - \pi^2 \sin(\pi x)\right) \tag{2}$$

where $x \in [-1,1]$, $t \in [0,1]$, and $C$ is the unknown diffusion coefficient.

The boundary conditions are given as:

$$u(\pm 1, t) = \sin(\pi x)e^{-t} \tag{3}$$

The initial condition is:

$$u(x,0) = \sin(\pi x) \tag{4}$$

Additionally, there are measurements of $u(x,1)$ at 10 points for the time $t = 1$.

The exact solution to the equation is:

$$u(x,t) = \sin(\pi x)e^{-t} \tag{5}$$

The goal is to recover the unknown diffusion coefficient $C$ and the complete solution field $u(x,t)$ using a Physics-Informed Neural Network (PINN) approach. We will try to predict $C$ using an initial guess of $C = 2$, while the true value of $C$ is known to be $C = 1$.

The physics-informed neural network successfully recovered both the unknown parameter $C$ and the solution field $u(x,t)$ with high accuracy. The network achieved a relative $L_2$ error of order $10^{-3}$, demonstrating that PINNs can effectively solve inverse problems in partial differential equations.

## 7 Inverse Problem on BZ Reaction

Building upon the forward problem for BZ reaction we solved above, we now address the inverse problem for the reduced Oregonator system. We would solve this by using synthetic data by solving using a numerical solver for stiff equations-'Radau'. We will take only a small data range because the stiffness of the equation highly increases the computational power needed to solve the system for a bigger range and more number of oscillations. We would limit ourselves to just a single oscillation.

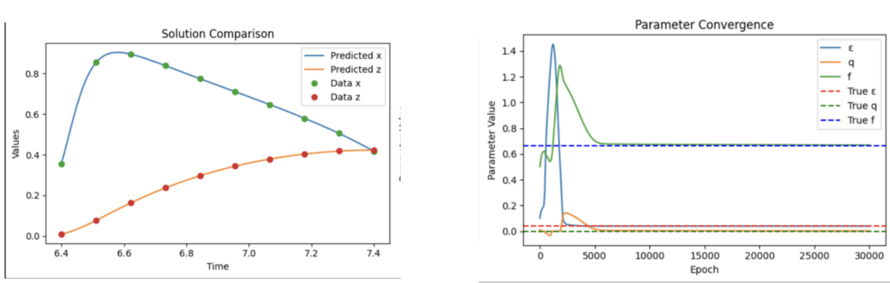



Figure 2: performances of inverse pinn

### 7.2 Model Performance and Parameter Estimation

For a small number of epochs, the model successfully estimates the parameters $f = 0.669743$, $\epsilon = 0.039910$. However, it struggles to solve for $q$. For lower to medium numbers of epochs, the estimated value of $q$ ranges from $0.002$ to $0.0002$, whereas the real value is $0.0008$. This discrepancy can be resolved by increasing the computational capacity of the machine being used and performing a higher number of epochs to get values as close to the true value as $0.0007993$.

## 8 References


- https://scholar.rose-hulman.edu/cgi/viewcontent.cgi?article=1286&context=rhumj
- https:https://arxiv.org/pdf/2011.04520
- https://arxiv.org/pdf/2407.10836
- https://amses-journal.springeropen.com/articles/10.1186/s40323-024-00265-3