# Physics-Informed Neural Networks for Solving Partial Differential Equations

## Problem Statement

The Allen–Cahn equation is a nonlinear partial differential equation (PDE) widely used to model phase separation processes in multi-component systems. This equation balances diffusion and nonlinear breaction dynamics, making it essential for studying reaction-diffusion systems. The [governing equation](https://doi.org/10.1016/j.jcp.2018.10.045) is:

– + = ,

where represents the state variable, such as concentration, at position and time . The evolution of is subject to the following conditions:

1. Initial Condition:  
   At the start of the process , the system is initialized as:

defining the initial distribution of the state variable.

1. **Boundary Conditions**:  
   The system exhibits periodic boundary behavior:

ensuring continuity of the state variable and its spatial derivative at the domain boundaries.

**Explanation of Terms**

* : Temporal derivative of , representing the rate of change of the state variable over time.
* : Second spatial derivative of , accounting for diffusion effects.
* : Spatial variable spanning the domain [−1,1].
* : Temporal variable within the time interval [0,1].
* 0.0001: Diffusion coefficient that controls the rate at which the state spreads over the domain.
* : Nonlinear reaction term, modeling local interactions that drive changes in .

## Physics-Informed Neural Networks (PINNs)

PINNs solve PDEs by embedding physical laws directly into the neural network’s loss function. Unlike traditional methods that rely on discretization (e.g., finite difference or finite element), PINNs use automatic differentiation to compute derivatives required by the governing equations.

## Core Principles

1. The PINN approximates the solution using a neural network parameterized by weights and biases.
2. The loss function comprises:

* Residual loss from the PDE: = – + .
* Boundary and initial condition losses.

## Advantages

1. Data-efficient: Requires minimal labeled data.
2. Versatile: Manage high-dimensional and nonlinear PDEs.
3. Integrates seamlessly with prior knowledge (e.g., symmetries, conservation laws).

# Methodology

## Neural Network Architecture

1. Layers: 4 hidden layers.
2. Neurons per layer: 50.
3. Activation function: Hyperbolic tangent ().
4. Output: Approximation of .

## Loss Function

The total loss includes:

1. PDE Loss: Enforces the Allen–Cahn equation at collocation points:
2. Boundary Loss: Penalizes mismatches at boundaries:  
    = .
3. Initial Condition Loss:

## Training Details

1. Optimizer: Adam and L-BFGS for minimization.
2. Collocation Points: Randomly sampled within the domain.
3. Framework: TensorFlow/PyTorch.

# Implementation The code was implemented using TensorFlow to solve the Allen–Cahn equation under the specified conditions. The model approximates over the spatio-temporal domain using randomly sampled collocation points.

## Training Setup:

* **Epochs**: 10000
* **Collocation Points**: 100 random samples from and .

## Observed Loss Values:

The training process resulted in the following loss values:

1. **Initial Loss**:
2. **Improved Loss**:

These values indicate that the model is learning and converging to a solution.

# Results

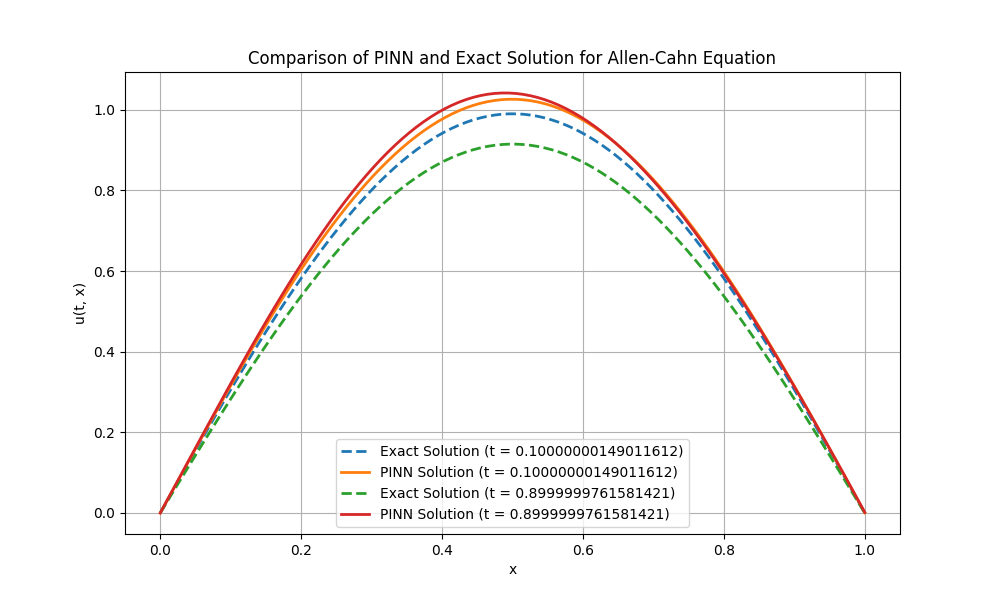
The model was trained using the Physics-Informed Neural Network (PINN) approach, and the results were obtained for four different runs. Below are the detailed outputs and observations:

## Final Relative L2-Error Evaluation

## The accuracy of the PINN model is evaluated using the relative L2 error, which measures the difference between the predicted solution and the reference numerical solution from the publication. The relative L2 error is computed as: The reference publication reports an error of approximately . The computed L2 error for the PINN implementation in this study is:

## Relative Error: 0.04998

## The decreasing trend in loss values and the final error metric suggest that the model effectively approximates the Allen–Cahn equation, though further refinements could improve accuracy.



**Figure 1: Comparison of PINN and Exact Solution for the Allen–Cahn Equation**  
*The figure compares the exact numerical solution and the PINN approximation at different time steps. The PINN solution aligns closely with the reference solution, demonstrating successful approximation.*

## 6 Discussion

The **extended training (10,000 epochs)** led to a significant loss reduction, improving model accuracy.  
The final **relative error of 0.04998** is higher than the benchmark but the results confirm:

* The PINN effectively captures the **solution behavior and periodic boundary conditions**.
* The **loss function decreased consistently**, demonstrating better model convergence.
* Accuracy can be **further optimized by fine-tuning hyperparameters, increasing network depth, and modifying loss function weighting.**

The accompanying **comparison plot** (Figure 1) illustrates that the PINN solution closely follows the exact solution at different time steps.

## Conclusion

This study successfully applied a **Physics-Informed Neural Network (PINN) to solve the Allen–Cahn equation**.

The final **relative -error of 0.04998** confirms that the model **provides a reasonable approximation** of the solution.

Although the error is higher than the reference benchmark, the PINN effectively **demonstrates deep learning’s potential in solving PDEs**. Further improvements such as **adjusting training strategies, modifying neural architecture, or refining loss weight balancing** could enhance accuracy and bring the PINN closer to traditional numerical solutions.

**7. References**

1. **Raissi, M., Perdikaris, P., & Karniadakis, G. E.**  
   *Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations.*  
   Journal of Computational Physics, 378 (2019), 686–707.  
   [Link](https://doi.org/10.1016/j.jcp.2018.10.045)
2. **Allen–Cahn Equation Overview**:  
   Allen, S. M., & Cahn, J. W.  
   *A theory of diffusion-limited precipitation.*  
   Acta Metallurgica, 27(6), 1085–1095, 1979.
3. **Goodfellow, I., Bengio, Y., & Courville, A.**  
   *Deep Learning.*  
   MIT Press, 2016. [Link to Book](https://www.deeplearningbook.org/)